Methods in Molecular Biology 1529

Springer Protocols

Ilan Samish *Editor*

Computational Protein Design

Humana Press

METHODS IN MOLECULAR BIOLOGY

Series Editor
John M. Walker
School of Life and Medical Sciences
University of Hertfordshire
Hatfield, Hertfordshire, AL10 9AB, UK

For further volumes:
http://www.springer.com/series/7651

Computational Protein Design

Edited by

Ilan Samish

Department of Plants and Environmental Sciences
Weizmann Institute of Science, Rehovot, Israel

Department of Biotechnology Engineering
Braude Academic College of Engineering, Karmiel, Israel

Amai Proteins Ltd., Ashdod, Israel

Humana Press

Editor
Ilan Samish
Department of Plants and Environmental Sciences
Weizmann Institute of Science, Rehovot, Israel

Department of Biotechnology Engineering
Braude Academic College of Engineering, Karmiel, Israel

Amai Proteins Ltd., Ashdod, Israel

ISSN 1064-3745 ISSN 1940-6029 (electronic)
Methods in Molecular Biology
ISBN 978-1-4939-6635-6 ISBN 978-1-4939-6637-0 (eBook)
DOI 10.1007/978-1-4939-6637-0

Library of Congress Control Number: 2016959982

Printed on acid-free paper

This Humana Press imprint is published by Springer Nature
The registered company is Springer Science+Business Media LLC
The registered company address is: 233 Spring Street, New York, NY 10013, U.S.A.

Preface

The aim of this first-ever book entitled *Computational Protein Design* (CPD) is to bring the latest know-how on the CPD methods in respect to the process, success, and pitfalls of the field. The book is organized so as to introduce and present the general methodology and main challenges followed by a description of specific software and applications. As seen in the description below, there is more than one way to cluster the different chapters, each highlighting a different aspect of the field.

While there has not been a book dedicated to CPD, books on protein design have often included chapters on CPD. Here, following a chapter on the framework of CPD (Chapter 1) and a summary of past achievements and future challenges (Chapter 2), a chapter on the experimental aspects of production of the designed protein is presented (Chapter 3). Beyond the need to understand the experimental aspects of the computational endeavor, this is to remind us that the final outcome of the computational process is the production of a real protein.

It is widely considered that a global minimum energy conformation (GMEC) reflects the actual native structure of the protein. The protein design process is intrinsically computationally intensive as sequence and structure space should be rigorously sampled in the search for the GMEC of the requested target. Deterministic search methods (Chapter 4) of which dead-end elimination (DEE) is among the first to be used, are guaranteed to find the GMEC while stochastic methods are not guaranteed to find it. Other methods, e.g., the A* search algorithm, were optimized to run in parallel taking advantage of the graphic processing unit (GPU) processor infrastructure (Chapter 13). Complementarily, the CPD effort should consider the solvating milieu, e.g., via a geometric potential (Chapter 5). In addition, the residue-level core building block focus of CPD should be analyzed and predicted in respect to phylogenetic, structural, and energetic properties. These should be treated according to the immediate and possibly changing microenvironment, e.g., as in protein–-protein complexes (Chapter 6). The GMEC considers a single minimum conformation and can be applied for the redesign of a given scaffold (Chapter 10), for requested functional motifs (Chapter 11) or for emphasizing specific types of available data, e.g., evolutionary information (Chapter 12). Yet, proteins within their native physiological surrounding are dynamic ensembles intrinsically requiring conformational dynamics. As such, it is important to *a priori* design the protein as a multistate entity (Chapter 7), a characteristic that can be introduced via integrating to the design process methods that analyze dynamics such as molecular dynamics (Chapter 8) or normal mode analysis (Chapter 9).

The computational design scheme can be tailored to specific types of proteins or domains, which in turn should be assessed as to their resemblance to the requested domain or specific designated characteristic. Examples include protein–protein interaction interfaces (Chapter 14), drug-resistance mutations (Chapter 15), symmetric proteins of identical sequence repeats (Chapter 16), self-assemblies exploiting synthetic amino acids (-Chapter 17), oligomerized conformations of the defensins (Chapter 18), ligand-binding proteins (Chapter 19), proteins with reduced immunogenicity (Chapter 20), antibodies (Chapter 21), membrane curvature-sensing peptides (Chapter 22), and allosteric drug-binding sites within proteins (Chapter 23). Taken together, these application focus areas

present the breadth of the CPD field along with the intrinsic achievements and challenges upon examining the "devil" in the details of key examples.

The general field of protein design, let alone the computational aspect of it, is expected to present an exponential increase in quality and quantity alike. Such change is fostered by the need to expand protein space for understanding biology, for applying biotechnology, and for expanding pharmaceuticals from the common small molecules to biologics – specific and side-effect-free proteins. Importantly, while scientific research of proteins is often focused towards pharmaceutical applications, CPD presents the possibility to expand the use of proteins in food-tech and white biotechnology, namely, the use of proteins for industrial applications. In addition, the field is nurtured by the exponential increase in raw sequence and structure data, and the increase in cost-effect computational hardware in general and hardware tailored to protein application, in particular. Not less important is the careful feedback loop of quantitative parameterization sequence and fold space followed by software design that will efficiently test our parameterization and produce novel protein design, which in turn can be materialized and characterized experimentally.

Karmiel, Israel ***Ilan Samish***

Contents

Contributors

Rudresh Acharya • *School of Biological Sciences, National Institute of Science Education and Research, Odisha, India; Homi Bhabha National Institute, Training School Complex, Anushakti Nagar, Mumbai, India*

Carlos Aleman • *Departament d'Enginyeria Química, E. T. S. d'Enginyeria Industrial de Barcelona, Universitat Politècnica de Catalunya, Barcelona, Spain; Center for Research in Nano-Engineering, Universitat Politècnica de Catalunya, Barcelona, Spain*

David Allouche • *Unité de Mathématiques et Informatique Appliquées de Toulouse, Castanet Tolosan, France*

Amy C. Anderson • *Department of Pharmaceutical Sciences, University of Connecticut, Storrs, CT, USA*

Isabelle André • *INSA, UPS, INP, Université de Toulouse, Toulouse, France; Laboratoire d'Ingénierie des Systèmes Biologiques et des Procédés, INRA, UMR792, Toulouse, France; CNRS, UMR5504, Toulouse, France*

Naureen Aslam • *Department of Computational Medicine and Bioinformatics, University of Michigan, Ann Arbor, MI, USA*

Chris Bailey-Kellogg • *Department of Computer Science, Dartmouth College, Dartmouth, USA*

Sophie Barbe • *INSA, UPS, INP, Université de Toulouse, Toulouse, France; Laboratoire d'Ingénierie des Systèmes Biologiques et des Procédés, INRA, UMR792, Toulouse, France; CNRS, UMR5504, Toulouse, France*

Arménio J. M. Barbosa • *UCIBIO, REQUIMTE, Departamento de Química, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Caparica, Portugal*

Ricardo J. F. Branco • *UCIBIO, REQUIMTE, Departamento de Química, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Caparica, Portugal*

Jeffrey R. Brender • *Department of Computational Medicine and Bioinformatics, University of Michigan, Ann Arbor, MI, USA*

Henrique F. Carvalho • *UCIBIO, REQUIMTE, Departamento de Química, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Caparica, Portugal; Instituto de Tecnologia Química e Biológica António Xavier, Universidade Nova de Lisboa, Av. da República, Oeiras, Portugal*

Matthieu Chartier • *Department of Biochemistry, Faculty of Medicine and Health Sciences, University of Sherbrooke, Sherbrooke, QC, Canada*

Quan Chen • *School of Life Sciences, Hefei National Laboratory for Physical Sciences at the Microscales, and Collaborative Innovation Center of Chemistry for Life Sciences, University of Science and Technology of China, Hefei, Anhui, China*

Roberto A. Chica • *Department of Chemistry and Biomolecular Sciences, University of Ottawa, Ottawa, ON, Canada*

Yoonjoo Choi • *Department of Computer Science, Dartmouth College, Dartmouth, USA*

James A. Davey • *Department of Chemistry and Biomolecular Sciences, University of Ottawa, Ottawa, ON, Canada*

Charlotte M Deane • *Department of Statistics, University of Oxford, Oxford, UK*

Nikolay V. Dokholyan • *Department of Biochemistry and Biophysics, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA*
Bruce R. Donald • *Department of Computer Science, Duke University, Durham, NC, USA; Department of Biochemistry, Duke University, Durham, NC, USA; Department of Chemistry, Duke University, Durham, NC, USA*
James Dunbar • *Department of Statistics, University of Oxford, Oxford, UK*
Juan Fernández-Recio • *Joint BSC-CRG-IRB Research Program in Computational Biology, Barcelona Supercomputing Center, Barcelona, Spain*
Vincent Frappier • *Department of Biochemistry, Massachusetts Institute of Technology, Cambridge, MA; Faculty of Medicine and Health Sciences, University of Sherbrooke, Sherbrooke, QC, Canada*
Pablo Gainza • *Department of Computer Science, Duke University, Durham, NC, USA*
Wenxun Gan • *Research Center of Basic Medical Sciences and Cancer Institute and Hospital, Tianjin Medical University, Tianjin, China*
Ivelin Georgiev • *Department of Computer Science, Duke University, Durham, NC, USA; Vaccine Research Center, National Institute of Allergy and Infectious Diseases, Bethesda, MD, USA*
Fabian Glaser • *Bioinformatics Knowledge Unit, The Lorry I. Lokey Interdisciplinary Center for Life Sciences and Engineering, Technion, Israel*
Karl E. Griswold • *Thayer School of Engineering, Dartmouth, USA*
Nurit Haspel • *Department of Computer Science, The University of Massachusetts Boston, Boston, MA, USA*
Wenkang Huang • *Department of Pathophysiology, Key Laboratory of Cell Differentiation and Apoptosis of Chinese Ministry of Education, Shanghai JiaoTong University School of Medicine (SJTU-SM), Shanghai, China*
Olga Iranzo • *Aix Marseille Université, Centrale Marseille, CNRS, iSm2 UMR 7313, Marseille, France*
Armando J. De Jesus • *Department of Chemistry and Biochemistry, University of Colorado, Boulder, CO, USA; The BioFrontiers Institute, University of Colorado, Boulder, CO, USA*
Sagar D. Khare • *Department of Chemistry and Chemical Biology, Center for Integrative Proteomics Research, Rutgers, The State University of New Jersey, Piscataway, NJ, USA*
Daisuke Kihara • *Department of Biological Sciences, Purdue University, West Lafayette, IN, USA; Department of Computer Science, Purdue University, West Lafayette, IN, USA*
Patrice Koehl • *Department of Computer Science and Genome Center, University of California, Davis, CA, USA*
Konrad Krawczyk • *Department of Statistics, University of Oxford, Oxford, UK*
David La • *Department of Biochemistry, University of Washington, Seattle, WA, USA*
Jie Li • *Computational and Systems Biology Group, Genome Institute of Singapore, Agency for Science, Technology and Research, Singapore, Singapore*
Haiyan Liu • *School of Life Sciences, Hefei National Laboratory for Physical Sciences at the Microscales, University of Science and Technology of China, Hefei, Anhui, China; Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei, Anhui, China*
Anna Lowegard • *Program in Computational Biology and Bioinformatics, Duke University, Durham, NC, USA*
Buyong Ma • *Basic Science Program, Leidos Biomedical Research, Inc, Frederick, MD, USA; Cancer and Inflammation Program, National Cancer Institute, Frederick, MD, USA*
David D. Mowrey • *Department of Biochemistry and Biophysics, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA*

Rafael Najmanovich • *Department of Pharmacology and Physiology, Faculty of Medicine, University of Montreal, Montreal, QC, Canada*

Ruth Nussinov • *Department of Human Genetics, Sackler School of Medicine, Sackler Inst. of Molecular Medicine and Molecular Medicine, Tel Aviv University, Tel Aviv, Israel; Basic Science Program, Cancer and Inflammation Program, Leidos Biomedical Research, Inc., National Cancer Institute, Frederick, MD, USA*

Adegoke Ojewole • *Program in Computational Biology and Bioinformatics, Duke University, Durham, NC, USA*

Chiara Pallara • *Joint BSC-CRG-IRB Research Program in Computational Biology, Barcelona Supercomputing Center, Barcelona, Spain*

Stephanie M. Reeve • *Department of Pharmaceutical Sciences, University of Connecticut, Storrs, CT, USA*

Miguel Romero-Durana • *Joint BSC-CRG-IRB Research Program in Computational Biology, Barcelona Supercomputing Center, Barcelona, Spain*

Ana C. A. Roque • *UCIBIO, REQUIMTE, Departamento de Química, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Caparica, Portugal*

Ilan Samish • *Department of Plants and Environmental Sciences, Weizmann Institute of Science, Rehovot, Israel; Department of Biotechnology Engineering, Braude Academic College of Engineering, Karmiel, Israel; Amai Proteins Ltd., Ashdod, Israel; Dept of Biotechnology Engineering, Braude Academic College of Engineering, Karmiel, Israel; Amai Proteins Ltd., Ashdod, Israel*

Thomas Schiex • *Unité de Mathématiques et Informatique Appliquées de Toulouse, UR 875, INRA, Castanet Tolosan, France*

Dina Schneidman • *Department of Bioengineering and Therapeutic Sciences, University of California, San Francisco, CA, USA*

David Shultis • *Department of Computational Medicine and Bioinformatics, University of Michigan, Ann Arbor, MI, USA*

David Simoncini • *Structural Bioinformatics Team, Division of Structural and Synthetic Biology, Center for Life Science Technologies, RIKEN, Yokohama, Kanagawa, Japan; MIAT, UR-875, INRA, Castanet Tolosan, France*

Jeremy R. H. Tame • *Drug Design Laboratory, Graduate School of Medical Life Science, Yokohama City University, Yokohama, Kanagawa, Japan*

Christine E. Tinberg • *Department of Biochemistry, University of Washington, Seattle, WA, USA*

Seydou Traoré • *INSA, UPS, INP, Université de Toulouse, Toulouse, France; INRA, UMR792, Université de Toulouse, Toulouse, France; Laboratoire d'Ingénierie des Systèmes Biologiques et des Procédés, CNRS, UMR5504, Toulouse, France*

Barsa Tripathy • *School of Biological Sciences, National Institute of Science Education and Research, Odisha, India; Homi Bhabha National Institute, Training School Complex, Anushakti Nagar, Mumbai, India*

Deeptak Verma • *Department of Computer Science, Dartmouth College, Dartmouth, USA*

Arnout R. D. Voet • *Structural Bioinformatics Team, Division of Structural and Synthetic Biology, Center for Life Science Technologies, RIKEN, Yokohama, Kanagawa, Japan*

Qing Wei • *Department of Computer Science, Purdue University, West Lafayette, IN, USA*

Peng Xiong • *School of Life Sciences, Hefei National Laboratory for Physical Sciences at the Microscales, and Collaborative Innovation Center of Chemistry for Life Sciences, University of Science and Technology of China, Hefei, Anhui, China*

HANG YIN • *Department of Chemistry and Biochemistry, The BioFrontiers Institute, University of Colorado, Boulder, CO, USA*
DAVID ZANUY • *Departament d'Enginyeria Química, E. T. S. d'Enginyeria Industrial de Barcelona, Universitat Politècnica de Catalunya, Barcelona, Spain*
JIANYANG ZENG • *Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, P. R. China*
JIAN ZHANG • *Department of Pathophysiology, Key Laboratory of Cell Differentiation and Apoptosis of Chinese Ministry of Education, School of Medicine (SJTU-SM), Shanghai JiaoTong University, Shanghai, P. R. China*
KAM Y. J. ZHANG • *Structural Bioinformatics Team, Division of Structural and Synthetic Biology, Center for Life Science Technologies, RIKEN, Yokohama, Kanagawa, Japan*
NING ZHANG • *Research Center of Basic Medical Sciences and Cancer Institute and Hospital, Tianjin Medical University, Tianjin, China*
YANG ZHANG • *Department of Computational Medicine and Bioinformatics, University of Michigan, Ann Arbor, MI, USA; Department of Biological Chemistry, University of Michigan, Ann Arbor, MI, USA*
JIE ZHENG • *Department of Chemical and Biomolecular Engineering, The University of Akron, Akron, OH, USA*
YICHAO ZHOU • *Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, P. R. China*
CHENG ZHU • *Department of Biochemistry and Biophysics, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA*

Part I

Computational Protein Design

Chapter 1

The Framework of Computational Protein Design

Ilan Samish

Abstract

Computational protein design (CPD) has established itself as a leading field in basic and applied science with a strong coupling between the two. Proteins are computationally designed from the level of amino acids to the level of a functional protein complex. Design targets range from increased thermo- (or other) stability to specific requested reactions such as protein–protein binding, enzymatic reactions, or nanotechnology applications. The design scheme may encompass small regions of the proteins or the entire protein. In either case, the design may aim at the side-chains or at the full backbone conformation. Herein, the main framework for the process is outlined highlighting key elements in the CPD iterative cycle. These include the very definition of CPD, the diverse goals of CPD, components of the CPD protocol, methods for searching sequence and structure space, scoring functions, and augmenting the CPD with other optimization tools. Taken together, this chapter aims to introduce the framework of CPD.

Key words Computational protein design, Protein structure prediction, Structural bioinformatics, Computational biophysics, Synthetic biology, Negative design

> *"Most people make the mistake of thinking design is what it looks like. People think it's this veneer—that the designers are handed this box and told, 'Make it look good!' That's not what we think design is. It's not just what it looks like and feels like. Design is how it works."*
> Steve Jobs, Apple's C.E.O in an interview to the New-York Times. Nov. 30th 2003, The Guts of a New Machine
> http://www.nytimes.com/2003/11/30/magazine/the-guts-of-a-new-machine.html

1 Introduction

The aim of this chapter is to describe the essence of computational protein design (CPD), which, as Steve Jobs explained (*see* exert above) is "*how it works*". Proteins, nature's main structural building blocks, workers, and nano-machines, were designed over 3 billion years of evolution; optimizing the biological need for stable yet dynamic function under diverse and changing ecological niches. Evolution follows two approaches—the classical divergent evolution includes a slow change in the sequence (evolutionary drift) followed by survival

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_1,

of the fittest proteins from the evolving genepool. The fittest are not necessarily the strongest or the most stable as fitness requires being sufficiently stable to accommodate function, often under more than one condition; along with the ability to degrade the protein when it is not needed or is damaged. In parallel, there are numerous examples of convergent evolution where different evolutionary pathways lead to functionally and structurally similar active sites. CPD follows both of these evolutionary approaches thus narrowing the overall "survival of the fittest" criterion from the organism to the protein level. Furthermore, different and complementary approaches are often applied to a requested design with the methodology following the available toolbox and scientific approach of the computational designer.

The field of CPD has been reviewed in the frame of the general methodology [1–8] as well as specific methodological aspects such as library-scale CPD [9], multistate approaches and backbone flexibility [10–13], electrostatics [14], fragment databases [15], and energy landscapes [16, 17]. Specific CPD applications and protein family targets have been reviewed such as protein therapeutics [18], ligand binding and enzyme catalysis [19–22], binding specificity [23, 24], membrane proteins [25, 26], metalloproteins [27], collagens [28], conformational switches [29], and protein–protein interactions [30]. Numerous other aspects are presented as part of this very book which is the first book with this title. Here I aim to present the general framework of CPD.

2 CPD and In Vitro (Directed) Evolution

In many ways, CPD is the natural extension of noncomputational protein design and in vitro evolution which have evolved over the last half century [31]. Moreover, as complementary approaches, the methods should not be viewed as "either/or" but rather as different ways to reach a common goal with the ability to intertwine several methods. For example, in several cases CPD partially succeeded and was optimized by directed evolution which was re-termed in this context as affinity maturation [32].

Rational protein design commonly relies on the biochemical and biophysical know-how of the scientist who predicts one or more specific mutation sites as the loci potentially leading to the requested design. Saturated mutagenesis in which a specific locus is mutated to several or all amino acids is often applied when the target site is identified but the local-structure function relationships of all residues is unknown, e.g. as applied for resolving the photosystem II mechanism of acclimation to the ambient temperature [33]. In other cases a full domain or a full protein is the design target. In vitro evolution circumvents the challenging need to assess each mutation discretely by applying an assay that can test many genetic alterations at once with the *post-factum* analysis of the gene

or amino acid sequence leading to the one or more sequences that provide satisfactory results.

In vitro evolution, also termed "directed evolution," consists of consecutive rounds of error-prone polymerase chain reaction (PCR) and DNA shuffling [34, 35]. It makes use of the two basic principles of Darwinian evolution including an (accelerated) evolutionary drift that diversifies the genepool and a focus on "survival of the fittest" selection assays. Rational protein design and directed evolution as well as the many methods which close the continuous gap between these methodologies, may benefit from computational methods powered by the relatively cheap in silico power. Moreover, these methods are constrained by the availability of mass screening, which is not accessible for many design targets. This chapter aims to draw a common thread to the different pathways of CPD with an emphasis on the challenges along the different milestones of the process. The next chapter, which should be considered as a natural follow-up to this one, is focused on specific solutions that were applied to encounter these challenges, thus providing a case-study approach to the achievements and challenges of the field.

3 Maturity of the CPD Field and the Lack of an Objective Assessment

The stage of the CPD field is still premature and evolving, e.g. this is the first book with this title. The proof of the CPD success is simply the growing number of available specific functional designed proteins. In the related field of protein structure prediction John Moult sparked an important revolution by establishing the Critical Assessment of Structure Prediction (CASP) competition two decades ago [36]. In this competition there is an important separation of jurisdiction between the software developers, users and the judges; thus obtaining an objective critical assessment of the state of different structure prediction subclasses and the strengths and weaknesses of each method. Unfortunately, there is no such competition in CPD resulting in the lack of fully objective comparisons of the methods involved. Accordingly, this chapter aims to present common themes found in different CPD methods in a qualitative rather than quantitative manner.

While CPD is still evolving, success stories of computationally designed proteins highlight the current success and future potential of CPD. Actually, the very table of contents of this book (especially part III of the book) provides a glimpse as to the scope of successful CPD attempts. These encompass specific protein families such as membrane curvature-sensing peptides, ligand-binding proteins, or antibodies via designed structural motifs, e.g. symmetric proteins or self-assemblies, and to the design of dynamic characteristics, e.g. allosteric sites.

The success of specific CPD attempts and the lack of overall uniformity in methodology is not necessarily a disadvantage. The

plethora of computational available tools and methods highlight the complexity of the field and the need for designated solutions for specific subclasses; whether these are structural (e.g. specific folds) or functional (e.g. stabilization). In essence, any characteristic parametrization, whether statistical knowledge-based or energy-based, can be inversely applied for CPD.

4 Definition of CPD

CPD can be defined in more than one way. This very statement is at the heart of CPD, which defines a field with fuzzy borders that are intimately connected to numerous other fields. Indeed, computational protein designs are often found in publications that do not use this explicit term. When searching online databases for research papers and reviews that mention this precise term till 2014, the Web of Science and PubMed databases show 260 and 170 publications, respectively (Fig. 1). As a multidisciplinary field, some of the CPD chemical and computational publications are not indexed in PubMed thus resulting in a higher number of publications when searching the Web of Science database. In this database, the trend

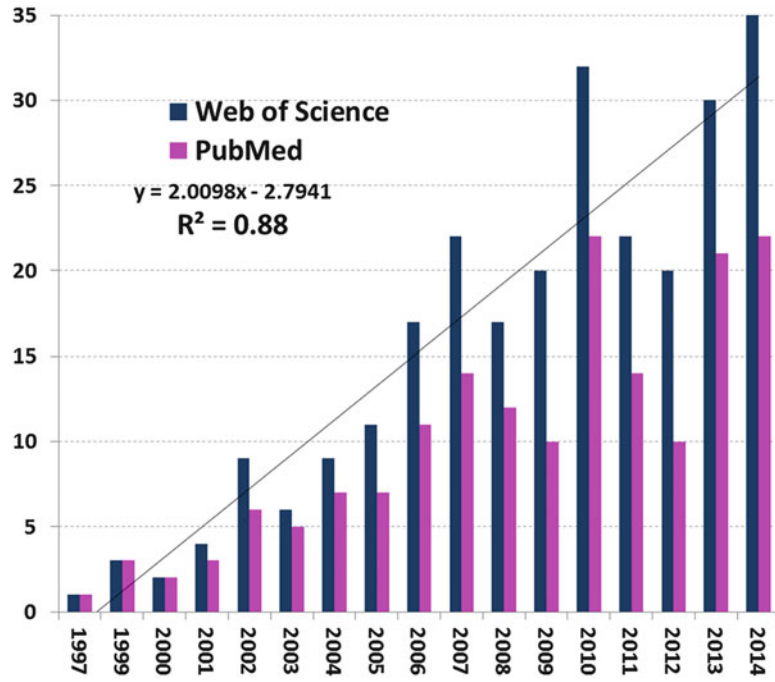


Fig. 1 Publications with the term "Computational Protein Design" as datamined from the Web of Science (*blue*) and PubMed databases (*pink*). The graph is meant to provide a rough estimation of the growth and changes in the field. It includes only research papers and reviews and does not include CPD publications which do not mention the explicit searched term

line exhibits a clear rise in the number of publications, yet with quite a bit of variation between years, as typical for a yet young and evolving field. Since 2009 every year there are over 20 publications with over 30 publications in the last couple of years (2013–2014).

With this background in mind, CPD is defined as *the computer-aided rational (or semi-rational) design of a protein (or part thereof) to fold to a requested structure or to facilitate a requested (possibly novel) function or biophysical property* (e.g. *stability*).

This definition encompasses a complex and nonlinear protocol (*see* Fig. 2) which touches upon several multidimensional aspects of the CPD field:

1. *Resolution of the CPD output*—The resolution of the CPD output is not part of the definition and depends on the

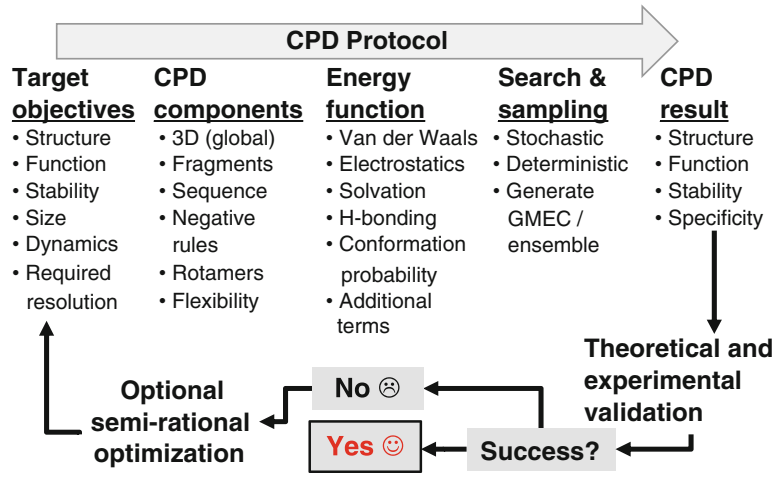


Fig. 2 A schematic description of the CPD protocol. First, careful characterization of the target objectives is conducted in the level of structure, function, stability, size, dynamics, and required resolution for the CPD result. Each CPD case-study should have different weights on each of these aspects. Second, a decision is taken as to which components are part of the CPD protocol—ranging from quantitative description of global features (such as coiled-coils Crick parameters) via usage of fragment and/or rotamer libraries to specific sequence features and negative design rules. Third, an energy function is fit to the previous steps. The energy function most commonly includes bonded- and nonbonded interactions along with rotamer or other conformation probability and additional terms which are case-study specific. Fourth, search and sampling methods fit for the CPD required framework are chosen. These can be stochastic or deterministic, generating a single design or an ensemble of designs. Fifth, the design output coordinates and sequence are produced and assessed for structural features such as stability and for functional features such as specificity. Next, the design is validated theoretically by comparing it to known structures and quantitative available parameterization followed by experimental production and characterization. If the design goal is not achieved, the design can benefit from other semi-rational optimization methods such as in vitro evolution.

requested target. In some cases a sub-atomic design scheme is required, e.g. a combination of quantum-mechanics calculations, while in others the structure resolution per se is not part of the goal. Moreover, different parts of the design target may be designed in different resolutions.

2. *The target size of the CPD*—CPD may target anything from a small region to a full protein. How small of a region is still regarded in the frame of CPD is an open question as a single residue site-directed mutagenesis is commonly regarded as protein engineering rather than design. Yet, design of a binding site or designing a protein with altered specificity may include very few amino acids.
3. *The target identity of CPD*—CPD may target a structure, a function, or a biophysical property. Each of these end-points dictate a different approach to the design scheme. The holy grail of protein design, whether computational or not, is the "inverse protein folding problem" defined in 1992 by Yue and Dill [37]. Therein, the goal is to design a protein sequence that will fold into a requested and defined structure. Nevertheless, CPD may target aspects which are not a specific structure but rather a specific characteristic thereof. Once there is a quantitative parameterisation, whether an amino acid scale for e.g. protein–protein interfaces or a defined deviation between mesophiles and thermophiles, the targeted trait can be designed with the aid of computation.
4. *The level of "rationality" of CPD*—CPD ranges from a sub-atomic resolution target structure designed via a single sequence to varying level of random mutations—from simultaneous saturation mutagenesis of designed residues and till random mutagenesis or even DNA shuffling conducted in the frame of directed evolution. Furthermore, the rational design can be coupled to a less rational design in a stepwise fashion with a first version of the requested protein designed rationally designed and subsequent steps designed with the aid of high-throughput screening. In the case of protein–protein interaction design, this process is termed "affinity maturation [38]."

An interesting example demonstrating all of the points is one of the first attempts to design an artificial enzyme in which Kemp elimination catalysts were designed [39]. The first step of this design protocol included quantum mechanics level transition-state calculations to create an idealized active site of the requested catalytic mechanism. The calculations suggested how to position protein functional groups so as to maximize transition state stabilization. The high-resolution rational approach was only for the catalytic residues with the potential list of template protein scaffolds

including about 100 proteins. These were narrowed down to 59 candidate enzymes using modeling and practical considerations. All the designs were expressed and assayed as to their enzymatic requested activity. The leading candidate was further optimized using in vitro evolution. Structurally, only residues involved in the catalytic mechanism were designed in high-resolution while the final assay was a functional rather than a structural assay. Hence, in this one example, the resolution and rationality of the design protocol exhibited a large variation between the key catalytic residues and other parts of the enzyme.

5 Objectives of Computationally Designed Proteins

It is important to define the objectives underlying the development and use of the field, namely, what are the computationally designed proteins expected to achieve? Such goals include basic and applied goals alike and can be divided by the type of basic understanding of the protein and the type of application pursued:

1. *Protein folding or the inverse folding problem*—the entropic hydrophobic effect [40] underlying protein folding is long known, yet the details of protein folding are still not fully elucidated. The inverse protein folding problem, namely, the problem of finding which amino acid sequences fold into a known three-dimensional (3D) structure [37, 41, 42] is in essence the holy grail of protein design.
2. *Specificity*—The design of specific interactions (protein–protein or protein–ligand) is related to the application of negative design rules (described below). Here, one can a priori focus the design efforts on regions that determine specificity, or, alternatively, add similar templates (decoys or related molecules) to examine the target affinity in respect to a background of unwanted interactions.
3. *Stability and extremophilicity*—Our body invests energy in maintaining a mesophilic mild environment for proteins including narrow range of temperature, salt concentrations, pH etc. Yet, designed proteins are often expected to function in hostile environments whether these are fermenters in the biotechnology industry where protein yield is a goal or whether these are synthetic biology applications e.g. bio-detergents. Concomitantly, the CPD approach provides a unique method to study the very determinants underlying the requested extremophile trait.
4. *Synthetic biology*—Natural proteins were optimized according to the need of organisms and the constraints of the evolutionary process, e.g. not enabling large leaps at a time and not focusing on traits that don't affect organism survivability. In

vitro evolution attempts to harness turbo-mode rules of evolution with new survival assays to produce proteins of interest. Nevertheless, the process is still constrained by the aforementioned components. Taken together, CPD provides an important toolbox for synthetic biology applications [43, 44].

5. *Negative design rules*—While the natural intuitive logic focuses on the direct objective, often the unwanted objective is not less important. CPD offers a focused path to study negative design rules which are often overlooked due to methodological challenges in studying them. In other words, while the natural focus of biology is answering the question "how do things work?" this is often the easy question. The question that is not less easy is: "how do things not work in the wrong direction?" The two questions are not two sides of the same coin but rather two complementary fields that only when combined answer the question of "how do things work in a living system?" A good example of combining positive- and negative-design rules in a related field encompasses the success of drugs as given by the therapeutic index (TI). The index combines the positive effect of manipulating the requested target with the negative side-effects, generally expressed by the lethal-dose (LD) which is usually due to lack of specificity and/or is due to toxicity of the drug or metabolites or degradation products thereof. (*see* **Note 1**)

In summary, while evolution (in vivo or in vitro) examines the overall fitness of the organism, CPD enables a focused design with positive and negative rules alike. These rules can be statistical knowledge-based rules where the underlying physics is not fully understood or may not be fully parameterized, or, alternatively, biophysical rules underlying specific enthalphic or entropic contributions, or lack of, to the requested design.

6 Structural Levels of CPD: Design Target and Design Building Block

The structural levels of CPD include two opposing aspects—the structural level of the target of CPD and the building block to achieve the CPD.

6.1 Structural Levels of CPD: Design Target

The CPD procedure can be applied in many different structural levels. This is not only a description of the final goal but also strongly affects the CPD procedure as different structural levels dictate different search and sampling strategies as well as different scoring functions.

1. *De novo CPD*—The most classical CPD procedure is the so-called de novo design where a totally new fold and/or function are pursued, e.g. as is the case for the betadoublet beta-

sandwich design of Richardson [45], the TOP7 design of Kuhlman and Baker [46], the helix-bundle designs of DeGrado and coworkers (e.g. ref. 47), the transmembrane Zn^{2+} transporter of DeGrado [48], or the recent enzyme designs of Baker (e.g. ref. 49).

2. *Core stabilization*—The driving force for folding of soluble proteins is the entropic hydrophobic effect in which the collapse of the hydrophobic protein core maintains the disorder of the aqueous solvent around the solvent accessible hydrophilic amino acids of the protein. However, the hydrophobic effect results in a molten globule which is later optimized for enthalpic contributions of specific interactions and packing. As this process is often not optimized for stability, the design of better protein cores is a long-standing approach within CPD [50–52]. In general, most CPD attempts thus far included a component of core stabilization. Other approaches to stabilization include targeting the most unstable parts of the protein, e.g. loops (*see* **Note 2**).
3. *Solubilization of protein–solvent interface*—One of the most classic examples of genetic diseases, sickle-cell anemia, includes a hydrophobic patch on the surface of the hemoglobin β-subunit following a single Glu → Val mutation. As such, maintaining the solubility of the protein may assist in avoiding aggregation. Likewise, membrane proteins were solubilized to allow for the study of the membrane protein within an aqueous milieu as well as in order to study the basic features of membrane proteins e.g. references [53, 54].
4. *Symmetry*—The complexity of the CPD process can be largely trimmed by adding symmetry to the structural design. This can be done for symmetric proteins such as beta-propeller proteins [55], for coiled coils [56], or crystallographic symmetry [57].
5. *Binding site*—The binding site is literally the heart of the protein and usually requires special care which is different from the general approaches to other CPD regions. These range from quantum-mechanics optimization to grafting an existing site to a de novo designed template. For example, a binding site CPD includes many different case-studies such as changing the bound metal, e.g. as done for ferritin [58], de novo designed metal-binding [59] or nonbiological cofactor-binding [60] proteins, and enzymes [19–22].
6. *Protein–protein interactions* (*PPI*)—While a binding site is often specifically designed for a non-amino acid moiety, protein–protein interaction CPD include the stable or transient interaction between spatial patches of amino acids that are on the surface-accessible part of the protein [30]. Numerous case-studies of PPI CPD were applied with altered specificity [61,

62] and affinity [63]. Likewise, new PPI were designed for binding a conserved surface of the influenza hemagglutinin [32]. In this frame, the large field of antibody CPD (e.g. [64]) is essentially the design of new PPI incorporating unique features of the antibody such as the hyper-variable loops.

7. *Dynamics*—Proteins are often regarded as XYZ coordinates of frozen structures with a global minimum energry conformation (GMEC) structure represented within PDB files. However, proteins are four-dimensional machines (space and time dimensions) with intrinsic local flexibility and global dynamics. A ligand-controlled conformational switch [65], an minimal 75-residue allosterically-regulated Kemp eliminase catalyst [66], or a Zn^{2+} transporter [48] provide an example to CPD focusing on such functional dynamics which must be a major focus of any CPD involving dynamic function.
8. *Membrane proteins and other "unique" protein groups*—As presented in this book, many protein families have designated CPD schemes which harness family-specific parameterization. Perhaps the most important such group is membrane proteins [25, 26], which constitute over a quarter of all genes, most communication between cells and organelles and as such also most drug targets. Thus far, this is the youngest and least understood field in structural biology. Successful membrane protein CPD includes a specific transmembrane integrin-binding helix [67] and the Zn^{2+} transporter [48].

6.2 Structural Levels of CPD: Design Building Block

CPD requires designated software or the integration of existing software in a manner tailored to the requested goal. Many chapters in this book provide detailed examples to such tools. In this frame, CPD can be applied in several structural levels—from optimization of an active site by quantum mechanics to global geometric features. Hierarchically, from small to large, the main structural features include:

1. *Rotamers and conformers*—The basic building blocks of amino acid side-chains and their role in structural bioinformatics are reviewed elsewhere [68]. Briefly, the Dunbrack rotamer library [69–71], representing the main side-chain conformations in a backbone-dependent manner, became the standard lookup tables scanned within the CPD procedure. As each side-chain can accept only a discrete number of conformations represented in the rotamer library, these libraries are at the heart of CPD. Alternatively, much larger conformer libraries, e.g. reference [72], can account for side-chain conformations which are not at a local or global energy minimum. Unlike the average side-chain conformation of rotamer libraries, here each conformation depicts a specific side-chain conformation from a

high-resolution structure. Taken together, these side-chain structural libraries are the three-dimensional natural extension of sequence space to describe the possible structures at each position.

2. *Flexible backbone*—Most often, while side-chain conformations are thoroughly scanned via rotamer- or conformer-libraries, the backbone conformation is copied from an existing structure. Consequently, it is important to sample alternative local conformations via multistate approaches or the artificial introduction of backbone flexibility [10–13]. Such flexibility enables not only the introduction of larger side-chains at each template position, but also enables to fit the new structure to the newly introduced local geometrical constraints.
3. *Fragments*—As the scientific committee still doesn't know to address the physics-based complexity of protein GMEC structure design sufficiently well, it is beneficial to reassemble known high-resolution structural fragments in a knowledge-based approach. The most famous such example is the Rosetta software of the Baker lab with RosettaDesign [73] tailored for CPD. Here, a nine-amino acid fragment library is used for the initial construction of the designed region. Next, rotamer library optimization and an energy function including local and global features are applied. These include careful knowledge-based pseudo-energetics of hydrogen bonds, solvation energy, and the usual force-field components such as steric clash and electrostatics.
4. *Geometrical global features*—Last but not least, the design of domains and full proteins often applies equations addressing global features. Perhaps the most known of these are the family of coiled coils [56] comprising 10 % of proteins. Here, equations correlating sequence and helical bundle geometry are useful for the de novo design of the protein fold [57, 74]. Other knowledge-based potentials include the Ez potential for assessing the cross-membrane pseudo-energetics [75], which was applied to design a transporter [48], or even equations assessing solvent accessibility.

7 Search and Sampling Procedures

The topic of search and sampling [68] in CPD is the beating heart of the process. In analogy, all the above description composes the ingredients of this blood but without proper circulation an insufficient number of components will be included in CPD; predisposing the process to failure. Complimentary, efficient search and sampling methods allow for higher resolution designs as additional layers of information can be included in the design cycle. The topic requires a book devoted to it and is introduced elsewhere [68] with specific

focus areas described as focus areas in this book. Consequently, here only a very brief description of the topic and related jargon is presented. Search and sampling methods are grossly classified as stochastic and deterministic. Deterministic methods have access to the complete data and if they converge they are bound to find the GMEC. These include dead-end elimination (DEE) which is often combined with the A* search algorithm (DEE/A*), self-consistent field method (SCMF), belief propagation, molecular dynamics (MD), branching methods, graph decomposition, cost function network (CFN) algorithms, Markov random field solvers (MRF) and linear programming.

In contrast to deterministic methods, stochastic search methods have a random component and may give a different answer each run pending on the specific number produced by the random number generator which is part of the algorithm. The most known stochastic method is Monte Carlo (MC) where different additional measures are applied to drive convergence and decrease the number of random steps. These include biased MC, MC-quench or combining sampling power of MC with the speed of methods such as SCMF. The iterative stochastic elimination (ISE) aims at producing a manageable high-scoring ensemble rather than a single GMEC, such that the ensemble can be later searched with other methods [7]. Alternatively, temperature is introduced to control the distance between steps, as done in simulated annealing (SA) or the replica exchange method (REM). Often, to avoid convergence in the wrong local minima, occasional jumps (jump walking or j-walking) are introduced. Biological methods such as genetic algorithms (GA) aim to imitate the evolutionary process by improving the population of results. Last but not least, often hierarchical methods are applied for the different parts of the CPD procedure, each fit for a different search space and resolution.

8 CPD as a Feedback Loop: Negative Design, Quality Assessment, and Experimental Validation

CPD is not a standalone procedure for optimizing a target structure or function. Not less important is the unwanted result. Indeed, many successful CPD case-studies hardwired the so-called negative-design into the CPD protocol [49, 76–83]. Negative design may include unwanted conformation or binding partner or even an unwanted structural characteristic. Next, the theoretical model of CPD should be assessed with every possible type of quality assessment (QA) tool, whether general for all proteins or specific for the target protein family. Last, and most important, the suggested sequence should be assessed experimentally with the resulting experimental validation serving in a feedback iterative loop to improve the CPD. Moreover, often the CPD is successful

only to a certain limit or lacks the ability to score the best design within an ensemble. In such cases there are two options—either conducting experimental validation to many designs or adding an additional method to optimize the design, e.g. directed evolution.

9 Notes

1. When considering negative design rules, a good practical example of a positive vs. negative design metric is the common pharmaceutical therapeutic index (TI). TI is composed of the ratio between the lethal (or toxic) dose affecting 50 % of the population (LD_{50} or TD_{50}) and the effective dose for 50 % of the population (ED_{50}) i.e. $TI = TD_{50}/ED_{50}$. In essence, this is a ratio between the negative and positive effects. In molecular terms, the drug can bind with very high affinity to the target protein or, alternatively, the target protein may be well-designed for drug binding. Yet, the drug may also bind to other proteins, or, alternatively, the destruction of the protein's function may affect biochemical pathways that are beyond the pathway that was the focus of the drug design. Interestingly, while pharmaceutical companies focus energy on the study of such side-effects, this is still not the common scheme in CPD.
2. *Loop design* is the most difficult part of the protein target to design or to predict. Indeed, in protein structure prediction, the loops usually account for most of the RMSD between the model and the actual structure. To circumvent the challenge, some designs, e.g. TOP7 [46] confined the loop regions to the minimal length possible. The challenge includes several aspects: First, loops have no periodic structure-confining constraints as secondary structures exhibit. Second, loops are intrinsically flexible and, for longer loops, may even be intrinsically disordered. Third, loops are regularly part of soluble regions and do not have a confining domain they adhere to or a knowledge-based rule such as a hydrophobic core. Last, even short loops may be highly dependent on the precise geometry of the secondary structures from which they stem. Structure prediction software should give special attention to loops, though not all do it as a separate entity within the modeling scheme. As with other structure prediction tools, a consensus tool combining orthogonal methods may provide better results than the individual methods [84]. Designated tools focus efforts on the unique properties of this region. For example, SuperLooper [85] offers an online servers datamining a large (half-billion) loop structures derived from structural data. A known tool focusing on loop modeling is LoopBuilder [86] which tackles the challenge by an extensive sampling of backbone conformations, side-chain addition, the use of a statistical potential to

select a subset of these conformations, and, finally, an energy minimization and ranking with an all-atom force field.

References

1. Samish I, MacDermaid CM, Perez-Aguilar JM, Saven JG (2011) Theoretical and computational protein design. Annu Rev Phys Chem 62:129–149. doi:10.1146/annurev-physchem-032210-103509
2. Pantazes RJ, Grisewood MJ, Maranas CD (2011) Recent advances in computational protein design. Curr Opin Struct Biol 21 (4):467–472. doi:10.1016/j.sbi.2011.04.005
3. Saven JG (2011) Computational protein design: engineering molecular diversity, nonnatural enzymes, nonbiological cofactor complexes, and membrane proteins. Curr Opin Chem Biol 15(3):452–457. doi:10.1016/j.cbpa.2011.03.014
4. Tian P (2010) Computational protein design, from single domain soluble proteins to membrane proteins. Chem Soc Rev 39 (6):2071–2082. doi:10.1039/b810924a
5. Suarez M, Jaramillo A (2009) Challenges in the computational design of proteins. J R Soc Interface 6 Suppl 4:S477–S491. doi:10.1098/rsif.2008.0508.focus
6. Lippow SM, Tidor B (2007) Progress in computational protein design. Curr Opin Biotechnol 18(4):305–311. doi:10.1016/j.copbio.2007.04.009
7. Rosenberg M, Goldblum A (2006) Computational protein design: a novel path to future protein drugs. Curr Pharm Des 12 (31):3973–3997
8. Butterfoss GL, Kuhlman B (2006) Computer-based design of novel protein structures. Annu Rev Biophys Biomol Struct 35:49–65. doi:10.1146/annurev.biophys.35.040405.102046
9. Johnson LB, Huber TR, Snow CD (2014) Methods for library-scale computational protein design. Methods Mol Biol 1216:129–159. doi:10.1007/978-1-4939-1486-9_7
10. Davey JA, Chica RA (2012) Multistate approaches in computational protein design. Protein Sci 21(9):1241–1252. doi:10.1002/pro.2128
11. Lassila JK (2010) Conformational diversity and computational enzyme design. Curr Opin Chem Biol 14(5):676–682. doi:10.1016/j.cbpa.2010.08.010
12. Mandell DJ, Kortemme T (2009) Backbone flexibility in computational protein design. Curr Opin Biotechnol 20(4):420–428. doi:10.1016/j.copbio.2009.07.006
13. Ollikainen N, Smith CA, Fraser JS, Kortemme T (2013) Flexible backbone sampling methods to model and design protein alternative conformations. Methods Enzymol 523:61–85. doi:10.1016/B978-0-12-394292-0.00004-7
14. Vizcarra CL, Mayo SL (2005) Electrostatics in computational protein design. Curr Opin Chem Biol 9(6):622–626. doi:10.1016/j.cbpa.2005.10.014
15. Verschueren E, Vanhee P, van der Sloot AM, Serrano L, Rousseau F, Schymkowitz J (2011) Protein design with fragment databases. Curr Opin Struct Biol 21(4):452–459. doi:10.1016/j.sbi.2011.05.002
16. Saven JG (2001) Designing protein energy landscapes. Chem Rev 101(10):3113–3130
17. Kuhlman B, Baker D (2004) Exploring folding free energy landscapes using computational protein design. Curr Opin Struct Biol 14 (1):89–95. doi:10.1016/j.sbi.2004.01.002
18. Hwang I, Park S (2008) Computational design of protein therapeutics. Drug Discov Today Technol 5(2-3):e43–e48. doi:10.1016/j.ddtec.2008.11.004
19. Feldmeier K, Hocker B (2013) Computational protein design of ligand binding and catalysis. Curr Opin Chem Biol 17(6):929–933
20. Wijma HJ, Janssen DB (2013) Computational design gains momentum in enzyme catalysis engineering. FEBS J 280(13):2948–2960. doi:10.1111/febs.12324
21. Khare SD, Fleishman SJ (2013) Emerging themes in the computational design of novel enzymes and protein-protein interfaces. FEBS Lett 587(8):1147–1154. doi:10.1016/j.febslet.2012.12.009
22. Nanda V, Koder RL (2010) Designing artificial enzymes by intuition and computation. Nat Chem 2(1):15–24. doi:10.1038/nchem.473
23. Havranek JJ (2010) Specificity in computational protein design. J Biol Chem 285 (41):31095–31099. doi:10.1074/jbc.R110.157685
24. Sharabi O, Erijman A, Shifman JM (2013) Computational methods for controlling binding specificity. Methods Enzymol 523:41–59. doi:10.1016/B978-0-12-394292-0.00003-5
25. Senes A (2011) Computational design of membrane proteins. Curr Opin Struct Biol 21 (4):460–466. doi:10.1016/j.sbi.2011.06.004

26. Perez-Aguilar JM, Saven JG (2012) Computational design of membrane proteins. Structure 20(1):5–14. doi:10.1016/j.str.2011.12.003
27. Parmar AS, Pike D, Nanda V (2014) Computational design of metalloproteins. Methods Mol Biol 1216:233–249. doi:10.1007/978-1-4939-1486-9_12
28. Nanda V, Zahid S, Xu F, Levine D (2011) Computational design of intermolecular stability and specificity in protein self-assembly. Methods Enzymol 487:575–593. doi:10.1016/B978-0-12-381270-4.00020-2
29. Ambroggio XI, Kuhlman B (2006) Design of protein conformational switches. Curr Opin Struct Biol 16(4):525–530. doi:10.1016/j.sbi.2006.05.014
30. Kortemme T, Baker D (2004) Computational design of protein-protein interactions. Curr Opin Chem Biol 8(1):91–97. doi:10.1016/j.cbpa.2003.12.008
31. Joyce GF (2007) Forty years of in vitro evolution. Angew Chem Int Ed Engl 46 (34):6420–6436. doi:10.1002/anie.200701369
32. Fleishman SJ, Whitehead TA, Ekiert DC, Dreyfus C, Corn JE, Strauch EM, Wilson IA, Baker D (2011) Computational design of proteins targeting the conserved stem region of influenza hemagglutinin. Science 332 (6031):816–821. doi:10.1126/science.1202617
33. Shlyk-Kerner O, Samish I, Kaftan D, Holland N, Sai PS, Kless H, Scherz A (2006) Protein flexibility acclimatizes photosynthetic energy conversion to the ambient temperature. Nature 442(7104):827–830. doi:10.1038/nature04947
34. Lane MD, Seelig B (2014) Advances in the directed evolution of proteins. Curr Opin Chem Biol 22:129–136. doi:10.1016/j.cbpa.2014.09.013
35. Packer MS, Liu DR (2015) Methods for the directed evolution of proteins. Nat Rev Genet 16(7):379–394. doi:10.1038/nrg3927
36. Moult J, Fidelis K, Kryshtafovych A, Schwede T, Tramontano A (2014) Critical assessment of methods of protein structure prediction (CASP)—round x. Proteins 82(Suppl 2):1–6. doi:10.1002/prot.24452
37. Yue K, Dill KA (1992) Inverse protein folding problem: designing polymer sequences. Proc Natl Acad Sci U S A 89(9):4163–4167
38. Whitehead TA, Baker D, Fleishman SJ (2013) Computational design of novel protein binders and experimental affinity maturation. Methods Enzymol 523:1–19. doi:10.1016/B978-0-12-394292-0.00001-1
39. Rothlisberger D, Khersonsky O, Wollacott AM, Jiang L, DeChancie J, Betker J, Gallaher JL, Althoff EA, Zanghellini A, Dym O, Albeck S, Houk KN, Tawfik DS, Baker D (2008) Kemp elimination catalysts by computational enzyme design. Nature 453(7192):190–195. doi:10.1038/nature06879
40. Tanford C (1978) The hydrophobic effect and the organization of living matter. Science 200 (4345):1012–1018
41. Bowie JU, Luthy R, Eisenberg D (1991) A method to identify protein sequences that fold into a known three-dimensional structure. Science 253(5016):164–170
42. Godzik A, Kolinski A, Skolnick J (1992) Topology fingerprint approach to the inverse protein folding problem. J Mol Biol 227 (1):227–238
43. Carbonell P, Trosset JY (2015) Computational protein design methods for synthetic biology. Methods Mol Biol 1244:3–21. doi:10.1007/978-1-4939-1878-2_1
44. Richter F, Baker D (2013) Computational protein design for synthetic biology. In: Zhao H (ed) Synthetic biology tools and applications. Elsevier Inc., San Diego, CA
45. Quinn TP, Tweedy NB, Williams RW, Richardson JS, Richardson DC (1994) Betadoublet: de novo design, synthesis, and characterization of a beta-sandwich protein. Proc Natl Acad Sci U S A 91(19):8747–8751
46. Kuhlman B, Dantas G, Ireton GC, Varani G, Stoddard BL, Baker D (2003) Design of a novel globular protein fold with atomic-level accuracy. Science 302(5649):1364–1368. doi:10.1126/science.1089427
47. Kaplan J, DeGrado WF (2004) De novo design of catalytic proteins. Proc Natl Acad Sci U S A 101(32):11566–11570. doi:10.1073/pnas.0404387101
48. Joh NH, Wang T, Bhate MP, Acharya R, Wu Y, Grabe M, Hong M, Grigoryan G, DeGrado WF (2014) De novo design of a transmembrane Zn(2)(+)-transporting four-helix bundle. Science 346(6216):1520–1524. doi:10.1126/science.1261172
49. Huang PS, Love JJ, Mayo SL (2007) A de novo designed protein protein interface. Protein Sci 16(12):2770–2774. doi:10.1110/ps.073125207
50. Desjarlais JR, Handel TM (1995) De novo design of the hydrophobic cores of proteins. Protein Sci 4(10):2006–2018. doi:10.1002/pro.5560041006
51. Ventura S, Vega MC, Lacroix E, Angrand I, Spagnolo L, Serrano L (2002) Conformational strain in the hydrophobic core and its

implications for protein folding and design. Nat Struct Biol 9(6):485–493. doi:10.1038/nsb799

52. Keating AE, Malashkevich VN, Tidor B, Kim PS (2001) Side-chain repacking calculations for predicting structures and stabilities of heterodimeric coiled coils. Proc Natl Acad Sci U S A 98(26):14825–14830. doi:10.1073/pnas.261563398
53. Slovic AM, Kono H, Lear JD, Saven JG, DeGrado WF (2004) Computational design of water-soluble analogues of the potassium channel KcsA. Proc Natl Acad Sci U S A 101 (7):1828–1833. doi:10.1073/pnas.0306417101
54. Slovic AM, Summa CM, Lear JD, DeGrado WF (2003) Computational design of a water-soluble analog of phospholamban. Protein Sci 12(2):337–348. doi:10.1110/ps.0226603
55. Voet AR, Noguchi H, Addy C, Simoncini D, Terada D, Unzai S, Park SY, Zhang KY, Tame JR (2014) Computational design of a self-assembling symmetrical beta-propeller protein. Proc Natl Acad Sci U S A 111 (42):15102–15107. doi:10.1073/pnas.1412768111
56. Woolfson DN, Bartlett GJ, Bruning M, Thomson AR (2012) New currency for old rope: from coiled-coil assemblies to alpha-helical barrels. Curr Opin Struct Biol 22(4):432–441. doi:10.1016/j.sbi.2012.03.002
57. Lanci CJ, MacDermaid CM, Kang SG, Acharya R, North B, Yang X, Qiu XJ, DeGrado WF, Saven JG (2012) Computational design of a protein crystal. Proc Natl Acad Sci U S A 109 (19):7304–7309. doi:10.1073/pnas.1112595109
58. Swift J, Wehbi WA, Kelly BD, Stowell XF, Saven JG, Dmochowski IJ (2006) Design of functional ferritin-like proteins with hydrophobic cavities. J Am Chem Soc 128 (20):6611–6619. doi:10.1021/ja057069x
59. Summa CM, Rosenblatt MM, Hong JK, Lear JD, DeGrado WF (2002) Computational de novo design, and characterization of an A(2)B (2) diiron protein. J Mol Biol 321(5):923–938
60. Cochran FV, Wu SP, Wang W, Nanda V, Saven JG, Therien MJ, DeGrado WF (2005) Computational de novo design and characterization of a four-helix bundle protein that selectively binds a nonbiological cofactor. J Am Chem Soc 127(5):1346–1347. doi:10.1021/ja044129a
61. Shifman JM, Mayo SL (2003) Exploring the origins of binding specificity through the computational redesign of calmodulin. Proc Natl Acad Sci U S A 100(23):13274–13279. doi:10.1073/pnas.2234277100
62. Kortemme T, Joachimiak LA, Bullock AN, Schuler AD, Stoddard BL, Baker D (2004) Computational redesign of protein-protein interaction specificity. Nat Struct Mol Biol 11 (4):371–379. doi:10.1038/nsmb749
63. Potapov V, Reichmann D, Abramovich R, Filchtinski D, Zohar N, Ben Halevy D, Edelman M, Sobolev V, Schreiber G (2008) Computational redesign of a protein-protein interface for high affinity and binding specificity using modular architecture and naturally occurring template fragments. J Mol Biol 384 (1):109–119. doi:10.1016/j.jmb.2008.08.078
64. Lippow SM, Wittrup KD, Tidor B (2007) Computational design of antibody-affinity improvement beyond in vivo maturation. Nat Biotechnol 25(10):1171–1176. doi:10.1038/nbt1336
65. Dagliyan O, Shirvanyants D, Karginov AV, Ding F, Fee L, Chandrasekaran SN, Freisinger CM, Smolen GA, Huttenlocher A, Hahn KM, Dokholyan NV (2013) Rational design of a ligand-controlled protein conformational switch. Proc Natl Acad Sci U S A 110 (17):6800–6804. doi:10.1073/pnas.1218319110
66. Korendovych IV, Kulp DW, Wu Y, Cheng H, Roder H, DeGrado WF (2011) Design of a switchable eliminase. Proc Natl Acad Sci U S A 108(17):6823–6827. doi:10.1073/pnas.1018191108
67. Yin H, Slusky JS, Berger BW, Walters RS, Vilaire G, Litvinov RI, Lear JD, Caputo GA, Bennett JS, DeGrado WF (2007) Computational design of peptides that target transmembrane helices. Science 315(5820):1817–1822. doi:10.1126/science.1136782
68. Samish I (2009) Search and sampling in structural bioinformatics. In: Bourne P, Gu J (eds) Structural bioinformatics. Wiley, New York, pp 207–236
69. Dunbrack RL Jr, Karplus M (1993) Backbone-dependent rotamer library for proteins. Application to side-chain prediction. J Mol Biol 230 (2):543–574. doi:10.1006/jmbi.1993.1170
70. Dunbrack RL Jr (2002) Rotamer libraries in the 21st century. Curr Opin Struct Biol 12 (4):431–440
71. Shapovalov MV, Dunbrack RL Jr (2011) A smoothed backbone-dependent rotamer library for proteins derived from adaptive kernel density estimates and regressions. Structure 19(6):844–858. doi:10.1016/j.str.2011.03.019

72. Subramaniam S, Senes A (2014) Backbone dependency further improves side chain prediction efficiency in the Energy-based Conformer Library (bEBL). Proteins 82(11):3177–3187. doi:10.1002/prot.24685
73. Kuhlman B, Baker D (2000) Native protein sequences are close to optimal for their structures. Proc Natl Acad Sci U S A 97 (19):10383–10388
74. Grigoryan G, Degrado WF (2011) Probing designability via a generalized model of helical bundle geometry. J Mol Biol 405 (4):1079–1100. doi:10.1016/j.jmb.2010.08.058
75. Schramm CA, Hannigan BT, Donald JE, Keasar C, Saven JG, Degrado WF, Samish I (2012) Knowledge-based potential for positioning membrane-associated structures and assessing residue-specific energetic contributions. Structure 20(5):924–935. doi:10.1016/j.str.2012.03.016
76. Xu F, Zahid S, Silva T, Nanda V (2011) Computational design of a collagen A:B:C-type heterotrimer. J Am Chem Soc 133 (39):15260–15263. doi:10.1021/ja205597g
77. Shifman JM, Mayo SL (2002) Modulating calmodulin binding specificity through computational protein design. J Mol Biol 323 (3):417–423
78. Havranek JJ, Harbury PB (2003) Automated design of specificity in molecular recognition. Nat Struct Biol 10(1):45–52. doi:10.1038/nsb877
79. Bolon DN, Grant RA, Baker TA, Sauer RT (2005) Specificity versus stability in computational protein design. Proc Natl Acad Sci U S A 102(36):12724–12729. doi:10.1073/pnas.0506124102
80. Grigoryan G, Reinke AW, Keating AE (2009) Design of protein-interaction specificity gives selective bZIP-binding peptides. Nature 458 (7240):859–864. doi:10.1038/nature07885
81. Fry HC, Lehmann A, Saven JG, DeGrado WF, Therien MJ (2010) Computational design and elaboration of a de novo heterotetrameric alpha-helical protein that selectively binds an emissive abiological (porphinato)zinc chromophore. J Am Chem Soc 132(11):3997–4005. doi:10.1021/ja907407m
82. Koga N, Tatsumi-Koga R, Liu G, Xiao R, Acton TB, Montelione GT, Baker D (2012) Principles for designing ideal protein structures. Nature 491(7423):222–227. doi:10.1038/nature11600
83. Fry HC, Lehmann A, Sinks LE, Asselberghs I, Tronin A, Krishnan V, Blasie JK, Clays K, DeGrado WF, Saven JG, Therien MJ (2013) Computational de novo design and characterization of a protein that selectively binds a highly hyperpolarizable abiological chromophore. J Am Chem Soc 135 (37):13914–13926. doi:10.1021/ja4067404
84. Jamroz M, Kolinski A (2010) Modeling of loops in proteins: a multi-method approach. BMC Struct Biol 10:5. doi:10.1186/1472-6807-10-5
85. Hildebrand PW, Goede A, Bauer RA, Gruening B, Ismer J, Michalsky E, Preissner R (2009) SuperLooper—a prediction server for the modeling of loops in globular and membrane proteins. Nucleic Acids Res 37:W571–W574. doi:10.1093/nar/gkp338
86. Soto CS, Fasnacht M, Zhu J, Forrest L, Honig B (2008) Loop modeling: sampling, filtering, and scoring. Proteins 70(3):834–843. doi:10.1002/prot.21612

Chapter 2

Achievements and Challenges in Computational Protein Design

Ilan Samish

Abstract

Computational protein design (CPD), a yet evolving field, includes computer-aided engineering for partial or full de novo designs of proteins of interest. Designs are defined by a requested structure, function, or working environment. This chapter describes the birth and maturation of the field by presenting 101 CPD examples in a chronological order emphasizing achievements and pending challenges. Integrating these aspects presents the plethora of CPD approaches with the hope of providing a "CPD 101". These reflect on the broader structural bioinformatics and computational biophysics field and include: (1) integration of knowledge-based and energy-based methods, (2) hierarchical designated approach towards local, regional, and global motifs and the integration of high- and low-resolution design schemes that fit each such region, (3) systematic differential approaches towards different protein regions, (4) identification of key hot-spot residues and the relative effect of remote regions, (5) assessment of shape-complementarity, electrostatics and solvation effects, (6) integration of thermal plasticity and functional dynamics, (7) negative design, (8) systematic integration of experimental approaches, (9) objective cross-assessment of methods, and (10) successful ranking of potential designs. Future challenges also include dissemination of CPD software to the general use of life-sciences researchers and the emphasis of success within an in vivo milieu. CPD increases our understanding of protein structure and function and the relationships between the two along with the application of such know-how for the benefit of mankind. Applied aspects range from biological drugs, via healthier and tastier food products to nanotechnology and environmentally friendly enzymes replacing toxic chemicals utilized in the industry.

Key words Computational protein design, Inverse folding problem, De novo design, Directed evolution, Rational design, Synthetic biology, Negative design, Enzyme design, Protein–protein interaction

> *"The abundance of substances of which animals and plants are composed of, the remarkable processes whereby they are formed and then broken down again claimed the attention of mankind, and hence from the early days they also persistently captivated the interest of chemists. . . . To determine the structure of the molecule the chemist proceeds in a similar way to the anatomist. By chemical actions he breaks the system down into its components and continues with this division until familiar substances emerge. Where this decomposition has taken different directions, the structure of the original system can be inferred from the decomposition products. Usually, however, the structure will only be finally elucidated by the reverse method, by building up the molecule from the decomposition products or similar substances, i.e. by what is termed synthesis. Nevertheless, the chemical enigma of Life will not be solved until organic chemistry has mastered another, even more difficult subject, the proteins, in the same*

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_2,

way as it has mastered the carbohydrates. It is hence understandable that the organic and physiological chemists are increasingly turning their attention to it. . . .”
Emil Fischer, Nobel Lecture, December, 12th 1902

1 Introduction: The Birth of Computational Protein Design

In *1902* Emil Fischer's Nobel lecture [1] presented the idea of protein design (*see* exert). He emphasized that molecules can be elucidated only by the reverse method, namely, design from decomposition products, which in the case of proteins are the amino acids. At the time Fischer stated that proteins are far more difficult than carbohydrates, for which he received the Nobel. Indeed, it was only in *1972* that Chris Anfinsen received a Nobel Prize for the “connection between the amino acid sequence and the biologically active conformation.” Anfinsen's famous experiment included denaturing and renaturing ribonuclease A; thus setting the stage for the sequence-structure–function relationships underlying protein science [2]. In *1981* Drexler speculated that it should be possible to design novel proteins and that such proteins could provide a general capability for molecular manipulation [3]. In *1983* Pabo wrote about designing proteins and peptides concluding that it may be difficult to design proteins which carry out a particular function but the use of pre-folded backbone configuration may be useful at this stage [4]. Pabo pointed at the so called *inverse folding problem* of using a known backbone conformation on which new sequences can be applied; thus modifying function. In agreement with Pabo, in *1987* Wodak reviewed the field with the title “computer-aided design in protein engineering” where the key features of CPD were laid out in a manner that is accurate till this very day, and not only in e.g. the Wodak lab's DESIGNER [5, 6] CPD software.

In *1985* DeGrado conducted what should be regarded as the first CPD: a design, synthesis, and characterization of a 17-residue helical peptide that was the tightest calmodulin-binding peptide produced [7]. This first CPD attempt, described in more detail below, includes many of the main features of current CPD including the need to produce and characterize the suggested design, the crosstalk between human and computer input and the iterative feedback process of the CPD scheme to learn and improve the design.

Other early attempts were “computer-aided” by visually inspecting the protein for suggesting specific point mutations. For example, in *1985* Rutter and coworkers replaced two glycines by alanines in the binding site of trypsin, thus altering binding specificity [8].

While DeGrado and others used computer-aided protein design in early days, according to PubMed, the term “protein design” was introduced only in *1986* by Vonderviszt, Matrai, and Simon [9]. As in the talk of Fischer, Simon's paper did not focus on the protein design per se. Rather, they implied the potential use of analysis of protein environment trends as parameterization required

for protein design. It took an additional decade for the term "computational protein design" to enter the literature. In *1997*, Dahiyat, Sarisky, and Mayo introduced the term as part of a systematic design of a ββα motif (Table 1) in which they designed 20 of the 28 motif residues [18, 19]. Early attempts of CPD often did not use this term despite describing science that is in the core of the CPD field till this very day. In parallel, numerous CPD publications refer to CPD with related terms that relate to protein design but do not focus on the related computational methodology. These include protein design, synthetic biology, rational design, and more.

Of special note is the fuzzy division between "protein design" and CPD as often there is a significant contribution from computational tools to protein designs that are conducted with an expert know-how that is formulated by computation. This review will emphasize attempts of computer-assisted designs but will focus on protein designs in which the computational part is central to the design methodology.

Thus, in a century since Fischer's visionary Nobel lecture, science has moved from yearning to understanding protein structure by designing it from building blocks to applying a computational general design algorithm. Not less important, protein design is often termed "the inverse folding problem" as the success of using building blocks to fold a protein into a given structure and function is the true proof that folding is well-understood. Consequently, the know-how and success of CPD contribute directly to that of protein structure prediction in healthy and diseased proteins. Within these frameworks, the CPD field is constantly growing into new basic- and applied-scientific research.

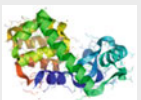
Here, rather than providing an overview of methodological components [121, 122], the idea is to present CPD examples in chronological order showing the achievements and pending challenges in a timeline perspective. In other words, rather than providing a grocery list of available computationally assisted protein design, this review is aimed towards presenting the state of the field as it evolves on the chronological milestone road. Taken together, these case-studies encompass the breadth of the CPD field, the plethora of distinct flavors of it as well as the common threads of success and pitfalls computational protein designers are encountered with (Table 1). The concluding remarks focus on the latter; providing scientific questions for years to come.

2 The First Decade of Computational Protein Design, 1985–1994

In *1985* DeGrado, a leading pioneer in protein design, designed with coworkers the tightest-binding peptide inhibitors of calmodulin known till then [7]. Computationally, the 17-residue helical peptide designs included computer-graphics based modeling of the calmodulin target as well as computer modeling [123] of the

Table 1

Examples of CPD case-studies including the main CPD method and experimental characterization and validation tools utilized to characterize the resulting design. All pictures were drawn with PyMol. In case of NMR models, only the first model is presented. In relevant cases the biological assembly rather than the asymmetric unit is presented. Secondary structures are shown by "dumbbell" cartoon highlighting their irregularities. For clarity, structural water was omitted

	Novelty	Target	Computational methods	Main characterization methods	PDB (Å resolution)	Year, first and last authors, journal, and references	Protein structure (first PDB only)
1.	Helical calmodulin-binding peptides	Peptides binding calmodulin	Homology modeling, binding site characterization, iterative synthesis	Calmodulin binding to Melex, Trp fluorescence, CD	–	1985 DeGrado, Cox, *J Cell Biochem* [7]	
2.	De novo design of a four-helix bundle (*Felix*)	Antiparallel four-helix bundle (79 residues)	Design rules followed by MD and structural modeling	CD, SEC, Fluorescence	1flx (theoretical model)	1990 Hecht, Ogden [10]	
3.	Grafted metal-binding site	Thioredoxin	DEZYMER	In vivo activity, CD, EPR, absorption, and fluorescence spectroscopy	–	1991 Hellinga, Richards, *JMB* [11]	
4.	Enzyme design for a non-natural substrate	α-lytic protease	Single mutation free energy perturbation (FEP) with molecular mechanics (AMBER) and rotamer library (PROPAK)	Kinetic measurements of design and specificity	–	1991 Wislon, Agard, *JMB* [12]	
5.	Hydrophobic core design	Bacteriophage T4 lysozyme	Modified rotamer library and scoring function including standard force-field and conformational entropy components	X-ray, CD	1177 (2.05 Å), 2178 (2.0 Å), 1179 (1.9 Å), 1180 (1.8 Å), 1181 (2.0 Å), 1182 (2.1 Å)	1992, Hurley, Matthews, *JMB* [13]	
6.	β-sandwich de novo design (*Betadoublet*)	β-sandwich protein	SYBYL (Tripos), CHAOS, GCG	CD, NMR H/D exchange	1btd (theoretical)	1994 Quinn, Richradson, *PNAS* [14]	

7.	Hydrophobic core design	Phage 434 cro protein (five to eight core residues)	Rotamer library + genetic algorithm within Repacking of Core (ROC)	CD, NMR	–	1995 Desjarlais Hnadel, *Prot Sci* [15]	
8.	Hydrophobic core design	Ubiquitin	Nine designs, each with three to eight core-residue mutations, ROC	CD, Fluorescence, ANS binding, NMR	1ud7 (NMR)	1997, Lazar, Handel, *Prot Sci* [16] 1999, Johnson, Handel, *Structure* [17]	
9.	Full sequence design 1 (*FSD-1*)	ββα motif typified by the zinc finger DNA binding	20 of 28 motif positions were designed avoiding metal and Cys. Template = 1zaa	CD, NMR structure backbone root mean square deviation (RMSD) = 1.04 Å	1FSD, 1FSV (NMR)	1997 Dahiyat, Mayo, *Science* [18, 19]	
10.	Hyper-thermophile protein	Protein G—β1 domain (Gβ1)	7 mutations selected via ORBIT	CD, AUC, NMR, kinetics (Biacore)	1gb4 (NMR)	1998, Malakauskas, May, *Nat Struct Biol* [20]	
11.	Coiled-coil oligomers	Right-handed dimer, trimer, and tetramer coiled-coils	Hydrophobic-polar residue patterning and side-chain packing. Comparison to mutations in other locations (representing a less folded state)	X-ray, CD	1rh4 (1.9 Å)	1998, Harbury, Kim, *Science* [21]	
12.	Three-helix bundle	De novo designed 73-residue, single chain antiparallel three-helix bundle *α3C, α3D*	ROC and NBSEARCH for core design, InsightII for modeling, Discover for minimization	NMR, SEC, AUC, CD, Fluorescence, ANS-binding, H-D exchange	2a3d (NMR)	1998 [22] (Design). 1999, Walsh, DeGrado, *PNAS* [23] (structure)	

(continued)

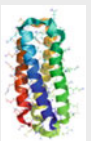
Table 1
(continued)

	Novelty	Target	Computational methods	Main characterization methods	PDB (Å resolution)	Year, first and last authors, journal, and references	Protein structure (first PDB only)
13.	Coiled-coil mimetic	GCN4-derived mimetic of interleukin-4 two-helix coiled coil	SMD, MD	CD, NMR, SPR	–	1999 Domingues, Serrano, *Nat Struct Bio* [24]	
14.	Dimerization of a monomer (VPA)	Dimerizing protein L-based on a strand-swapped dimer	RosettaDesign 3 positions were design in a eight-residue redesign	CD, X-ray	1jml (1.9 Å)	2001, Kuhlman, Baker, *PNAS* [25]	
15.	β-sheet peptide automatic redesign	Automatic redesign of one to three residues in the *betanova* β-sheet peptide design producing an array of peptides	PERLA	CD, NMR	NMR available (not deposited to PDB)	2001, Lopez de la Paz, Serrano, *JMB* [26]	
16.	Protozyme enzyme-like protein	Thioredoxin-based PNPA hydrolase	94 non-glycine positions designed via ORBIT	Enzyme kinetics (product production), MS	–	2001, Bolon, Mayo, *PNAS* [27]	
17.	Heterdimeric coiled coils	Six heterodimeric coiled coils	Knowledge-based coiled coil rules, designated rotamer library (sampled via DEE/A*) and minimization with solvation correction	X-ray, CD, AUC,	1kd8 (1.9 Å), 1kd9 (2.1 Å), 1kdd (2.14 Å)	2001, Keating, Kim, *PNAS* [28]	

18.	Diiron binding (*DFtet*)	A_2B_2 four-helix bundle binding diiron	Genetic algorithm fitting to a previous smaller structure which was elongated following coiled-coil parameters. 700,000 sequence design runs and contact energetics relative to competing topologies	CD, AUC, Co^{2+} titration, ferroxidase activity	–	2002, Summa, DeGrado, *JMB* [29]	
19.	Hydrophobic core	13 Spectrin SH3 core redesigns	PERLA, rotamers with 5° sub-rotameric states for χ_1 and χ_2	X-ray, fluorescence	1e6h (2.0 Å), 1e6g (2.3 Å), 1h8k (1.8 Å)	2002, Ventura, Serrano, *Nat Struct Biol* [30]	
20.	Binding specificity	Calmodulin binding to smMLCK	Eight residues were mutated via ORBIT modified for protein–protein interactions	CD, binding	–	2002, 2003 Shifman, Mayo *JMB* [31], PNAS [32]	
21.	Antibiotic resistance redesign	Changing β-lactamase specificity from ampicillin- to cefotaxime-resistance	19 residues at the interface region were redesigned with PDA, MC/SA and experimental screening of a ~200,000 sequence library	Antibiotic-resistance minimum inhibitory concentration (MIC) assay	–	2002, Hayes, Dahiyat, *PNAS* [33]	
22.	Core stabilization	Human growth hormone (hGH) core stabilization	11 mutations were applied with the 45 target core residues, PDA, new scoring function with backbone and side-chain entropy component and newly weighted other components	CD, cell proliferation assay	–	2002, Filikov, Dahiyat, *Prot Sci* [34]	
23.	Core stabilization	G-CSF core stabilization	10–14 mutations within the 35 redesigned core residues, PDA, homology modeling of template	CD, storage stability, cell-proliferation, assay, biological activity, pharmako-kinetics	–	2002, Luo, Dahiyat, *Prot Sci* [35]	

(continued)

Table 1
(continued)

	Novelty	Target	Computational methods	Main characterization methods	PDB (Å resolution)	Year, first and last authors, journal, and references	Protein structure (first PDB only)
24.	Artificial specific endonuclease (*E-Drel*)	Engineered I-Dmol/I-Crel endonuclease where these domains are fused and which targets a long chimeric DNA target	Three-residue inter-domain linker and 14 interface residues identified by computational Ala scanning and redesigned with RosettaDesign	X-ray, lacZ-based solubility assay, biochemical DNA cleavage assay	1 mow (2.4 Å)	2002, Chevalier, Stoddard, *Mol Cell* [36]	
25.	MHC-I binding peptides	Automatic design of nine-residue peptides inhibiting MHC-I	DESIGNER, correlation with a dataset of known MHC-I binding peptides	Fluorescence (peptide-protein binding), T-cell response inhibition (ELISPOT)	–	2003, Ogata, Wodak, *JBC* [37]	
26.	Water solubilization of a membrane protein	Water-soluble pentameric phospholamban	Ten water-facing residues were mutated optimizing pairwise interactions with MC/SA sampling	X-ray, CD, AUC, SEC	1yod (1.8 Å)	2003, Slovic, DeGrado *Prot Sci* [38] (design) 2005 *JMB* [39] (structure)	
27.	Binding specificity	GCN4-derived dimerization specificity	Genetic algorithm for multi-state positive- and negative-design	CD, binding thermodynamics	–	2003, Havranek, Harbury, *Nat Struct bio* [40]	
28.	De novo metal-binding design	Due ferro four-helix-bundle	88 out of 114 residues were designed by SCADS	AUC, CD, metal-binding, NMR	2hz8 (NMR + QM/MM refinement)	2003 Calhoun, Saven, *JMB* [41] 2008 *Structure* [42]	

29.	Novel fold with atomic-level accuracy (*Top7*)	93-residue novel αβ fold with five β-strands and two α-helices	Rosetta package: 3D backbone via fragments followed by side-chain MC	CD, X-ray, GuHCl denaturation, NOESY & HSQC NMR	1qys (2.5 Å)	2003 Kuhlman, Baker, *Science* [43]	
30.	De novo domain redesign	WW domain (three antiparallel β-strands)	SPANS (SPA for numerous states)	CD, NMR	–	2003, Kraemer-Pecore, Desjarlais, *Prot Sci* [44]	
31.	Protein redesign (increased folding kinetics and thermostability)	Human procarboxypeptidase A2 and 8 other folds	RosettaDesign	X-ray, NMR, CD, AUC, GuHCL	1vjq (2.1 Å) 2gjf (NMR)	2003, 2007, Dantas, Baker *JMB* [45, 46]	
32.	Enzyme-design	Phenol-oxidase	Manual binding pocket sculpting in CPD protein	CD, size-exclusion chromatography, binding kinetics	1jmb (2.2 Å) (template-like design)	2002, 2004, Kaplan, DeGrado *PNAS* [29, 47, 48]	
33.	PPI specificity	PPI specificity in DNase (colicin E7) -inhibitor protein (immunity protein Im7) pairs	1. Modeling point mutations at interface sights of both binding partners and then combining the single point mutations. 2. Sampling alternate rigid-body orientations followed by interface hydrogen bond design facilitated by previous structure	X-ray, SPR, fluorescence, toxicity plate assay, DNase activity	1ujz (2.1 Å), 2erh (2.0 Å)	2004, Kortemme, Baker *NSMB* and 2006 Joachimiak, Baker *JMB* [49, 50]	
34.	Water-soluble potassium channel	Water-soluble KcsA potassium channel	SCADS, 35 solvent-exposed residues were designed with an environmental energy fit to solvent-exposure	CD, AUC, SEC, toxin-binding	–	2004, Slovic, DeGrado, *PNAS* [51]	

(continued)

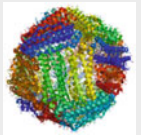
Table 1
(continued)

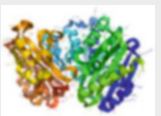
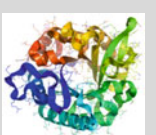
	Novelty	Target	Computational methods	Main characterization methods	PDB (Å resolution)	Year, first and last authors, journal, and references	Protein structure (first PDB only)
35.	Thermostabilization of an enzyme	Thermostabilization of homodimeric hydrolase enzyme yeast cytosine deaminase (yCD)	Rosetta redesign of residues not in the dimer interface or catalytic site	X-ray, CD, kinetic measurements	1ysd (1.9 Å), 1ysb (1.7 Å)	2005, Korkegian, Stoddard, *Science* [52]	
36.	Negative design for reengineering a homodimer to a heterodimer	SspB homodimer	ORBIT, modified by capping unfavorable vdW energetics and adding an MC negative-design module	X-ray, GuHCl-denaturation, chromatography dimerization assay	1zsz (2.0 Å)	2005, Bolon, Sauer, *PNAS* [53]	
37.	Redox-active rubredoxin mimic	Minimal (40-residue) de novo designed rubredoxin mimic	SCADS	UV-vis, CD	–	2005, Nanda, DeGrado, *JACS* [54]	
38.	Nonbiological cofactor binding	Four-helix bundle that selectively binds two DPP-Fe	Backbone design following different constrains followed by three rounds of SCADS sequence design	CD, SEC, AUC, UV-Vis, EPR	–	2005, Cochran, DeGrado, *JACS* [55]	
39.	Ferritin-like protein surface hydrophobicity	12-subunit Dps ferritin-like protein. Each four-helix bundle subunit underwent three to ten hydrophilic → hydrophobic mutations	SCADS modified for symmetry (homo-oligomers)	CD, MS, SEC, iron mineralization, Dynamic light scattering	–	2006, Dmochowski, Swift, *JACS* [56]	
40.	Binding selectivity design	TRAIL-variant for selective DR5 binding	WHATIF, FOLD-X	SPR, FACS	–	2006, van der Sloot, Quax *PNAS* [57]	

41.	Endonuclease DNA binding and specificity redesign	Redesign of intron-encoded homing endonuclease I-MsoI	RosettaDesign protocol of MC optimization of discrete side-chain conformations and identities with continuous minimization of protein and DNA dihedral angles	X-ray, competitive cleavage assay, binding assay	2fld (2.0 Å)	2006, Ashworth, Baker, *Nature* [58]	
42.	Antibody Fc variants	Antibody Fc variants with enhances Fcγ-receptor-mediated effector function. Four sites were mutated	Protein design automation (PDA), Sequence prediction algorithm (SPA)	SPR, ADCC assay, B-cell depletion (in macaques), AlphaScreen	–	2006, Lazar, Dahiyat, *PNAS* [59]	
43.	Protein–protein interface	Protein G—β1 domain (Gβ1)	24 interface residues, side-chain-pruned docking of 18 complexes to derive docking parameterization, ORBIT, RESCLASS	HSQC NMR, AUC, dissociation kinetics		2007, Huang, Mayo, *Prot Sci* [60]	
44.	Anti-transmembrane helix peptide (*CHAMP*)	Computed Helical Anti Membrane Protein Peptide binding α_{IIb} and α_v integrin subunits	Protcad, scwrl3, GROMOS minimization, HELANAL	CD, ATR-IR, FRET, DN-TOXCAT, Hemolysis, platelet aggregation, and adhesion	–	2007, Yin, DeGrado, *Science* [61]	
45.	De novo loop design	Tenascin-C ten-residue loop	Ten-residue loop design via Rosetta, loop fragments, and filtering	X-ray, ^{1}H NMR, CD	2rb8 (1.45 Å), 2rbl (2.1 Å)	2007, Hu, Kuhlman, *PNAS* [62]	
46.	De novo design of a erythropoietin (EPO) receptor binder	Design by grafting three binding residues from the EPO—EPO receptor complex and grafting them on a new scaffold	PDB search for scaffold supporting the grafted residues, filtering by protein–protein interface native-like measurables	CD, SPR, Luciferase Reporter Assay		2007, Liu, Lai, *PNAS* [63]	

(continued)

Table 1
(continued)

	Novelty	Target	Computational methods	Main characterization methods	PDB (Å resolution)	Year, first and last authors, journal, and references	Protein structure (first PDB only)
47.	De novo thermostable redesign	51-residue *Drosophila* engrailed homeodomains (two designs)	1. MC with simulated annealing. 2. FASTER with HERO search	CD, NMR	2p6J (NMR)	2007, Shah, Mayo, *JMB* [64]	
48.	Single-chain porphyrin-binding four-helix bundle	108-residue asymmetric diphenyl-porphyrin metalloprotein	SCADS, STITCH, MC/SA, MD	NMR, CD, UV-vis, SEC, AUC	–	2007, Bender, DeGrado, *JACS* [65]	
49.	Antibody affinity	Electrostatics-based ranking of antibody affinity improvement validated on anti-epidermal growth factor and anti-lysozyme antibodies	DEE/A* sampling of rigid backbone. Bound-case binding energetics and unbound-case approximating flexible binding. Poisson-Boltzmann continuum electrostatics and continuum solvent van der Waals were used for design success ranking	Antibodies displayed on yeast surface with binding affinity measured by incubating cells with varying antigen concentrations	–	2007, Lippow, Tidor, *Nature Biotech* [66]	
50.	PPI interface	TEM-BLIP	PDBmodDesign	SPR, double-mutant cycles	–	2008, Potapov, Schreiber, *JMB* [67]	
51.	Nobel metal binding in Ferritn	Human H ferritin designed to bind Noble metals that form nanoparticles	Four inner-surface and four outer-surface mutations in each of the 24 subunits. SCADS modified for symmetry	X-ray, MS, CD, DLS, SPR, transmission electron microscopy (TEM)	3erz (3.06 Å, Hg^{2+} complex), 2z6m (2.72 Å, apoprotein), 3es3 (2.8 Å Au^{3+} complex)	2008, Butts, Dmochowski, *Biochemistry* [68]	

52.	PPI interface	SHV-1-BLIP	EGAD two-state minimizer DEE for bound and unbound with MC followed by heuristic step and minimization	X-ray, kinetics measurements	3c4p (1.7 Å) 3c4o (1.7 Å)	2008, Renolds, Handel, *JMB* [69]	
53.	Coiled-coils with four metallo-porphyrin arrays	55-residue four-helix coiled-coil binding four nonbiological iron porphyrins	Three heptad repeats were added to the previous design of a 34-residue peptide assembling to a coiled coil. Specific mutations were introduced to maintain an antiparallel orientation	SEC, CD, binding stoichiometry, AUC, EPR	–	2008, McAllister, DeGrado, *JACS* [70]	
54.	Enzyme-design	Retro-aldolase	RosettaDesign & RosettaMatch	X-ray, array of enzyme kinetics assays	3b5v, superseded by 3hoj (2.2 Å) and 3b5l (1.8 Å)	2008, Jiang, Baker, *Science* [71]	
55.	Enzyme-design	Kemp-eliminase	RosettaDesign & RosettaMatch	X-ray, array of enzyme kinetics assays, in-vitro evolution	2rkx (2.25 Å)	2008, Rothlisberger, Tawfik, Baker, *Nature* [72]	
56.	Loop remodeling for altering enzyme specificity	Guanine deaminase	Two to five residue loop modeling fitting a hypothetical enzyme-ligand transition state		3e0l (2.37 Å)	2008, Murphy, Baker, *PNAS* [73]	

(continued)

Table 1
(continued)

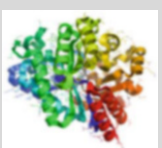
	Novelty	Target	Computational methods	Main characterization methods	PDB (Å resolution)	Year, first and last authors, journal, and references	Protein structure (first PDB only)
57.	Altering binding specificity	Calmodulin–CamKII interaction	18 residue redesign with ORBIT focusing on inter-molecular interactions	CD, SPR	–	2009, Yosef, Shifman, *JMB* [74]	
58.	Binding peptides with high specificity	bZIP-binding peptides including against c-Jun, c-Fox, and c-Maf	CLASSY, negative design	Protein arrays	–	2009, Grigoryan, Keating, *Nature* [75]	
59.	Protein–DNA interaction specificity	Monomeric homing endonuclease I-AniI	Eight designs, each with up to six residues redesigned by Rosetta with flexible backbone and a genetic algorithm	Kinetic assays, fluorescence competition binding assay	–	2009, Thyme, Baker, *Nature* [76]	
60.	Altered nonribosomal peptide synthetase enzyme gramicidin S synthetase A (GrsA-PheA) activity	Phenylalanine adenylation domain of GrsA-PheA for a set of noncognate substrates (Leu and charged residues)	K* including active-site (two mutations) and nonactive site "bolstering" mutations (up to two mutations)	Spectrophotometric activity assay	–	2009, Chen, Donald, *PNAS* [77]	
61.	Binder of an abiological chromophore	A_2B_2 four-helix bundle that selectively binds two emissive abiological (porphinato)zinc chromophores of DPP-Zn	SCADS	CD, SEC, AUC, transient absorption spectroscopy, fluorescence lifetime measurements	–	2010, Fry, Therien, *JACS* [78]	
62.	Protein–DNA interaction specificity	I-Msol homing endonuclease	Six residues redesigned by Rosetta, loop-closure algorithm, and a genetic algorithm	X-ray, DNA-cleavage assay	3mip (2.4 Å), 3mis (2.3 Å), and 3ko2 (2.9 Å)	2010, Ashworth, Baker, *NAR* [79]	

63.	Generation of longer emission wavelength in red fluorescent protein	Red fluorescent protein mCherry	13 residues redesigned via Phoenix and FASTER. The CPD was used to generate focused combinatorial libraries	mRojoA, mRouge, MS, site-directed mutagenesis, library-screening, spectroscopic characterization	3nez (1.7 Å), 3ned (0.95 Å)	2010, Chica, Mayo, *PNAS* [80]	
64.	TM diporphyrin-binding complex *PRIME* (PorPhyrins in Membrane)	24-residue TM helices assembling to a four-helix-bundle binding two diphenyl-porphyrins (Fe^{3+}DPP's)	Backbone template of an idealized porphyrin-binding four-helix bundle, extended energy-based conformer library, DEE search followed by MC/SCMF. Pairwise energies calculated with the addition of the Lazaridis implicit membrane solvation (IMM1), model ranking by helix–helix interaction energies	Characterization in a micelle using absorption spectroscopy, CD, and EPR	–	2010, Korendovich, DeGrado, *JACS* [81]	
65.	Protein–protein interaction (PPI)	Hyperplastic discs protein binding to P21-activated kinase 1 kinase domain PAK1	Rosetta-based DDMI protocol (Dock, Design, Minimize Interface)	NMR, CD, fluorescence polarization	(Chemical shifts ID 1670 in BMRB DB)	2010, Jha, Kuhlman, *JMB* [82]	
66.	Multistate design for stabilization of a protein core	Gβ1, the β1 domain of *Streptococcal* protein G	Ten core positions underwent multistate CPD using NMR-derived ensemble templates. FASTER, CLEARSS library design	Combinatorial library GdmCl-based stability determination	–	2010, Allen, Mayo, *PNAS* [83]	
67.	Prediction of resistance mutations	MRSA DHFR resistance mutations (maintaining function without inhibitor binding)	10 active-site residues subjected to K* CPD algorithm. Four mutants experimentally validated	X-ray, CD, enzyme kinetics	3LG4 (V31Y/F92I double mutant, 3.15 Å)	2010, Frey, Anderson, *PNAS* [84]	

(continued)

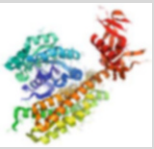
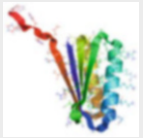
Table 1
(continued)

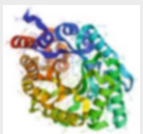
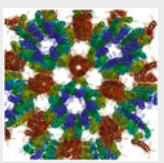
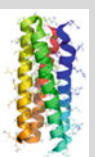
	Novelty	Target	Computational methods	Main characterization methods	PDB (Å resolution)	Year, first and last authors, journal, and references	Protein structure (first PDB only)
68.	Carbon nanotube surface coating (*HexCoil-Gly, HexCoil-Ala, DSD-Gly, DSD-Ala*)	Single-walled carbon nanotube coating with hexameric coiled coil	Three selection rules for backbone choice followed by side-chain design of DEE/A*, minimization, MC/SA	X-ray, CD, AUC, SEC, Absorption spectroscopy, TEM	3s0r (2.44 Å)	2011, Grigoryan, DeGrado, *Science* [85]	
69.	High-affinity protein binder (*HB36, HB80*)	Influenza Hemagglutinin binder	Target-surface scaffold selection, RosettaDock, RosettaDesign + directed evolution	X-ray, SPR, Bio-layer interferometry	3r2x (3.1 Å)	2011, Fleishman, Baker, *Science* [86]	
70.	De novo design of a binding pair	Ankryn-repeat-based Tyr-Tyr binding between Prb and Pdar.	RosettaDesign, PatchDock, motif search, experimental affinity maturation	X-ray, ELISA, SPR, fluorescence polarization, DLS, CD, NMR	3q9n (2.0 Å), 3q9u (2.3 Å), 3qa9 (Prb only, 1.9 Å)	2011, Karanicolas, Baker, *Mol Cell* [87]	
71.	PPI: β-strand assembly (*βdimer1*)	Homodimer of γ-adaptin	Rosetta-based DDMI including five rounds of interface sequence design and minimization	X-ray, AUC, size-exclusion chromatography (SEC), binding-affinity	3zy7 (1.09 Å)	2011, Stranges, Kuhlman, *PNAS* [88]	
72.	Antigen design	HIV 4E10 epitope	Flexible backbone remodeling and resurfacing	CD, SPR, ELISA	–	2011, Correia, Schief *JMB* [89]	

73.	Thermostable terpene synthase	Thermostable tobacco 5-epi-aristolochene synthase	SCADS, 12 mutations which were all >12 Å from the active site	CD, GC-MS activity assay	–	2011, Diaz, Weiss, *Prot Sci* [90]	
74.	Collagen heterotrimer	Collagen A:B:C-type heterotrimer	Positive and negative design for Pro and hydroxyl-Pro inclusion in triplets and energy gap between model and competing structures. MC/SA	CD	–	2011, Xu, Nanda, *JACS* [91]	
75.	Enzyme-design by single mutation (*AlleyCat*)	Kemp-eliminase	Single-site scanning mutagenesis, docking, and super-rotamer modeling of transition state	NMR, CD, enzyme kinetics with pH-profile, CD,	2kz2 (NMR)	2011, Korendovych, DeGrado, *PNAS* [92]	
76.	Enzyme-design by few mutations	Kemp-eliminase	Mutation modeling and docking	CD, X-ray, enzyme kinetics	4e97 (1.3 Å), 4ekp (1.64 Å), 4ekq (1.54 Å), 4ekr (1.49 Å), 4eks (1.64 Å)	2012, Merski, Shoichet, *PNAS* [93]	
77.	Solubilization of the TM domain of nicotinic acetyl-choline receptor	TM domain of nicotinic acetyl-choline receptor α1 subunit	SCADS and loop modeling via MODELLER. Overall 21 residues in the TM region and loops were redesigned	LC/MS, CD, NMR	2lkg (NMR, major conformation), 2lkh (NMR, minor conformation)	2012, Cui, Xu, *BBA* [94]	
78.	Redesign of a mononuclear Zn metalloenzyme for organo-phosphate hydrolysis	De novo metalloenzyme design for the hydrolysis of the R_P isomer of a coumarinyl analog of the nerve agent cyclosarin	PDB scaffold search, RosettaMatch, RosettaDesign	X-ray, absorption spectra kinetic measurements	3t1g (2.35 Å)	2012, Khare, Baker, *Nat Chem Bio* [95]	

(continued)

Table 1
(continued)

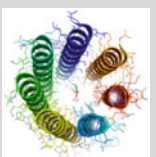
	Novelty	Target	Computational methods	Main characterization methods	PDB (Å resolution)	Year, first and last authors, journal, and references	Protein structure (first PDB only)
79.	High-affinity epitope scaffold	HIV 2F5 epitope design using backbone-grafting onto scaffold proteins	Six- and seven-residue epitope design via Rosetta, Epitope fragment search in the PDB, loop closure via cyclic coordinate descent (CCD) and MC	X-ray, CD, static light-scattering, SPR	3rij (2.3 Å), 3ri0 (2.25 2.8 Å), 3rhu (Å), 3rfn (1.8 Å)	2012, Azoitei, Schief, *JMB* [96]	
80.	Protein–protein orthogonal pair within a signaling circuitry	GTPase (Cdc42) redesigned to be activated by a designed GEF intersectin partner	RosetaBackrub, KIC, Rosetta	X-ray, CD, SPR, WASP fluorescence titration, GAP assay, ELISA, live-cell fluorescence microscopy, and XCELLigence Assay	3qbv (2.65 Å)	2012, Kapp, Kortemme, *PNAS* [97]	
81.	Ideal folds from secondary structure patterns	Five different-topology folds	New secondary structure connectivity rules, negative rules, and RosettaDesign (MC/SA) and RosettaHoles	NMR, CD, SEC-MALS	NMR structures: 2kl8, 2lv8, 2ln3, 2lvb, 2lta	2012, Koga, Baker, *Nature* [98]	
82.	Self-assembling, register-specific collagen hetero-trimers	Minimalistic de novo design of self-assembling, register-specific collagen hetero-trimers	Novel scoring function and genetic algorithm search procedure, focusing on axial inter-strand ionic interactions	CD, NMR	–	2002, Fallas, Hargernic, *Nat Comm* [99]	

83.	Enzyme design (*HG-1, HG-2, HG-3* and directed evolution HG-3.17)	Kemp eliminase	QM, MD, iterative approach	CD, MS, X-ray, enzyme kinetics	3O2L (2.0 Å), 3NYD (1.23 Å), 3NYZ (1.51 Å), 3NZ1 (1.56 Å), 4bs0 (1.09 Å)	2012, Privett, Mayo *PNAS* [100] Directed evolution to HG-3.17 by Blomberg, Hilvert *Nature* [101]	
84.	Iron-sulfur [4Fe-4S] cluster protein	Four-helix Coiled coil iron-sulfur protein (*CCIS*) monomer	ProtCad with metal-binding first approach	CD, EPR, SEC	–	2012, Gryb, Noy, *Biochim Biophys Act* [102]	
85.	Crystal structure space group de novo design	P6-space group crystal design from a three-helix 26-residues coiled-coil	Construction of a 'grid-like' ensemble of coiled coil P6-structures followed by side-chain design	X-ray, CD	4dac (2.1 Å), 3v86 (2.91 Å)	2012, Lanci, Saven, *PNAS* [103]	
86.	Altered functionality of a de novo design	Altering the oxidation of hydroquinones to hydroxylation of arylamines in a designed four-helix bundle	MSL	NMR, CD	2lfd (NMR)	2012, Reig, DeGrado, *Nature Chem* [104]	
87.	Conformational switch (*UniRapR*)	Rapamycin-regulated single chain of FKB12 and FKB12-Rapamycin binding protein	Medusa, replica exchange and discrete molecular dynamics	Immunoprecipitation, live-cell and live zebrafish imaging	–	2013, Dagliyan, Dokholyan, *PNAS* [105]	
88.	Binder of a hyperpolarizable abiological chromophore (*SCRPZ-1, SCRPZ-2, SCRPZ-3*)	Four helix bundle binding a ruthenium-zinc abiological hyperpolarizable chromophore	SCADS, loop grafting, MD	CD, Pump-probe absorption electronic spectroscopy, Hyper-Raayleigh light scattering, X-ray reflectivity	–	2013, Fry, Therien, *JACS* [106]	

(continued)

Table 1
(continued)

	Novelty	Target	Computational methods	Main characterization methods	PDB (Å resolution)	Year, first and last authors, journal, and references	Protein structure (first PDB only)
89.	GPCR solubilization and application as a biosensor	μ-opioid receptor	SCADS, comparative modeling, solubilized by mutating 53 membrane-exposed residues	CD, MS, homogeneous tim-resolved fluorescence (HTRF)	–	2013, Perez-Aguilar, Liu, *PLOS one* [107], (2014 biosensor application [108])	
90.	De novo lysozyme protein inhibitor	Hen egg lysozyme hot-spot centric active site protein binder	Target-surface scaffold selection, RosettaDock, RosettaDesign + directed evolution	X-ray, SEC, ^{19}F NMR, SEC, HEL activity assay, Yeast surface display	3vb8 (2.9 Å)	2013, Procko, Baker, *JMB* [109]	
91.	De novo high-affinity and selective ligand-binding protein	Selective, high-affinity binder of the steroid digoxigenin	Ligand conformer library, RosettaMatch, RosettaDesign, PDB scaffold scan, CCP4 (shape complementarity)	X-ray, SEC, CD, AUC, fluorescence polarization, isothermal titration calorimetry (ITC), yeast surface display	4j8t (2.05 Å), 4j9a (3.2 Å)	2013, Tinberg, Baker, *Nature* [110]	
92.	Hot-spot centric de novo pH-sensitive IgG binding protein	Fc-domain His-433 pH-sensitive IgG binding protein	RosettaMatch, RosettaDesign, PDB scaffold scan	CD, ELISA, directed evolution	–	2014, Strauch, Baker, *PNAS* [111]	
93.	Statistical energy function (SEF) boosted by experimental selection for foldability for automated CPD	Novel SEF exemplified by de novo CPD of a helical and mixed topology protein	New SEF with selection of structure neighbors with adaptive criteria (SSNAC) to address the multidimensional properties jointly, MC/SA minimization	CD, NMR, antibiotic resistance	2mlb (NMR), 2mn4 (NMR)	2014, Xiong, Liu, *Nat Comm* [112]	

94.	Helical bundles	Helical bundles including an antiparallel, untwisted three-helix bundle with 80-residue helices, an antiparallel right-handed four-helix bundle and a parallel left-handed five-helix bundle	Parametric backbone generation and Rosetta	CD, X-ray, EM	4uot (five-helix bundle, 1.69 Å), 4tql (three-helix bundle, 2.8 Å), 4uos (four-helix bundle, 1.63 Å)	2014, Huang, Baker, *Science* [113]	
95.	Water-soluble α-helical barrels	Pentameric, hexameric and heptameric Water-soluble α-helical barrels	CCBuilder, SOCKET, PoreWalker, coiled-coil assembly rules	X-ray, CD, AUC	4pna (2.0 Å), 4pn8 (2.0 Å), 4pn9 (2.0 Å), 4pnb (2.0 Å), 4pnd (2.0 Å)	2014, Thomson, Woolfson, *Science* [114]	
96.	Synthetic coiled-coils	Antiparallel homodimeric coiled-coils with no cross-specificity	DFIRE, CCCP structure generator, CLASSY	AUC, disulfide-exchange, CD	–	2014, Negrod, Keating, *JACS* [115]	
97.	Epitope-focused vaccine design	RSV helix-turn-helix epitope in a three-helix bundle scaffold	Fold-from-loops (FFL) protocol in Rosetta	X-ray, NMR-HSQC, CD, SPR, ELISA (on immunized macaques)	4L8I (2.0 Å), 4JLR (2.71 Å), 4N9G (2.5 Å)	2014, Correia, Schief, *Nature* [116]	
98.	TM transporter (*Rocker*)	De novo TM Zn^{2+} transporting four-helix-bundle	MaDCaT, Ez, MD, Volocidy	X-ray, ssNMR (^{1}H, ^{13}C cross-polarization MAS, ^{19}F-CODEX), AUC, Liposome flux	4p6j (2.8 Å),4p6k (2.7 Å), 4p6l (2.8 Å), 2muz (NMR)	2014, Joh, DeGrado *Science* [117]	

(continued)

Table 1
(continued)

	Novelty	Target	Computational methods	Main characterization methods	PDB (Å resolution)	Year, first and last authors, journal, and references	Protein structure (first PDB only)
99.	De-immunization	T-cell epitope removal from different proteins	T-cell epitope identification by SVM, redesign by Rosetta	Flow cytometry, fluorescence	–	2014, King, Baker, *PNAS* [118]	
100.	Self-assembling symmetry	β-propeller with perfect six-bladed symmetry	Visual template datamining, RosettaDock, phylogenetic ancestral reconstruction, circular permutation for the 'Velcro' strap	X-ray, CD, AUC, MS, Differential Scanning Fluorimetry	3ww7 (1.7 Å), 3ww8 (1.4 Å), 3ww9 (1.33 Å), 3wwa (1.99 Å), 3wwb (1.7 Å), 3wwf (1.6 Å)	2014, Voet, Tame, *PNAS* [119]	
101.	Repeat protein	Leucine-rich repeat protein from ribonuclease inhibitor family	Rosetta, new code for adjusting repeat geometry	CD, FTIR, NMR HSQC, AUC	–	2014, Ramisch, Andre, *PNAS* [120]	

calmodulin–peptide interaction focusing on electrostatic potential surfaces and structural modeling. These included side-chain positioning using geometries taken from a known homologous structure of an intestinal calcium-binding protein, interactive computer graphics, and minimization using the AMBER [124] force-field. The acquired know-how of the calmodulin-peptide structure and binding characterization was tested by iterative peptide synthesis and characterization. Hence, this early attempt of CPD underscores the need to integrate all available know-how and methods for the requested target as well as the need to combine theory and experiment in an interactive and iterative feedback loop.

In *1990* Hecht and Ogden and the Jane and David Richardson lab designed a de novo four-helix bundle, termed *Felix* [10]. This is an example in which protein design rather than CPD was the leading method. Even for designing the hydrophobic core, the authors write that: "*Space-filling models of Felix were constructed and the sequence was then modified to remove lumps or fill holes. This is easier to do with physical models than on the computer.*" Computationally, several structures were modeled followed by application of molecular dynamics (MD). Positive- and negative-design rules were conducted manually, including for residues preferring helicity, for the radial distribution of hydrophobicity along the helices and for helix capping. Hence, this case-study proves that it is required not only to focus on the requested design combining existing and newly found parameterization, but rather attention should be devoted to the so-called negative design of avoiding unwanted designs.

In *1991* Hellinga and coworkers used CPD software aimed at sites with predefined geometry (DEZYMER [125]). They introduced a copper-binding site into thioredoxin by mutating four amino acids [11]. In the analysis of the design they concluded that two residues are pivotal for the metal ligation while the two other are pivotal for removing alternative modes of binding, thus highlighting the need to focus on negative design.

In *1991* Wilson, Mace and Agard presented a generalized model for altering substrate specificity [12]. Using a $\Delta\Delta G$ free energy perturbation approach, the free energy of the free substrate, free enzyme, and complex were computed separately as to non-bonded and solvation energetics over the different potential conformations suggested by the PROPAK [126] rotamer-library based CPD software. The approach was tested using a protease in which the specificity for cleaving leucine was raised by three orders of magnitude following a single mutation. While this CPD example entails merely a single mutation, the components of the approach include many of the later CPD methodology.

In *1992* Hurley and Matthews redesigned the core of bacteriophage T4 lysozyme [13]. This case-study, coming from the lab most known for thoroughly studying the effect of mutation on

protein structure and function, includes several insights. Only nine solvent inaccessible amino acids were subjected to redesign. Moreover, a core valine residue was not part of the redesign as it binds structural water. The repacking was limited to residues that are more hydrophobic compared to the *wild-type* residues. In addition, as all potential sites occur in α-helical regions, no net increase in the number of β-branched amino acids (Val and Ile) was allowed. While each addition of a β-branched amino acid to a helix has a small energetic cost of less than 0.5 kcal/mol, it was feared that the accumulation of such residues will destabilize the structure. For packing calculations, the Ponder and Richards rotamer library [126] was used truncating rare (<5 %) rotamer conformations. Hydrogens were omitted and reduced van der Waals radius was applied to account for local relaxation. The free energy was calculated with a standard local minimization as well as a component accounting for the loss of side-chain conformational entropy. Four amino-acids were mutated showing a similar stability compared to the template structure (0.5 kcal/mol destabilization). The destabilization of each single mutation was much larger thus showing the overall cooperative nature of the overall core repacking design.

In *1994* Jane and David Richardson, de novo designed *beta-doublet*, a β-sandwich protein [14]. It is no surprise that such an endeavor came from pioneers in visualization (Richardson diagram, also known as ribbon diagram), parameterization, and quality control of protein structures. A four-stranded β-sheet dimer designed from scratch included an intersubunit disulfide bridge. Internal side chains were chosen for their statistical preference for β-sheet formation and their ability to tightly pack in a protein core. This knowledge-based parameterization was corroborated by side-chain repacking of rotamers. This design scheme focused on negative design, specifically disfavoring the Greek Key topology. To minimize alternative folding modes, turns were shortened as much as possible. Binding of 1-anilinonaphthalene-8-sulfonate (ANS) was higher, compared to binding to well-folded proteins. Along with low unfolding cooperativity and poor NMR characteristics, this may indicate a loosely packed hydrophobic core or even a molten-globule structure; highlighting the challenge of obtaining thermostable de novo designed proteins, let alone those composed of β-sheets.

3 The Second Decade of CPD, 1995–2004

Setting the framework for CPD, in *1995* DeGrado and coworkers reviewed the hierarchic approach to protein design including helix stabilization, coiled coils, four helix bundles, β-sheets, mixed α-β structures, DNA-binding proteins, and functional proteins [127].

The presented approach emphasized the need for quantitative parameterization of the various levels of structure and function within the design target. Such parameterization can be either physics-based or knowledge-based. In either ways, it should be integrated into quantitative potential (scoring) functions.

In *1995* Desjarlais and Handel presented a novel computational framework for the de novo design of hydrophobic cores [15]. The CPD was conducted via the Repacking of Core (ROC) program, later developed to their Sequence Prediction Algorithm (SPA) [128]. The approach included two steps—a custom-made rotamer library for hydrophobic residues (Val, Ile, Leu, Phe, and Trp) and a genetic algorithm (GA) for optimizing sequence and structure space of the designed protein. The method was exemplified on the phage 434 Cro helical protein with five to eight amino acid changes in the hydrophobic core. Two of the three attempted designs resulted in a stable protein. This first study into a pivotal protein region helped to substantiate the notion that the noncore residues of a protein play a role in determining the uniqueness of the folded structure [15].

In *1997* Desjarlais and Handel applied their ROC program for the stabilization of a mainly β-sheet protein, ubiquitin [16]. Nine designs with three to eight mutations each were experimentally characterized. Unlike their 434 Cro [15] redesign, all ubiquitin designs were less stable relative to the *wild-type* protein. The authors postulate that this may be due to the fact that in contrast to the α-helical 434 Cro protein, ubiquitin is mainly composed of β-strand secondary structures which may dictate more stringent packing requirements. One of the designs was structurally elucidated confirming that the core side-chains had less favorable conformations and higher flexibility compared to the *wild-type* [17].

In *1997* Dahiyat and Mayo opened the field of full-protein fully automated computational de novo protein design [18, 19]. The CPD scheme was termed ORBIT [18] for Optimization of Rotamers by Iterative Techniques. The so called full sequence design 1 (*FSD-1*) was not a typical protein of over 200 amino acids, but rather a small, 28-residue sequence; a length considered a peptide rather than a protein. Nevertheless, the remarkable achievement included a complex ββα motif based on the polypeptide structure of a zinc finger domain in which 20 of the 28 residues were subjected to design. Moreover, while such a small DNA-binding motif is folded in nature with the aid of a zinc ion, the zinc-ligating residues (two cysteines and two histidines) were replaced in the design with two phenylalanines, an alanine, and a lysine without the need for the metal ion. As a side-remark, the use of a charged lysine in such a core position highlights the need to take caution in stigmatizing amino acids as "hydrophobic" or "hydrophilic" as in this case the long hydrophobic neck of this charged residue filled the hydrophobic requirement within this position. The 1.9×10^{27} possible

amino acid sequences were searched by application of the Dead End Elimination (DEE) theorem [129]; highlighting the intertwined connected between CPD and search and sampling methods [130]. FSD-1 displayed low identity to any other existing sequence, thus establishing it as a 'de novo' design. In this fixed-backbone design, an existing crystal-structure template was utilized in which eight residues were left as is and the remaining 20 were subjected to design. The hierarchical approach of confining key positions was further confined by considering 7, 10, and 16 optional amino acids for each core, surface, and boundary position, respectively. The backbone dihedral angle further confined two positions to glycine, thus de facto leaving 18 positions for CPD. The combined structure space defined by the accessible backbone-dependent Dunbrack rotamer library [131] applied over the accessible fold space, resulted in 1.1×10^{62} possible rotamer sequences. The experimental validation included Nucleic Magnetic Resonance (NMR) structural elucidation exhibiting 1.98 Å and 0.98 Å Cα-atom root means square deviation (RMSD) between the design and the template structure for the full and the core residues (residues 8 to 26), respectively. The difference between these two numbers highlights the intrinsic flexibility and disorder associated with nonsecondary structure elements, especially when positioned at the edge of the protein sequence.

In *1998* the Mayo lab applied the ORBIT [18] for the design of a hyperthermophilic Streptococcal protein G β1 domain [20]. The stability enhancement stemmed from seven mutations which optimized core packing, increased burial of hydrophobic surface area, more favorable helix dipole interactions, and improvement of secondary structure propensity. The resulting protein displayed a melting temperature above 100 °C and a 4.3 kcal/mol thermodynamic stabilization compared to the *wild-type* at 50 °C. Structure, activity, and binding to an antibody were similar to the *wild-type* structure thus changing only the thermal stability of the protein.

In *1998* the Kim lab designed right-handed coiled coils applying backbone flexibility, hydrophobic-polar residue patterning for the superhelical axis and the hydrophobic core along with modeling of packing [21]. Backbone coordinates were determined by exploring a parametric family of superhelical backbones described originally by Francis Crick. Negative design was applied by mimicking a less-folded state via permutations on the mutation location and calculating the energy gaps to such permutations. Dimeric, trimeric, and tetrameric bundles were designed. The tetramer was structurally resolved exhibiting a striking 0.2 Å RMSD for the core residues.

In *1998* the DeGrado lab de novo designed an antiparallel three-helix bundle, *α3C*, in an iterative process with specific interactions added incrementally [22]. In this design many steps were designed rationally without the aid of the computer. Two rounds of

core design were conducted fully by CPD. A previously designed dimer (CoilSer) that was found to be a trimer was the initial template for the design. In this structure, some hydrophobic Leu residues adopt a less likely rotamer suggesting the availability of better core packing. The trimer was trimmed by one turn. In the first round, GlyAsn and ProGlyAsn loops were added to turn the discrete helices into a single subunit. In the second round, helix capping was introduced and in the third round nonnative characteristics were eliminated by negative design. Specifically, the 17 residues of the hydrophobic core were repacked using 30 runs of ROC followed by 30 runs of ROC for a subset of six residues. Further, to avoid both clockwise and counterclockwise turning of the helices within the trimer, charged residues were designed to cause electrostatic repulsion and favor only one conformation. This is a direct negative design step. Thus, the designed helix capping interactions and electrostatic interactions between partially exposed residues assisted in achieving a unique, native-like structure. In *1999*, three surface exposed residues were changed thus designing *α3D* in which the homology between the helices was decreased thus simplifying structural elucidation [23].

In *1999* the Serrano lab redesigned the two-helix coiled-coil interleukin-4 using GCN-4 as a template [24]. This is not a classical CPD case-study but rather a computer-aided sequential rational design where deep understanding of the binding interface enabled grafting of the positive electrostatic convex binding site shape from the four-helix-bundle protein to a new two-helix template. The side-chains of the mutated positions were structurally predicted via the rotamer-library-based software SMD [132]. Interestingly, MD simulations were applied as in silico screening of the mutations prior to decision on experimental characterization. Depending on the size of the interleukin-4 binding site (to interleukin-4 receptor alpha) grafted on the GCN4 template, the binding affinities ranged from 2 mM to 5 μM.

In *2001* the Baker lab applied CPD to convert the monomeric protein L to an obligate dimer by just three mutations [25]. The design relied on a β-hairpin single mutation domain swapped dimer in which a β-turn straightens and the C-terminal strand inserts into the β-sheet of the partner. The Rosetta [133] module RosettaDesign [134] focused on an eight-residue region and added just two mutations to the domain swapping mutation resulting in an obligate dimer.

In *2001* the Serrano lab applied PERLA [135] for the redesign of their previously designed 20-residue β-sheet protein *betanova* [136] aiming to create a set of double- and triple-mutations with different folding stabilities so as to compare predicted and experimental folding stabilities [26]. Briefly, PERLA includes a custom-made rotamer library, an all-atom force-field, and a combination of statistical terms including solvation and entropy. Relaxation of the

local strains is achieved by sub-rotamer states and most parameters are balanced with respected to a reference denatured state. DEE is applied to prune the search space and then side-chain conformations are weighted using a mean-field approach. Here, two CPD schemes were applied: First, four positions adjacent to aromatic residues were discretely redesigned aiming at utilizing the Nuclear Overhauser effects (NOEs) between the aromatic residues and the new mutations for evaluating structural effects. Second, multiple-residue mutations were designed with the most promising designed experimentally characterized. Increase in core hydrophobicity or van der Waals contacts stabilized the design. At one site the algorithm did not predict a hairpin destabilization, possibly due to alternative conformations. Alternatively, the sequence of folding events should be taken into account along with the balance between long-range electrostatic interactions and short-range van der Waals interactions. β-sheet propensities were also shown to correlate with stabilization. Some of the mutants stabilized the design by 1 Kcal/mol. Taken together; this early study displays the usage of CPD algorithms for the study of structure–stability relationships and parameterization of their underlying causes.

In *2001* Bolon and Mayo applied ORBIT [18] to computationally design protozymes which are enzyme-like proteins exemplified on a thioredoxin scaffold catalyzing a nucleophilic hydrolysis of *p*-nitrophenol acetate [27]. ORBIT applies a force-field and DEE theorem to compute sequences that are optimal for a given scaffold. The use of an inert scaffold required the design of a new cleft, which was relatively open to the surrounding milieu, thus possibly affecting efficiency. The 94 non-glycine positions reflected 10^{101} rotamer sequences that were scanned using the DEE algorithm within ORBIT [18]. The rate enhancement of ~25-fold ($K_M = 170 \pm 20$ μM, $k_{cat} = 4.6 \pm 0.2 \times 10^{-4}$ s^{-1}) is comparable to that of early catalytic antibodies (Table 2).

In *2001* the Kim lab designed six dimeric coiled coils with a range of stabilities by combining knowledge-based rules (specifically the *a* and *d* hydrophobic positions in the heptad repeat), rotamer selection and sampling followed by minimization [28]. The first two parts address the large accessible search space while the last one assists in achieving quantitative estimates of interaction energies. For example, a hydrophobic Val was constrained to the gauche (−) rotamer, which is known to be favored in this position. In parallel to choosing a small subset of rotamers, subrotamers were introduced by including +/110° of the χ_1 and χ_2 rotamer positions. Interestingly, to address the difficulty of modeling solvent-exposed charged residues, residues at the *e* and *g* positions of the heptad repeat were truncated beyond the C_δ position. Minimization was carried out without electrostatics but with an explicit hydrogen-bonding term and the overall solvent-exposed residue energetics were later fixed by an empirical solvation correction.

Table 2
CPD approaches towards engineering improved Kemp eliminases. The kinetic data displayed is modified after [100]. An array of protein templates and design schemes were applied towards a common goal, often involving feedback from previous research on this new enzyme target. Likewise, the scope of the CPD ranged from a single mutation to a QM-optimized catalytic site and full-protein template re-engineering

Approach	Template protein	Catalysis k_{cat}/K_M [M^{-1} min^{-1}]	Catalysis [$(k_{cat}/K_M)/k_{uncat}$]	pH optimum (catalytic residue)	References
Catalytic antibodies	Immunglobulin	5500	~5×10^9 (pH 7.5)	pH > 7.0 (Glu)	[137]
CPD iterative approach (QM, MD, + directed evolution)	Xylanase	430 (2.3×10^5)	4×10^8 (pH 7.25)	pH = 7.0 (Asp) 18	[100, 101]
CPD (Rosetta, KE70 + directed evolution)	Aldolase	126 (54,800)	1×10^8 (5×10^{10}, pH 7.25)	pH > 7.0 (His-Asp dyad)	[138]
CPD (Rosetta, KE07 + directed evolution)	Hisf protein	12.2 (2600)	1×10^7 (2×10^9, pH 7.25)	pH > 7.0 (Glu) 7,20	[72, 139]
CPD (Rosetta, KE59 + directed evolution)	Glycerophosphate synthase	~160 (60,430)	1×10^8 (5×10^{10}, pH 7.25)	pH > 8.0 (Glu) 7,21	[72, 140]
AlleyCat—CPD single-site mutation (super-rotamer minimization)	Calmodulin C-terminal domain	5.8	4×10^5 (pH 8.0)	pH > 8.0 (Glu) 22	[92]
CPD artificial cavity (most active construct L99A/M102H)	T4-lyzozyme	1.8	7×10^7 (pH 5.0)	pH = 5.0 (His)	[93]

The propensity of residues to be in helices was also added to the equation. The designed structures displayed an impressive <0.7 Å for all non-hydrogen atoms.

In *2002* the DeGrado lab computationally designed an A_2B_2 four-helix bundle protein binding diiron called *DueFerro tetramer* or *DFtet* [29]. The de novo design focused on the gap between the requested fold and alternative folds thus explicitly incorporating positive- and negative-design considerations. The design was built using a template of a previous design which was then elongated to increase stability by extending the four-helix bundle Crick parameters. Residues were chosen to increase helical propensity, stabilize one of the competing topologies via computing contact energetics. The best four designs following 700,000 iterations of sequence design were modeled structurally and the best design was validated experimentally.

In *2002* the Serrano lab de novo designed 13 divergent spectrin SH3 core sequences to determine their folding properties [30]. The PERLA-based [135] redesign included nine nonconsecutive positions resulting in a larger buried hydrophobic volume. The computational design over-packed the core resulting in an expansion of the β-barrel. This was further validated by conducting Ile → Val mutations which all resulted in strain removal and stabilization. Eleven of the 13 designs folded well with similar characteristics to the folded *wild-type*. Two structurally resolved designs were similar to the *wild-type* with small changes at a loop region following discrepancies at the χ_2 side-chain positions relative to the design.

In *2002* Shifman and Mayo modulated calmodulin binding specificity by CPD [31]. The calmodulin binding interface was optimized to improve binding specificity towards one of its natural targets, smooth muscle myosin light chain kinase (*smMLCK*). ORBIT [18] considered 10^{22} sequences to optimize the calmodulin–smMLCK interface. Thus, without considering negative design explicitly, a design with eight mutations enabled similar binding affinity to the target and 1.5- to 86-fold decreased affinity to six other targets. In *2003* a follow-up included optimization of the CPD for PPI [32]. First, the pairwise portion of the energy function was weighted to enhance intermolecular interactions and attenuate intramolecular ones. Second, the large dielectric constant (ε) routinely used, effectively underemphasized the long-range electrostatics term in the energy function relative to more local terms such as van der Waals and hydrogen bonding interactions. Consequently, the dielectric constant at the boundary- and surface-optimization region was lowered from $40r$ to $4r$. Third, a romater library that contained rotamers representing expansion about the χ_1 and χ_2 angles was applied. Six designs were tested on eight targets of which the best showed a specificity change of 0.9- to 155-fold. Hence, by optimizing the protein– protein binding, the

natural promiscuous binding was decreased. Yet, without direct incorporation of negative design, this decrease displayed large variation among the alternative targets.

In *2002* Xencor applied the Protein Design Automation CPD software (PDA [141]) and demonstrated it by redesigning 19 residues in the vicinity of β-lactamase's active site to confer resistance against antibiotic cefotexime [33]. The PDA defines a library of mutant sequences at specific positions. After finding the global minimum energy conformation (GMEC) an MC/SA search algorithm is applied to find near-optimal sequences which are then processed to generate a probability table of mutations at each designed position. The CPD reduced the large sequence space to a library of ~200,000 sequences which were experimentally screened obtaining variants exhibiting a 1280-fold increase in cefotaxime resistance along with a 40-fold decrease in ampicillin resistance.

In *2002* Xencor applied CPD to stabilize solubility and improve thermosstability of the human growth hormone (hGH) [34] and to stabilize the granulocyte-colony stimulation factor (G-CSF) [35]. In both cases, only core residues were redesigned. As the CPD scheme of the two targets was similar, they are described here together. In both cases, the DEE-based PDA CPD scheme was applied. Interestingly, new terms for side-chain and backbone entropies were added to the scoring function as a combined measurable reflecting the loss of conformational entropy during core packing of the designed core residues. Other scoring function components such as polar hydrogen burial, dielectric constant, and surface-based nonpolar exposure penalty were weighted into a new scoring function. The 45 core residues were redesigned resulting in 11 mutations. Three designs were tested experimentally achieving thermostabilization of 13–16 °C without compromising biological activity. Similarly, the G-CSF was redesigned to improve pharmacological properties for the prevention of chemotherapy-related neutropenia [35]. Here, a homology model based on the bovine structure was used as a template with 25–34 core residues redesigned with PDA. Several mutants with 10–14 mutations were experimentally characterized. Without compromising biological activity, a thermostabilization of 13 °C and a tenfold improvement in shelf-life was obtained.

In *2002* the Baker, Monnat and Stoddard labs designed an artificial endonuclease by fusing the N-terminal domain of homing endonuclease I-Dmol to an I-Crel monomer, creating a new 1400 $Å^2$ interface between the domains [36]. The design, termed *E-Drel*, for engineered I-Dmol/I-Crel, was initially modeled by superimposing a single helix from the N-terminal domain of I-Dmol on the same helix in I-Crel and linking the two domains using a three-residue linker –NGN- which encourages β-turn formation. All interface positions were redesigned using RosettaDesign

[134]. The relative contribution of side chains to the interface free energy were evaluated by computational alanine scanning [142]. The CPD focused on six residues exhibiting steric clashes in the original model and extended to eight additional residues predicted to contribute substantially to the interface free energy. One thousand separate designs were conducted over two backbone models eliminating results that may affect the active site and reducing redundant results. The 16 top-scoring designs, each with 8–12 interface mutations were screened in vivo. The resulting structurally- and functionally characterized E-Drel enzyme bound the DNA target site with nanomolar affinity and cleaves it at precisely the same rate as the *wild-type* enzyme.

In *2003* the Wodak lab conducted automatic design of major histocompatibility complex class I (MHC-I) 9-residue binding peptides which impair CD8+ T-cell recognition [37]. While this is a 9-amino acid peptide design rather than a protein design, it is presented here as an early example of computationally designing peptide–protein interactions. DESIGNER [5, 6], which combines a fitness function with an optimization procedure selecting highly scoring sequences. To select amino acid sequences with lowest free energies, a DEE procedure was applied as well as a heuristic procedure with 250,000 iterations. In an early ensemble-like approach, DESIGNER was run on all six representative MHC-peptide complexes available in the PDB. In addition, the top-scoring peptides were scanned against peptides known to bind the same MHC allele. The six strongest binders not only bound MHC but also formed stable complexes and three displayed significant inhibition of CD8+ T-cell recognition.

In *2003* the Saven and DeGrado labs designed a water-soluble analog of the pentameric phospholamban membrane protein [38]. Solubilization enables to study the protein, including ligand or drug interaction, in the much friendlier soluble milieu. Here, 11 solvent-exposed residues were identified in the transmembrane (TM) helix. Ten residues were redesigned using a pairwise potential including intrahelical pairwise residue interactions, contribution to the helix macrodipole, interhelical electrostatic interactions, solubility, and sequence entropy. The water-soluble analog mimicked all the TM protein characteristics including oligomerization state, helical structure, and stabilization upon phosphorylation. A truncated version of the helix bundle was resolved crystallographically [39] displaying a parallel tetramer, rather than an antiparallel pentamer; suggesting that buried and interfacial hydrogen bonds are pivotal for oligomerization.

In *2003* Havernek and Harbury approached molecular recognition by entwining positive- and negative-design using a multi-state framework for engineering specificity in GCN4-based coiled-coils [40]. Their approach selects sequences maximizing the transfer free energy of a protein from a target conformation to a set of undesired

competitor conformations. The algorithm identified three specificity motifs that have not been observed in naturally occurring coiled coils. Their genetic algorithm (GA) considered four states including homodimer, heterodimer, aggregated-, and unfolded-state which focus on homospecificity, solubility, and stability. Unlike previous CPD approaches, they selected sequences that maximize the transfer free energy from a target state to an ensemble of competitors, thus requiring separate structural optimization for each state. Further, they evaluated prediction by molecular mechanics with continuum solvent allowing for direct prediction of observed free energies. Seven of the eight engineered pairs showed $\Delta G_{\text{specificity}}$ values exceeding the largest control value that was obtained fortuitously.

In *2003* the Saven and DeGrado labs designed a de novo monomeric helical dinuclear metalloprotein [41]. The 114-residue four-helix-bundle due ferro single-chain (DF_{SC}) was modeled in the backbone level using previous oligomeric structures and inter-helical turns. While 26 residues were predetermined including ligand-binding residues and one of the turns, all other 88 residues were computationally designed using the Statistical Computationally Assisted Design Strategy (SCADS [143]). The fixed positions relied on previous designs of due ferro peptide ensembles [47, 144]. The software provides site-specific amino acid probabilities, which are then used to guide sequence design. This successful design was the first realization of complete de novo design, where backbone structure, activity, and sequence are specified in the design process. Several years later, the structure was solved combining NMR and unrestrained MD using nonbonded force-field for the metal shell, followed by quantum mechanical/ molecular mechanical dynamics used to relax the NMR-apparent local frustration at the metal-binding site [42].

In *2003* Kuhlman, Dantas and coworkers at the Baker lab presented a milestone in CPD—the first systematic de novo CPD of a 93-residue α/β novel topology protein, which folded in atomic-level accuracy (1.2 Å RMSD) to the design template [43]. The so called *TOP7* protein includes four β-strands flanked by two α-helices. The loops connecting the secondary structure elements are very short thus contributing to the atomic-level accuracy of the design. The starting models for the design were assembled from three- and nine-residue fragments via the Rosetta package [133]. 172 backbone-only models were generated, forming an ensemble of structures that all fit the requested fold. The sequences were generated using RosettaDesign [134] via a Monte Carlo (MC) search protocol focusing on van der Waals and hydrogen-bonding interactions within an implicit solvent. An additional reduction of search complexity was attained by restricting the β-strand positions to polar residues. With the Dunbrack rotamers [145] considered for each position, the procedure included $>10^{186}$ rotamer

combinations. A simultaneous optimization of sequence and structure was conducted by using the Rosetta approach for backbone optimization with each starting structure followed by 15 cycles of sequence design and backbone optimization.

In *2003* the Desjarlais lab de novo designed a WW domain using fully automated CPD emphasizing backbone flexibility [44]. Here, the labs' SPA [128] CPD software was coupled to a sampling procedure integrating information from an ensemble of backbone structures, thus setting the stage to multistate CPD. The new procedure was termed SPANS for sequence prediction algorithm for numerous states. The ensemble was generated by a simple MC expansion of $\pm 5^{\circ}$ perturbation of the backbone Φ and Ψ angles till a predetermined (0.3 Å) RMSD. Three antiparallel strands fold into a β-sheet WW domain. The 34–40 amino acid WW domain folds autonomously with two-state kinetics and is utilized as a module to bind proline-containing regions. Two CPD approaches were used, each with methods applied in many other applications. First, *SPANS-WW1* applied multiple "sub-rotamer" states which were sampled stochastically. The Boltzmann weights of these states were combined into one "super-rotamer" and included in the partition function. Alternatively, *SPANS-WW2* optimized each canonical rotamter by torsion-space steepest-descent minimization. Both designs exhibited WW domain biophysical characteristics yet with decreased stability relative to the template, especially for *SPANS-WW1* which included a less-dispersed hydrogen-bond network.

In *2003* the Baker lab applied RosettaDesign for the redesign of nine different globular folds achieving, on average 65 % deviation in sequence space with biochemical characteristics comparable with their natural templates [45]. One of these designs, human procarboxypeptidase A2, was structurally resolved in *2007* enabling to discretely analyze residues contributing to different types of hydrophobic packing: interhelical, inter-strand, and helix-strand packing [46]. While the original redesign had numerous mutations and 10 kcal/mol increased stability, relative to the *wild-type*, mutating merely four residues yielded a 5 kcal/mol stability increase.

In *2004* Kaplan and DeGrado designed a phenol-oxidase from first principles [48] using a computationally designed four-helix-bundle scaffold made out of four peptides of two kinds (A_2B_2) that assemble in a noncovalent manner [29]. Specifically, positions 15 and 19 were mutated to small amino acids thus sculpting the diiron binding pocket to bind the 4-aminophenol substrate. The resulting quinone monoamine product was produced with a $k_{cat}/K_M = 1500\ M^{-1}\ min^{-1}$ with efficiency sensitive to the size of the binding pocket, thus reporting on design specificity. Herein, although the three-dimensional structure of the backbone and sequence of the de novo designed scaffold protein was designed computationally,

the subsequent introduction of catalytic activity was accomplished without methods or by screening large number of variants.

In *2004* the Baker lab redesigned specificity of a protein–protein interaction between a bacterial nonspecific DNase (colicin E7) and its tightly bound inhibitor protein (immunity protein Im7) pairs [49]. The structurally resolved binding pairs offer straightforward activity assays and the computational design focused on destabilizing interactions with the *wild-type* partner while stabilizing the mutant complex. Interface positions on both binding partners were mutated and assessed as to their binding free energies and specificity changes between cognate and noncognate binding partners. Three positions were chosen for redesign in the DNase and nine in the inhibitor. The designed cognate pairs displayed low affinity relative to the *wild-type* pair, presumably due to a new water network, which was not part of the modeling. This suggsts focusing on explicit modeling of bound water in interface design. Nevertheless, the redesigned interface was structurally resolved displaying 0.62 Å RMSD between the model and the actual structure. Focusing on the hydrogen bond network and water therein, a *2006* follow-up study sampled alternate rigid body orientations to optimize the interface interactions and then utilized the resolved structure to further optimize the hydrogen bonding network, thus increasing the specificity difference between cognate to noncognate complexes by 300-fold [50].

In *2004* the DeGrado and Saven labs applied CPD to design a water-soluble analog of the potassium channel KcsA [51]. Using SCADS [143] and the previous solubilization application [38], 35 solvent-exposed residues were identified and subjected to mutation. The first round of the water-soluble K-channel (Denoted *WSK-1*) displayed high oligomers and thus additional mutations were applied on two solvent-exposed hydrophobic patches. The resulting *WSK-3* structure mimics the TM structure in secondary structure, tetrameric quaternary structure, and tight binding of a toxin and a channel blocker.

4 The Current Decade of CPD, 2005–2014: From Enzymes to Membrane Proteins

In *2005* the Stoddard and Baker labs conducted thermostabilization of the homodimeric hydrolase enzyme yeast cytosine deaminase (yCD), which converts cytosine to uracil [52]. Only three mutations enabled an increase of 10 °C in the melting temperature. All residues that were more than 4 Å from the active site and were not involved in the dimer interface were subjected to CPD. Half of the 65 residues were left unchanged following the redesign and half of the remaining suggested mutations were solvent exposed. The remaining suggested mutations were experimentally characterized individually suggesting a triple mutant as the most thermostable one.

In *2005* Sauer and coworkers compared positive- and negative-design strategies for reengineering a homodimer into a heterodimer [53]. Using the Stringent Starvation Protein B (*SspB*) α/β-fold homodimer as a model system, stability-focused (positive design) using the DEE search algorithm as implemented in ORBIT [18] and specificity-focused (negative design) were applied aiming to reengineer the homodimer into a heterodimer. While the positive design yielded a more stable heterodimer, only the incorporation of negative design yielded exclusive heterodimerization. Eight interface residues (four from each subunit) were subjected to design allowing for ten out of the 20 amino acids in each position. The authors note that the greatest challenge was modeling the energetic effects of destabilizing mutations in competing state. This challenge was approached by capping unfavorable van der Waals energies as an approximation for conformational relaxation that would alleviate atomic overlaps. Notably, in *2007* the Mayo lab used ORBIT [18] to design 13 and 11 residues on two monomer variants of streptococcal protein G—β1 domain (*Gβ1*) that were designed to heterodimerize [60]. Of the 24 positions, 15 "core" positions were restricted to seven hydrophobic residues and the rest to polar and charged residues. Applying such hydrophobic patches serves as negative designs destabilizing the monomer state. This specific design was successful in shifting a monomer to a dimer, albeit with a low binding constant. Overall, these studies showed the challenges of PPI design along with the importance of negative design, even at the expense of stability.

In *2005* the DeGrado, Saven and Dutton lab de novo designed a 40-residue redox-active minimal rubredoxin mimic [54]. This is one of the first b-sheet CPD, let alone with the rubredoxin tetrahedral metal-binding motif. The last three strands of the *Pyrococcus furiosus* rubredoxin were transformed using a twofold symmetric axis containing the metal ion. A hairpin motif (tryptophan zipper) was used to fuse the two sides. Other than the hairpin motif, active-site Cys, two Gly and an Ile residue, all amino acids were designed using SCADS [143]. The apoprotein and holoproteins were stable with 16 Fe(II/III) functional cycles under aerobic conditions.

In *2005* the DeGrado lab applied CPD for a de novo four-helix bundle protein that selectively binds two nonbiological cofactors termed *DPP-Fe* for 5, 15-Di[(4-carboxymethylene-oxy)phenyl] porphinato iron(III)-chloride [55]. Herein, the apoprotein folds upon binding the cofactors. The four-helix bundle was designed to maintain 17–19 Å between the metals, His-Fe coordinative interactions, second shell hydrogen-bonding, minimal steric clashes and D_2 symmetry with sampling via MC/SA. Then, three rounds of SCADS [143] sequence calculations were applied to 28 residues.

In *2006* Dmochowski, Saven, and coworkers designed ferritin-like proteins (*Dps*) with increasingly hydrophobic cavities [56]. The

resilience of the self-assembling complex to mutation which intuitively should denature the protein is striking. As many as 120 hydrophilic residues were mutated to hydrophobic or small amino-acids. The Dps complex is a 12-subunit iron warehouse in which each subunit is a four-helix-bundle with two helices facing the interior large iron-binding cavity. The SCADS [143] software extended for symmetric homo-oligomeric quaternary structures [146] was applied forming Dps3, Dps7, and Dps10, each with three, seven, and ten mutations in each of the dozen subunits. Not only was the mutation per se taken into account but also how much each residue is prone to an acceptable mutation. Amino acids participating in salt bridges within the hydrophobic core were not subjected to mutagenesis. The mutations increased the percent of hydrophobic surface within the iron-binding cavity from 52 % to 86 %. The high melting temperature of the complex as well as iron-mineralization function were largely unchanged for Dps3 and Dps3 and even Dps10 folded and assembled properly. Taken together, this study questions the importance of the hydrophilic surface for proper folding of proteins, let alone protein complexes; thus opening the door for CPD of hydrophobic surface regions.

In *2006* Quax, Serrano, and coworkers designed tumor necrosis factor-related apoptosis-inducing ligand (TRAIL) variants which initiated apoptosis exclusively via the DR5 receptor [57]. The DR5-selective TRAIL variants represent a reduced binding promiscuity CPD approach which in this case potentially permits tumor-selective therapies. The CPD scheme was straightforward including protein modeling via WHATIF followed by refinement via FOLD-X. Residues binding to nonconserved positions in the different four potential receptors were mutated via FOLD-X to all other amino acids obtaining 2720 models for the 34 designed sites. The binding energy of the models was used to assess selectivity yielding seven single-site variants for experimental validation.

In *2006* the Baker lab redesigned a cleavage specificity of the intron-encoded homing endonuclease I-MsoI [58]. The CPD aimed at changing one base pair in each recognition half site. The CPD approach used as input the *wild-type* crystallographic structure and considered (in turn) all symmetric base pair changes. New side chains next to these base pairs were attempted listing the predicted discrimination energy between the previous and new recognition sites. The modeling of the DNA-protein interface is challenging not only due to the highly charged electrostatic environment possibly requiring bound water molecules, but also as the binding may involve conformational changes in both binding constituents. The redesigned enzyme cleaves the new recognition site ~10,000 more effectively compared to the *wild-type* protein.

In *2006* Xencor Inc. designed antibody Fc variants with enhanced Fcγ-receptor-mediated effector function [59]. A

combination of "directed" diversity and "quality" diversity strategies were applied within the CPD scheme of optimizing the IgG Fc region for Fcγ-receptor affinity and specificity. Four positions were mutated in different combinations. Where structural information was available, substitutions that provide favorable interactions were designed, and where such information was incomplete, calculations provided a quality set of variants enriched for stability and solubility. At some positions, only residues with high propensity to the core, surface and boundary of the protein were allowed, thus focusing the search space sampled. The designed variants displayed over 2 orders of magnitude enhancement of in vitro effector function, enabled efficacy against cells with low levels of target antigens and resulted in increased cytotoxicity *in vivo*.

In 2007 the DeGrado lab designed a TM peptide that specifically targets a membrane protein [61]. The peptide was named *CHAMP* for Computed Helical Anti Membrane-Protein Peptide. The TM helices of the $\alpha_{IIb}\beta_3$ and $\alpha_v\beta_3$ integrins were the subject of the design by replacing the β_3 subunit with a new designed helix. The two subunits form a parallel GAS_{Right} motif [147] which was structurally modeled with the β_3 subunit was redesigned. Five and 15 template backbones were tested for the design of the *CHAMP* against the α_{IIb} and α_v helices, respectively. In the inner half of the membrane only eight residues were considered. Repacking of proximal positions was accomplished with a linearly dampened Lennard-Jones potential with van der Waals radii scaled to 90 %, as implemented in PROTCAD [29] and a membrane-depth dependent knowledge-based potential. 10,000 iterations of an MC with simulated annealing (MC/SA) were applied for the sequence and rotamer space search and sampling, with the rotamers optimized using DEE followed by exhaustive enumeration. The new designs were tested in micelles, bacterial membranes, and mammalian cells.

In *2007* the Kuhlman lab focused on high-resolution design of a protein loop [62]. Within the Rosetta software package a loop design protocol was developed. The protocol iterates between optimizing the sequence and conformation of a loop in search of low-energy sequence–structure pairs. 10-residue loops were designed for connecting the 2nd and 3rd strand of β-sandwich protein tenascin-C. Loop templates were datamined from 142 12-residue loops found in the protein databank (PDB) that superimpose the backbone atoms of the design target. These backbone templates were redesigned with many undergoing four to five mutations. Loops were filtered by searching for solvent accessible surface area to a 0.5 Å radii probe and by searching for unsatisfied hydrogen bonds. Two of three experimentally tested loop designs were solved showing similar structures compared to the design while a third design appeared in a significantly different structure; thus highlighting the potential for loop design along with the unique challenge in designing loops.

In *2007* Lai and coworkers de novo designed a protein that binds the erythropoietin receptor [63]. The CPD was based on grafting discontinuous interaction epitopes. The erythropoietin (EPO)—EPO-receptor complex structure was studied; identifying three key residues in EPO which were searched in the PDB - yielding 1756 potential scaffold proteins onto which the keystone residues were grafted. These were filtered for RMSD, shape-complementarity, packing density, and high buried accessible surface area yielding 15 potential scaffolds for further analysis. A fourth mutation was designed to eliminate a steric clash. The novel triple mutant, composed of an unrelated protein, rat PLCδ1-PH (pleckstrin homology domain of phospholipase C-δ 1) bound the EPO receptor with a K_D of 24 nM in vitro and gave an IC_{50} of 5.7 μM in a cell-based assay.

In *2007* the Mayo lab redesigned a 51-residue homeodomain aiming at thermostability [64]. Different sequence optimization algorithms were compared of which two were characterized. Amino acids were divided into buried and solvent-exposed, and further restricted at helix-capping sites. MC/SA yielded the best solution. The successful design had a thermal denaturation midpoint temperature of >99 °C.

In *2007* the DeGrado, Saven and Roder labs applied CPD for the de novo design of a single-chain asymmetric diphenylporphyrin four-helix bundle metalloprotein [65]. An MC/SA protocol was applied given five constraints: (a) a metal-metal distance of 17–19 Å, (b) optimal His to Fe bonding interactions, (c) second-shell His-Thr hydrogen bonding, (d) minimal steric clashes, and (e) D_2-symmetry. A previous four-chain design [55] was shortened by four residues at each end and replaced by loops. A new program, STITCH, identified loops within a nonredundant PDB set that superimposed well on five amino acids at the helical ends. Iterative cycles of SCADS [143] CPD chose the sequence for 100 of the 108 amino-acids, with eight keystone His and Thr residues fixed as part of the cofactor ligation. The experimentally characterized single-chain design demonstrated higher stability compared to the four-chain previous design both apo- and holo-forms with the latter increasing stability significantly.

In *2007* the Tidor computational lab and the Wittrup experimental lab joined forces to apply CPD for the improvement of antibody affinity [66]. The iterative CPD cycle focused on electrostatic binding contributions and single mutations. By combining multiple designed mutations, a tenfold and 140-fold affinity improvement was engineered to an anti-epidermal growth factor antibody and to an anti-lysozyme antibody, respectively. Interestingly, this study began by a general CPD approach that was in general not successful and led to the understanding that for antibody designs the calculated electrostatic term (using Poisson-Boltzmann continuum electrostatics calculations) for binding was

a better predictor for affinity improvement compared to the total calculated binding free energy. Thus, a full side-chain conformational search was maintained but only the electrostatic component was applied for affinity improvement.

In *2008* the Schreiber and Edelman-Sobolev labs redesigned a protein–protein interface between TEM1 β-lactamase and its inhibitor β-lacatamase inhibitor protein (BLIP) for high-affinity and binding specificity using a novel method [67]. Their novel PDBmodDesign method included replacing structural interface modules with fragments taken from nonrelated proteins and ranking the 10^7 starting templates with an accurate atom–atom contact surface scoring function. The resulting high affinity and specificity affirms their modularity approach.

In *2008* the Dmochowski, Saven and Christianson labs joined forces to design a human H ferritin protein that will bind noble metal ions Au^{3+} and Ag^{+}, reduce the ions and form nanoparticles within the protein's cavity [68]. The study followed up on the ferritin-like protein hydrophobic cavity design [56] and applied a similar CPD methodology. Here, 192 mutations were designed in the 24-subunit complex including four external- and four internal-surface mutations for each subunit. Two His and two Cys on the external surface were mutated to charged, polar, or small residues. In parallel, three Glu and a Lys on the internal surface were all mutated to Cys as an ion-binder residue. Combining positive- and negative-design this was aimed to promote noble metal ion binding in the cavity while avoiding such binding on the outside surface as well as minimizing protein aggregation. Following experimental difficulties of crystallization with gold ions, Hg^{2+} was used to probe the metal–thiol interactions. Probably due to decrease in aggregation, the outer-surface mutations stabilized the protein. Strikingly, the internal-surface mutations kept this high stability and exhibited Ag^0 and Au^0 nanoparticles upon soaking with their respective ions. Indeed, the crystal structure proved the CPD structure and requested function.

In *2008* Handel and coworkers redesigned BLIP to increase affinity to SHV-1 which unlike TEM (presented in the previous example), displays micromolar affinity, thus providing space for affinity improvement [69]. The EGAD design software succeeded to stabilize the interface by 10- to 1000-fold. The experimental structures generally agreed with the computational designs, except for salt-bridges. Additionally, the authors claim that the off-rotamer conformational sampling could be improved by adding a short minimization following the DEE rotamer search.

In *2008* the Saven, Therien, Blasie and DeGrado labs from the University of Pennsylvania designed nanostructured metalloporphirin arrays from coiled coils [70]. Following a previous design of a D_2-symmetric α-helical coiled coil (34 residues for each helix) that binds two nonbiological porphyrin cofactors [55], the four-helical

coiled-coil was extended by three-heptad repeats, enabling the binding of four iron porphyrins. Three charge patterning mutations were introduced to enforce an antiparallel orientation and two additional mutations were introduced to improve electrostatic interactions with the cofactor carboxylates. The resulting four-porphyrin complex was experimentally characterized. The modular addition of heptad repeats between the helical capping sections demonstrates the robustness of the coiled-coil structure, as defined by the Crick parameters. This design introduces the feasibility of engineering electrically and optically responsive multiporphyrin arrays.

In *2008* the Baker lab presented two computational enzyme designs—a group of retro-aldolases [71] and a Kemp eliminase [72], the latter with Tawfik. Both designs applied a similar scheme for enzyme design without cofactors [148]. These computational enzyme designs followed an algorithm presented in *2006*, which was successful in targeting ten different enzymes and identifying the native site in the native scaffold and ranking it within the top five designs for six of the ten reactions [149].

The retro-aldolase CPD strategy is described over 12 pages in the supplementary material of the publication highlighting the many aspects that must be addressed [71]. These range from the quantum-mechanical (QM) structural description of the catalytic sites to the computational and experimental ranking and validation of the designs. Briefly, composite active-site descriptions of transition states were applied to generate candidate catalytic sites via RosettaMatch [150] which fills a hash-table with catalytic amino-acid rotamers for the proposed catalytic site constraints. The remaining positions are redesigned to optimize the transition-state binding affinity using RosettaDesign [134]. Following structural refinement, the potential designs are ranked based on the total binding energy to the composite transition state as well as satisfaction of specific catalytic geometry. Designs were filtered if the van de Waals energetics was too high (>-5 kcal/mol), the binding pocket was too buried or was not sufficiently accessible. This CPD scheme resulted in 72 designs of which 32 displayed retro-aldolase activity of up to 4 orders of magnitude kinetic acceleration.

The *2008* Kemp eliminase CPD by the labs of Baker and Tawfik [72] achieved a 10^5 rate enhancement. In vitro evolution further enhanced k_{cat}/K_M by >200-fold. The CPD scheme was similar to the one of for the retro-aldolase. The successful designs showed high shape-complementarity with several polar or charged catalytic residues: out of 59 designs, 39 used Asp or Glu as a general base while 20 used His-Asp or His-Glu as a catalytic dyad. Such variation highlights the robustness of the CPD strategy which, in this case, exhibits variability in the functionally accessible set of catalytic residues. π-stacking interactions contributed towards stabilizing the transition state. The collaboration between the CPD approach

provided by the Baker lab and the directed evolution approach provided by the Tawfik lab continued with subsequent directed evolution efforts conducted by Khersonsky et al. [138–140]. Cumulatively, the latter efforts showed that CPD designs are highly evolvable and can be optimized for catalytic efficiency, reduced thermodynamic stability (which is often too high in computational designs), optimization of the catalytic site microenvironment for the required transition state preorganization, and the presentation of key changes that provide feedback for deciphering mechanism and further CPD efforts. While directed evolution is not the focus of this chapter, the collaboration highlights the need to embed within the CPD approach other fields in a multiple dimension feedback approach. Fortunately for the CPD field, this Kemp eliminase computational design sparked an array of follow-up research of which some is highlighted below [92, 93, 100, 151] with the key kinetic parameters summarized in Table 2.

In *2009* the Baker lab focused on loop remodeling to alter enzyme specificity [73]. Following benchmark tests on eight native protein-ligand complexes, a critical loop in guanine deaminase was redesigned such that it became 100-fold more active on ammelide and 25,000–fold less active on guanine. The two to five residue loop modeling succeeded in altering specificity. Nevertheless, it should be noted that the absolute activity towards the new substrate ($k_{cat}/K_M = 0.15\ s^{-1}\ M^{-1}$) is still 7 orders of magnitude lower than the activity of the *wild-type* enzyme towards its innate substrate; highlighting the comprehensive evolution of enzymes towards their functionality, which is likely to include far more than one loop.

In *2009* the Shifman lab applied CPD for increasing the binding specificity of calmodulin 900-folds [74]. Relying on the promiscuous binding of calmodulin to both CaM-dependent protein kinase II (CaMKII) and calcineurin (CaN), calmodulin was optimized to bind the former. The ORBIT-based [18] CPD emphasized intermolecular interactions and showed that the specificity increase was largely due to a decrease in binding to CaN.

In *2009* the Keating lab applied a computational framework for desigin of protein-interaction specificity allowing for CPD of selective basic-region leucine zipper (bZIP) binding peptides [75]. The 20 bZIP transcription factor family share high sequence similarity challenging specificity design. As shown by protein arrays, the CPD succeeded in designing selectivity by optimizing the affinity and specificity trade-off e.g. by sacrificing the stability score and by introducing negative design to disfavor complexes with undesired bZIP competitors. The bZIP microarray assay benefits from reversible folding of short coiled coils, and data from previous array measurements of many bZIP transcription factor pairs were critical for developing predictive models. Their CPD framework is denoted CLASSY for cluster expansion and linear programming-based

analysis of specificity and stability [75]. The CLASSY multi-state CPD applies integer linear programming followed by cluster expansion in which a structure-based interaction model is converted into a quick-to-evaluate sequence-based scoring function. Negative design is integrated by applying CLASSY to the design-target and to design-off-target states.

In *2009* the Baker lab conducted CPD on the monomeric homing endonuclease I-AniI which cleaves at the center of a 20-base-pair DNA target site [76]. The pseudo-symmetrical enzyme's N- and C-terminal domains bind to the left (−) and right (+) DNA target sites in very different manners as reflected by causes of CPD-based altered specificity: specificity on the (−) side was achieved by modulating single-turnover conditions (K_M) while that in the (+) side was achieved by modulating turnover number (k_{cat}). The Rosetta-based CPD scheme tailored for DNA–protein interactions relied on their previous study [58]. Loop rebuilding was used to model backbone shifts. In a feedback loop, the best designs were reverted position by position to the *wild-type* sequence to identify mutations that did not contribute significantly to the energy or specificity. Multi-state design [40] to assess the specificity offset between the altered and *wild-type* DNA target structure. Further, a genetic algorithm was applied to evolve sequence for preference of the target state compared to competitor states.

In *2009* the Donald lab conducted computational structure-based redesign of the phenylalanine adenylation domain of the nonribosomal peptide synthetase enzyme gramicidin S synthetase A (GrsA-PheA) for a set of noncognate substrates for which the *wild-type* enzyme has little or virtually no specificity [77]. Here the aim was increased specificity with the leading design exhibiting 1/6 of the enzyme/*wild-type* substrate activity. The *K** algorithm [152] was applied on the active site, a generally considered optimized region which is not the classical target for most CPD attempts. The double mutant selected showed a 19-fold increase of k_{cat}/K_m for the new Leu substrate and a 27-fold decrease of this measurable for the *wild-type* Phe substrate. On top of two active-site mutations, so called "bolstering" mutations were designed outside the active site aiming to stabilize the -active-site mutant. Indeed, such mutations gave an additional twofold increase in k_{cat}/K_m for the Leu substrate. Similarly, further designs for charged substrates were also successful experimentally.

In *2010* the DeGrado, Saven and Therien labs applied CPD for the design of an A_2B_2 four-helix bundle that selectively binds two emissive abiological (porphinato)zinc chromophores of DPP-Zn [78]. The positive and negative ligand-directed CPD is selective and did not bind related chromophores such as DPP-Fe^{3+}. To achieve the selective Zn-cofactor binding, a pentacoordinate environment with one His ligand was designed, yielding C_2 symmetry. One peptide chain included a His ligand while the other

included a Thr ligand; thus applying a negative design element that allows only the heterotetramer to bind the chromophore. SCADS [143] was applied for the recursive design of 62 variable positions. Cys (potentially making disulfide bridges), His (potentially ligand binding) and Pro (potential helix-breaker) were excluded at all positions, Met at interior positions. Three sequential rounds of sequence CPD were applied and the resulting design was validated experimentally.

In *2010* the Baker lab altered the cleavage specificity of the I-Msol homing endonuclease for three contiguous base pair substitutions [79]. Using a CPD scheme previously applied to the protein [58, 76], concerted design for all simultaneous substitutions was more successful than a modular approach against individual substitutions, highlighting the importance of context-dependent redesign and optimization of protein–DNA interactions. In a CPD and structure determination feedback loop, a structure of the CPD effort and its associated unanticipated shifts in DNA conformation was utilized to create an endonuclease that specifically cleaves a site with four contiguous base pair substitutions.

In *2010* the Mayo lab changed the emission wavelength of red fluorescent protein by CPD [80]. Herein, CPD was combined with small experimental combinatorial libraries of mCherry mutants. The library design procedure takes as input a list of scored sequences, and two sets of constraints: a list of allowed sets of amino acids, and a range of desired library sizes. The algorithm generates a list of the combinatorial libraries that satisfy these constraints, and then ranks the libraries by the degree to which they reflect the energetic preferences present in the list of scored sequences. Thus, CPD was used to perform an in silico prescreen to eliminate sequences incompatible with the protein fold and generate combinatorial libraries amenable to rapid experimental screening. The successful 20–26 nm red-shifted mutants found included targeted stabilization of the excited state via H-bonding and π-stacking interactions as well as destabilization of the ground state via hydrophobic packing. Overall, 13 residues were involved in the design.

In *2010* Warshel suggested that the current computational enzyme design approaches reflect incomplete understanding of the details of the enzymatic system and/or inaccurate modeling by the CPD algorithm [151]. Using his empirical valence bond (EVB) simulations of the Baker and Tawfik Kemp eliminase [72], his group showed that the attempt to predict the proper transition state stabilization and related overall preorganization effect are not likely to be achieved by gas phase models. Warshel showed that the transition state design displays a charge distribution that makes it hard to exploit the active site polarity, even with the ability to quantify the effect of different mutations. Further, the directed

evolution led to reduction of the solvation of the reactant state rather than to the expected transition-state stabilization applied by naturally evolved enzymes. This study highlights the need to carefully design the preorganized environment such that it will exploit the small changes in charge distribution during the formation of the transition state.

In *2010* the DeGrado, Therien, Blasie and Walker labs de novo designed a TM diporphyrin-binding protein complex [81]. The design, termed *PRIME* (PoRphyrins In MEmbrane), positions two non-natural iron diphenylporphyrins (Fe^{3+} DPP's) sufficiently close to provide a multicentered pathway for TM electron transfer. Unlike previous TM to soluble solubilization efforts, here the opposite path was applied with a four helix D_2-symmetrical bundle adapted for the membrane milieu. First, keystone cofactor-binding residues (His and Thr) were designed within an idealized four-porphyrin binding soluble four-helix bundle backbone template [70]. Then, an all side-chain DEE followed by MC/Self-consistent mean field (SCMF) approach was applied to explore the reduced search space along with the Lazaridis implicit membrane solvation (IMM1). The 24 positions were divided to four categories (buried, mostly buried, mostly exposed and completely exposed). These were given different degrees of side-chain conformational sampling with conformations selected from a conformer library. Models were ranked by oligomerization energy, i.e. the difference between the energy of the complex and that of the monomeric state (a membrane solvated helical state, with relaxed side chain conformations), and the lowest energy model was extensively experimentally characterized validating the design.

In *2010* the Kuhlman lab redesigned the binding of hyperplastic discs protein to P21-activated kinase 1 kinase (PAK1) domain [82]. The Iterative Rosetta-based DDMI (Dock, Design, Minimize Interface) protocol was used for docking the scaffold on a chosen hotpsot. Next, loops of an MC-based sequence optimization and backbone optimization by minimization were conducted. This resulted with potential redesigned interfaces that were filtered by knowledge-based criteria including binding energy density and the number of unsatisfied polar interface residues. Of six experimentally characterized designs, two aggregated and the rest had binding affinities of up to 100 μM.

In *2010* the Mayo lab combined CPD with experimental library screening demonstrating the successful synergism of the two approaches for thermostabilization of core positions of *Gβ1*, the β1 domain of Streptococcal protein G [83]; a protein previously designed by the lab to dimerize [60]. The lab's previous Fast and Accurate Side-chain Topology and Energy Refinement (FASTER) CPD software for single-state design was expanded here for the multistate design case. The combination enables the application of multistate design methods to large conformational libraries,

transformation of semi-rational CPD results to combinatorial mutation libraries, and the experimental stability determination of the designed libraries. The novel protein library design method took into account the library size and possible sets of amino-acids to best reflect the experimental results. The library design procedure was called *CLEARSS* for Combinatorial Libraries Emphasizing And Reflecting Scored Sequences. Five experimental crystallographic and NMR structures were used, each resulting in a 24-member design library. The results enabled to characterize the sequence space available for the multistate design.

In *2010* the Anderson and Donald lab applied CPD for the prediction of drug resistance mutations in methicillin-resistant *Staphylococcus aureaus* (MRSA) dihydrofolate reductase (DHFR) [84]. Using ensemble-based CPD algorithm K* which includes DEE search followed by energy minimization [152], potential resistance mutations were predicted. The process incorporated positive design to maintain catalytic function and negative design to interfere with binding of a lead inhibitor. Interestingly, the *wild-type* sequence was ranked low for both the natural ligand and the inhibitor; suggesting that numerous sequences may have improved binding to these ligands. Four of the ten top-ranking designs were experimentally evaluated, of which three were shown to maintain activity while lowering binding affinity 9- to 18-fold for the inhibitor. The top-ranked double-mutant was crystallized; validating the design by showing reduced hydrophobic interactions in one locus and introducing a steric bulk in another.

In *2011* the DeGrado lab applied CPD to design virus-like protein assemblies on carbon nanotube surfaces [85]. The surface properties and symmetry were used to define the sequence and superstructure of the designed surface-organizing peptides. Single-walled carbon nanotubes were covered with virus-like coating converting the smooth surface into a highly textured assembly with long-scale order, thus capable of e.g. directing the assembly of gold nanoparticles into helical arrays along the nanotube axis. Three selection rules were applied for the design, defining the intrinsic recognition motif and its packing into higher-order assembly in accord with the long-range order of the underlying surface. First, a group compatible with the target surface was identified, in this case avoiding a hydrophobic motif and using small residues Gly or Ala. Second, intersubunit packing was defined in accordance with the surface symmetry. The cylindrical nanotube suggested rotational-screw symmetry in the form of coiled coils with a radius of ~9 Å defining five to seven subunits. Third, designability of the coiled coils was assessed by searching existing tertiary motifs. Four designs were tested, sequences based on an existing protein (domain swapped dimer) and a de novo coiled coil, each with Gly or Ala as the nanotube-facing residue. Adding gold particles to the

outer surface enabled transmission electron microscopy (TEM) validation.

In *2011* the Baker lab took the challenge of PPI and designed a protein that binds to the conserved stem surface of influenza hemagglutinin [86]. The strategy focused on the design of shape-complementarity with hot-spot-like residue interactions, with the latter serving as anchors to the former. 865 potential scaffold proteins were searched to support the disembodied hot-spot residues and the shape complementarity. The coarse-grain binding modes were then refined by docking followed by scaffold redesign. Selected designs included 51 and 37 designs with two and three hot-spot residues, respectively. Designs that presented binding were subjected to directed evolution for increased binding; resulting in mutations supporting interactions of filling a void in the binding interface, favorable interactions in the unbound state, electrostatic complementarity, and desolvation. Two binding proteins displayed nanomolar affinity.

In *2011* the Baker lab applied a motif-based method to computationally design protein–protein complexes with native-like interface composition and interaction density as exemplified on the Prb-Pdar heterodimer [87]. The tight dimer was further optimized by directed evolution which surprisingly rotated one of the complex partners by 180°, showing that the specificity of the binding patch was not sufficient yet the binding hot-spot was sufficient to facilitate the binding within a noncrowded pure protein environment. The motif-based approach focused on a key polar aromatic residue (Trp or Tyr) which facilitate packing and hydrogen bonding followed by shape-complementarity. Here, the ankryn repeat which naturally associates with an array of proteins served as one scaffold (redesigned to Pdar). Each of several ankryn repeat protein structures was paired with a set of 37 structurally diverse thermostable proteins applying a surface feature-matching approach, PatchDock [153], followed by rigid-body docking to generate a set of bound orientations with shape-complementarity. The interface design started from screening a well-packed hydrogen-bond containing aromatic pair followed by expanding it to include a hydrophobic first shell of residues and a polar secondary shell of residues protecting the hydrophobic patch from the solvent. RosettaDesign was used to optimize residue identities at the interface periphery holding the hydrophobic inner layer fixed. Further, global long-range electrostatic complementarity was aimed at by biasing one partner to acidic residues and the other to basic ones. Finally, natural parameterizations of native interfaces, e.g. size, packing, void volume, and lack of steric clashes were used to filter the suggested designs. Notably, negative design was not applied in any step, possibly facilitating the 180° flip of binding orientation in an experimentally validated pair. Twelve designed pairs were experimentally screened of which five displayed a signal >2-fold over nonspecific binding. Finally, a combination of phage and yeast display was applied

to evolve tighter binding of the leading pair. Two mutations introduced in this step improved binding from a K_d of 130 nM to 180 pM.

In *2011* the Kuhlman lab designed a symmetric homodimer using β-strand assembly in which two solvent-exposed strands were designed to form an antiparallel β-strand pairing [88]. Looking for solvent exposed β-strands, automatic homodimer docking (similar to the DDMI protocol) was applied with the β-strand part designed with five rounds of symmetric sequence optimization and minimization at the interface; searching for an >850 $Å^2$ buried interface and minimizing unsatisfied buried polar atoms. Of the 5500 structures scanned, 1100 had an exposed β-strand. One structure, γ-adaptin was chosen. Two mainly hydrophobic and two mainly polar interface homodimers were characterized of which the former were more successful emphasizing the difficulty in designing hydrogen-bond networks. One promising structure *βdimer1* was structurally resolved showing that the design was successful.

In *2011* William Schief and coworkers applied CPD with flexible backbone remodeling and resurfacing for designing antigens [89]. In this intriguing approach, an HIV 4E10 epitope structure was implanted onto a new scaffold enabling antigen optimization. The remodeling refers to replacing a backbone segment by a new design. The resurfacing refers to redesigning the antigen surface outside the target epitope to obtain variants that maintain only the epitope. Briefly, their six-stage protocol includes segment selection (length, secondary structure), de novo backbone CPD of the segment followed by sequence design and minimization. Next, designs that did not meet energy, packing, and unsatisfied polar-atoms were filtered and surface hydrophobic residues were replaced by polar ones. Three designs of 16–17 remodeled segment were experimentally characterized showing a viable epitope while maintaining solubility and binding affinity.

In *2011* Korendovych and DeGrado applied an alternative minimalist approach to the Kemp eliminase design challenge [92]. Rather than conducting a comprehensive design of a full protein from the QM-optimized active site to the rest of the enzyme, they applied a single mutation in a minimal 75-residue allosterically regulated catalyst, termed *AlleyCat* (for ALLostEricallY Controlled cATalsyt), with activity ($k_{cat}/K_M = 5.8 \pm 0.3\ M^{-1}\ s^{-1}$) comparable to the original [72] Kemp eliminase design. The rationale was that protein folding energetics can dehydrate a carboxylate side-chain rendering it from the weakly basic aqueous state to a strongly basic dehydrated state. The computational design scheme applied on calmodulin C-terminal domain included in silico single-site Asp or Glu mutagenesis scanning of the C-terminal domain cavity, which naturally binds aromatic side-chains, suggesting that it can bind the benzisoxazole substrate. Low energy models including the point mutation which facilitated a cavity were next docked to the substrate. This determined whether the C-H hydrogen would be

appropriately positioned in the Michaelis complex. Finally, the Glu carboxylate was virtually fused to the substrate and the resulting "superrotamer" was optimized. Alternative mutations were used as control.

In *2011* the Weiss and Saven labs applied SCADS [143] to design a thermostable terpene synthase, an enzyme involved in the synthesis of antibiotics, flavorings, and fragrances [90]. A dozen mutations were selected for design in the tobacco 5-epi-arisolochne synthase (TEAS) for the catalysis of carbocation cyclization. All mutations were >12 Å from the substrate binding site so as to minimize an effect on the functional site. Amino acids identities were prepatterned at the mutated sites based on the number of Cβ atoms within 8 Å of the amino acids: for residues with 0–6 Cβ atoms were constrained to charged, polar, and small residues. For those with 7–8 Cβ atoms, aliphatic and aromatic residues were added to the potential mutations enabling mutation to all residues except Cys, Pro, His, and Thr. Last but not least, buried residues with 10 or more Cβ atoms were allowed to mutate to eight relatively hydrophobic residues. Mutations included both buried and surface-exposed positions with the latter eliminating surface-exposed hydrophobic patches and introducing salt bridges. The design retained activity in 65 °C and denatured in 80 °C, which is twice the temperature relative to the *wild-type*.

In *2011* the Nanda lab computationally designed an A:B:C-type heterotrimer collagen [91]. They applied positive and negative design constraints. A compositional constraint was used where all triplets in the design contained Pro or hydroxy-Pro. The energy score was constrained to allow the melting temperature to be above 26 °C. Specificity was enforced by optimizing the energy gap between the design and the best competing stoichiometry. The resulting empirical design displayed two of the nine available stoichiometries (B:2C and 2B:C). The ABC design indicated multiple species (due to permutations) which were removed upon increasing the salt concentration to 100 mM.

In *2012* several labs from the University of Pennsylvania and University of Pittsburgh applied Saven's SCAD CPD software to produce a water-soluble TM domain (α1 subunit) of the nicotinic acetylcholine receptor [94]. The template used for the CPD was a 4-Å low-resolution cryo-electron microscope (EM) structure in which hydrophobic residues with >40 % exposure to the membrane region were redesigned using a molecular mechanics force field entwined with an energy function that constrained the average hydrophobicity of surface-exposed residues to that expected for an average soluble protein of a similar size. In order to avoid spectral over-crowding in NMR spectra used to solve the structure, residues which were not highly favorable in a given site underwent an additional round of CPD with an additional constraint imposed so as to increase sequence diversity. In addition, a polyglycine linker

was designed between the C-terminus of helix-4 and the N-terminus of helix-1 using the loop builder in MODELLER [154]. The design was structurally resolved by NMR displaying high resemblance to the TM domain of the bacterial pentameric ligand-gated ion channel (GLIC); demonstrating the robustness and general applicability of the CPD scheme. Two conformations were resolved with overall dynamics that may be due to the dynamic loops. Moreover, anesthetics were bound to the same residue as in the bacterial GLIC validating the functionality of the solubilized protein.

In *2012* Baker and coworkers redesigned a mononuclear zinc adenosine deaminase metalloenzyme for organophospate hydrolysis of the R_P isomer of a coumarinyl analog of the nerve agent, cyclosarin [95]. First, a set of mononuclear zinc enzyme scaffolds with at least one open coordinate state was extracted from the PDB. The open coordinate state was utilized to ensure that structural zinc is excluded from the set. Previous gas-phase quantum-mechanical calculations of organophosphate hydrolysis were used to construct models of the reaction transition state bond lengths and angles. RosettaMatch [150] was used to search for hydrogen-bonding interactions to the phosphoryl oxygen, the nucleophilic hydroxyl moiety, and the leaving group oxygen. Next, RosettaDesign was used for shape-complementarity interactions to the transition state. These parameters along with the presence of a docking funnel timed the results to 12 potential proteins, of which a redesigned adenosine deaminase hydrolyzed the substrate 7-hydroxycoumarinyl phosphate (DECP). The eight-mutation design exhibited activity that was sevenfold higher than that of the buffer background. Directed evolution at eight positions increased activity kinetics to levels identical to the *wild-type* deaminase with over 140 catalytic turnovers per enzyme and high stereospecificity. The directed evolution improvement of k_{cat} was *post factum* realized as an increase in the basicity of an active site Glu residue.

In *2012* the Schief lab followed up on their previous epitope grafting research [89, 155] and applied CPD with Rosetta to design a new 2F5 HIV epitope with improved biophysical characteristics followed by transplanting the linear epitope onto different scaffolds [96]. Here, the epitope design used side-chain grafting while backbone-grafting was applied to transplant the design onto the new scaffold. Potential scaffolds were identified by searching the PDB for the core Asp-Lys-Trp sequence of the epitope. Side-chain grafting was conducted by binding interface optimization followed by sequence design for epitope accommodation and removal of extraneous interfacial interactions. The latter was facilitated also by initially changing the identity of all non-interacting scaffold residues to glycines. During the automated CPD, residues

within 4 Å of the epitope were allowed to change to any non-cysteine residue while other residues were allowed to change to small residues Gly, Ala, Ser, or Thr. For the backbone grafting, both N-terminal to C-terminal and C-terminal to N-terminal were considered with a 3 Å-RMSD threshold of the epitope to the scaffold set as an initial filter followed by a steric-clash filter. Loop closure utilized a Rosetta low-resolution scoring function, cyclic coordinate descent (CCD [156]) and MC sampling. Next, a high-resolution scoring function was applied to catch problematic conformations. Finally, a full-atom refinement was applied. For two of the three cases tested experimentally, binding to the antibody was increased 9- and 30-fold compared to side-chain grafting alone.

In *2012* Merski and Shoichet applied an alternative minimalist approach by engineering a Met102 → His mutation to the Leu99 → Ala cavity in T4 lysozyme [93]. Here, CPD was applied to engineer subsequent mutations that increased activity fourfold to $k_{cat}/K_M = 1.8\ M^{-1}\ min^{-1}$. The absence of ordered water or hydrogen bonds and the presence of a common catalytic histidine base in complexes of the enzyme with product analogs facilitated detailed analysis of the reaction mechanism and its optimization. Notably, in this design some of the stabilizing mutations followed previous studies on the T4 lysozyme showing that deep knowledge-based understanding of the template, whether theoretical or experimental, is key to the design efforts. In this iterative approach the first designs had low stability of $\Delta\Delta G = \sim -7$ kcal/mol relative to *wild-type* T4 lysozyme while subsequent designs increased stability to $\Delta\Delta G = \sim -2$ kcal/mol with a significant increase in catalytic activity.

In *2012* the Kortemme lab applied CPD to control protein signaling by designing a GTPase/guanine nucleotide exchange factor (GEF) orthogonal (non-cross-reacting) pair [97]. A new interaction was designed while maintaining correct interface with existing machinery. Integrating such a new protein pair into existing cellular circuitry requires consideration of certain design criteria: Not only must the redesigned GTPase be activated by its redesigned GEF partner, but it must also be protected from inadvertent activation by the *wild-type* GEF and all other endogenous GEFs. Further, the redesigned GTPase must also preserve interactions with both upstream regulators and downstream effectors. Here, the known interface between the GTPase Cdc42 and ITSN (GEF) was used as a template for the new design. Computational alanine scanning was used followed by backbone design using the computational second-site suppressor protocol [49]. These simulations identified substitutions in one protein that are significantly destabilizing to the complex formed with the *wild-type* partner but can be compensated for by complementary changes in the partner. Flexible backbone CPD used RosettaBackrub [157] and the robotics-inspired local loop reconstruction method for peptide

chains, called kinematic closure (KIC) [158]. One hundred resulting models were used as a backbone ensemble for interface redesign using one interaction pair as an anchor followed by backbone diversification. Then, soft and hard repulsive forces were applied iteratively aiming at modeling conformational changes that initially appear unfavorable but may be accommodated by subsequent refinement. The experimentally validated design was proven structurally and functionally. The interaction is activated exclusively by the engineered cognate partner while maintaining ability to interface with other GTPase signaling components in vitro. The orthogonality was also shown in mammalian cells.

In *2012* the Montelione and Baker labs applied new rules for designing ideal protein structures applying CPD for the design of five different folds [98]. Secondary structure connectivity rules were derived from simulation and from datamining available structures. For connecting two β-strands, 2- and 3-residue loops prefer L-hairpins while 5-residue loops give rise to R-hairpins. For connecting a β-strand to a α-helix, a parallel orientation is preferred for 2-residue loops while an antiparallel one is preferred for 5-residue loops. For the reverse connectivity (αβ), the general preference is for parallel connectivity, especially for short 2-residue loops and longer loops providing helix-capping. Similar rules were applied for connecting three secondary structures. Negative design was applied for local interactions and for the edge of β-strands, the protein surface and high core packing. Five new folds were designed, almost all with short 2- and 3-residue loops, 7-residue β-strands, and 18-residue α-helices. Ab initio simulations of 200,000–400,000 structure predictions were performed to map the folding energy landscape, selecting 10 % with well-funneled landscapes. Five folds were experimentally determined displaying 1.1–2.0 Å RMSD as compared to their respective designs.

In *2012* Fallas and Hartgerink applied CPD for the design of self-assembling, register-specific collagen heterotrimers focusing on sequence-specific axial salt-bridges [99]. A collagen composed of three distinct chains can trimerize in 27 unique combinations. Axial rather than later contacts, stabilize the heterotrimeric collagen target state. The energy score includes a component for the difference between the number of ionizable residues and the number of salt-bridges which was searched using a genetic algorithm. An automated sequence selection algorithm was successful as it balances between destabilization induced on triple helical assemblies by changing conformationally restricted imino acids (Pro) to ionizable residues and the stabilization conferred on the formation of axial interstrand ionic interactions. For each mutation, the gap between the target state and competing states was computed for all 27 states. Experimental validation showed that this minimalist function is sufficient, though could be optimized with the addition of components such as electrostatic repulsion and specific local energetic contributions.

In *2012* the Mayo lab published an interesting story of applying an iterative stepwise approach to computational enzyme design of Kemp eliminases termed *HG-1*, *HG-2,* and *HG-3* [100]. The paper highlights the evolution of the CPD process with increasing success following careful analysis of the result in the previous round, an approach named *the protein design cycle* [141]. The motivation for this study followed on the study of Warshel [151] showing that the Kemp eliminase design of Baker and Tawfik [72] was not an ideal enzyme and required a "shotgun" approach of selection, not to mention benefiting from *in vitro* evolution. Interestingly, for the case of HG-3, 17 rounds of directed evolution produced an enzyme which accelerated the reaction by 6×10^8-fold, thus approaching natural enzyme rates [101]. The directed evolution optimized substrate-enzyme shape-complementarity, substrate-catalytic base (Asp127) alignment and, above all, stabilization of a negative charge in the transition state which emerged over the course of the evolution, reminiscent of the serine-protease oxanion hole.

In *2012* four labs from four countries (Grzyb, Nanda, Lubitz, and Noy) joined forces to compare computational and empirical design of iron-sulfur cluster proteins [102]. Both approaches successfully yielded a cluster-binding helical bundle. The CPD of a several coiled coil iron-sulfur clusters (*CCIS*) aimed at increasing stability of the reduced state of the [4Fe-4S] cluster by improving packing, helix propensity, oligomerization prevention (by changing surface net charge), and charge pairing optimization. Each of these aims was tested in a different design. Structural modeling was conducted by multiple-threading alignment within I-Tasser [159], and CPD was conducted using ProtCad [160] using the metal-first approach [161]. All CCIS designs were helical. The design focusing on stabilizing the iron-sulfur cluster increased helicity upon binding the cluster, showing the success of the design within a marginally stable protein. In this case, attempts to improve the CPD by intuitive modifications had limited success as to improved stability of the [4Fe-4S] stability over redox cycling suggesting that a different backbone scaffold should be attempted.

In *2012* the Saven and DeGrado labs applied CPD for designing a protein crystal [103]. A three-helix coiled-coil was designed de novo to form a polar and layered P6-space group crystal. An ensemble of crystalline structure models consistent with the required space group was constructed of which designable structures were datamined. These include minima structures in the sequence-structure energy landscape. Within the 26-residue peptide forming the C_3-symmetry coiled coil, the eight interior positions (*a* and *d* in the heptad repeat) were hydrophobic Val and Leu residues. The other 16 amino acids (not including Pro and Cys) were allowed to be positioned in other places. 19,200 structures were designed to construct a grid over R and θ, representing the inter-protein distance and the angle of rotation around the

superhelical angle, respectively. The final design included a parallel GX_3G motif interfacing the coiled-coil interhelical contact and an antiparallel GX_3GX_3A motif between the coiled coils. Exploiting the symmetry of the honeycomb-like space group, the resulting structure had sub-Å RMSD relative to the designed model.

In *2012* the DeGrado lab altered the function of a de novo Due Ferri four-helix bundle from catalyzing the O_2-dependent two-electron oxidation of hydroquinones to selectively catalyzing *N*-hydroxylation of arylamines [104]. This was conducted by remodeling the substrate access cavity and by introducing an additional His ligand to the metal-binding cavity. Further second- and third-shell CPD was applied using the Molecular Software Library (MSL [162]) to stabilize the catalytic core. The resulting design had a 10^6-fold rate enhancement towards the altered function relative to the previous one.

In *2013* the Hahn and Dokholyan labs applied CPD for the rational design of a ligand-controlled protein conformational switch [105]. Their unique topology design of a rapamycin-regulated switch, denoted *uniRapR*, was utilized as a src kinase activator. A high-affinity binding pocket of FK506-binding protein and FKBP12-rapamycin were used with the two proteins connected by a double linker. The first 20 residues of FKB12 were removed making the N- and C-terminii close in space for insertion of the regulatory domain to the other protein. The conformational switching was assessed by replica-exchange and equilibrium discrete molecular dynamics.

In *2013* the Therien, Saven and DeGrado labs joined forces and computationally de novo designed a protein that selectively binds a highly hyperpolarizable abiological chromophore [106]. The 109-residue four-helix-bundle was designated *SCRPZ-1* and *SCRPZ-2* for the dimeric and monomeric form, respectively. The protein binds RuPZn, a hyperpolarizable super-molecular chromophore that features highly conjugated (porphnato)zinc and (poly-pyridyl) ruthenium. The antiparallel four helix bundle was designed to accommodate the size of the chromophore and ligate the metal ions. Loops for connecting the helices were selected from natural proteins and spliced to accommodate the structure. The SCADS [143] software was used in two rounds first placing the keystone residues and then the other positions. 17 residues were allowed in the helices. His and Cys were precluded as a negative design approach to avoid unwanted metal ligations and disulfide bonds, respectively. Likewise, Pro was precluded from the helices to avoid unwanted kinks. For SCRPZ-2 the surface was then redesigned to decrease hydrophobic patches and incorporate interhelical salt bridges to increase bundle stability. A third design included Cys, enabling binding onto functionalized silica surfaces. The protein

structure, stability, and nonlinear optical functional elements were proven with an array of experimental methods.

In *2013* the Liu and Saven labs applied CPD for the design of a solubilized G-protein coupled receptor (GPCR)—the μ-opioid receptor [107]. The pain and addiction receptor underwent 53 mutations on the exterior surface solubilizing it completely without loss of structural characteristics and antagonist (naltrexone) binding affinity. Interestingly, the CPD was not conducted on a high-resolution known structure but rather on a comparative model using the β2 adrenergic receptor as a model with the subsequent structure of the murine μ-opioid receptor validating the model. Amino acids with >40 % solvent accessible surface area that were within the TM region were targeted for redesign within the SCADS framework [143] and the previous solubilization protocol [38]. To account for solvation effects, an environmental effective energy was employed based on the local density of C_β atoms of each residue and parameterized using a dataset of soluble proteins having up to 288 residues, the size of the TM domain of the targeted receptor. In *2014* five labs from the USA and South Korea (Johnson, Lieu, Saven, Park, Xi) joined forces and implemented this solubilized opioid receptor within a graphene field effect transistor (GRET) biosensor [108]. The receptor exhibited high sensitivity and selectivity for an opioid receptor antagonist (naltrexone), with an impressive detection limit of 10 pg/mL. The approach is general and can be applied for any GPCR, the family of proteins which form most drug targets and which suffers from experimental challenges following their intrinsic dynamics and embedment in the membrane.

In *2013* Baker and colleagues applied CPD for the design of a de novo lysozyme inhibitor [109]. Unlike the dock and design approach, e.g. the CPD of a weak affinity binder for PAK1 [82], here a hot-spot centric CPD approach was applied. This approach was previously applied to design proteins that bind the erythropoietin receptor [63] or the influenza hemagglutinin [86]. Here, the challenge included targeting deeply recessed residues within the charged active site of hen egg lysozyme (HEL). First a dock-and design approach was pursued: Coarse-docking was conducted on the HEL active site from a library of scaffold followed by several rounds of refined docking using RosettaDesign. Designed potentially binding proteins were analyzed as to binding energetics, shape-complementarity, packing, and size, aiming at measurables similar to native HEL complexes. The top 24 designs were displayed in a yeast library assessing binding affinity and specificity. Interestingly, the top-binder appeared to bind via a patch that is different than the one designed computationally, as evident from error-prone PCR affinity maturation which yielded affinity increasing mutations in other regions. Following these rarely reported negative results, a hot-spot centric approach was applied: An existing HEL complex was studied with computational alanine scanning

finding residues significantly contributing to binding and targeting active site residues. The two binding residues (Arg and Tyr) were held fixed and scaffolds were docked on them using PatchDock [153] followed by RosettaDock refinement. The two binding residues were transplanted on the scaffold with the aid of rigid-body minimization and the surrounding residues were designed with RosettaDesign. The top 21 designs were experimentally tested for affinity and specificity and the top design was optimized by error-prone PCR in a yeast display framework. From analysis of the best binder displaying low nanomolar affinity, it was concluded that specific interactions across a rather large interface are pivotal. In addition, it seems that the directed evolution experimental approach corrected poor hydrogen-bonding and electrostatic repulsion that was not sufficiently optimized by the CPD, suggesting room for algorithmic improvement.

In *2013* Baker and coworkers applied CPD for the de novo design of selective binders to the steroid digoxigenin (DIG), an example of a small molecule to which a protein binder can be designed [110]. The CPD of small molecule binders is challenging and indeed only two of 17 designs bound the molecule. Deep sequencing and library selections optimized the binding to picomolar levels. Three characteristics of naturally occurring binding sites were aimed: shape complementarity, specific energetically favorable hydrogen-bonds and van der Waals protein–ligand interactions as well as a structural pre-organization in the unbound protein state, which minimized entropy loss upon ligand binding. RosettaMatch [150] was used to identify backbone constellations in 401 protein scaffold structures where a DIG molecule and side chain conformations interacting with DIG in a predefined geometry could be accommodated. Two successive rounds of sequence design were used. The purpose of the first was to maximize binding affinity for the ligand. The goal of the second was to minimize protein destabilization due to aggressive scaffold mutagenesis while maintaining the binding interface designed during the first round. During the latter round, ligand–protein interactions were upweighted by a factor of 1.5 relative to intra-protein interactions to ensure that binding energy was preserved. No more than five residues were allowed to change from residue types observed in a multiple sequence alignment (MSA) of the scaffold if (a) these residues were present in the MSA with a frequency greater than 0.6, or (b) if the calculated $\Delta\Delta G$ for mutation of the scaffold residue to alanine was large. Designs were evaluated as to their interface energy, ligand solvent exposed surface area, ligand orientation, shape-complementarity, and apo-protein binding site pre-organization. The latter was enforced by explicitly introducing second-shell amino acids. The binding affinity of the directed evolution optimized design is similar to those of anti-digoxin antibodies. As it is stable for extended periods and can be expressed

at high levels in bacteria, the design has to potential to provide a more cost-effective alternative for biotechnological and for therapeutic purposes as long as it can be made compatible with the human immune response.

In *2014* the Baker lab designed a pH-sensitive Fc-domain IgG binding protein using the hot-spot centric approach [111]. His-433 on the IgG domain was targeted as a pH-sensitive site that should bind only under a specific pH range. Ensembles of disembodied interaction residues were based on the IgG complex with protein A. Scaffolds with high bacterial expression and solubility that can host the keystone residues were then searched. The rest of the interface was designed with RosettaDesign with ranking assisted by shape-complementarity and computed binding energy. Nine of 17 designs exhibited binding signals. At pH 8.2 the design bound the target 500-fold more tightly compared to pH 5.5.

In *2014* Liu, Chen and coworkers presented a new CPD method with a comprehensive statistical energy function (SEF) and systematic integration of experimental selection for foldability which was proven experimentally on two de novo structurally resolved designs [112]. In this important paper they highlight some of the challenges of existing rule-based or general-CPD methods, the latter minimizing a general effective energy function. Challenges include low success-rate on common targets, insufficient reflection of the diversity in natural sequences sharing a common structure and lack of the rich functional conformational dynamics in CPD results. While SEF are an integral part of numerous CPD methods, a full-scale SEF for automated CPD is not available as most general methods focus on physics-based energy functions. SEFs share the spirit of rule-based CPD, though the latter can include very few components which are not well calibrated between them. As such, the rule-based design, which often necessitates a human expert, receives here a systematic and coherent formalism. The SEF components including single-residue and pairwise terms with individual terms were determined by the probability distributions of rotamer types and pairs of rotamer types. Complementary, structure properties considered for single positions include secondary structure types, solvent accessibility, and backbone Ramachandran angles. Structural properties of pair terms also include the relative positioning in 3D space. Next, a general strategy for selecting structure neighbors with adaptive criteria (*SSNAC*) addressed the fact that some target properties are at the boundary of predefined boundary intervals and the need to treat multi-dimensional properties jointly. Small sample effects were corrected. Further, the publication aimed to establish the general applicability of an experimental approach assessing structural stability by linking it to antibiotic resistance in bacterial cells expressing an engineered TEM1-β-lactamase fused to the protein of interest. Unstable proteins are prone to proteolysis leading to weak antibiotic resistance. Comparing the SEF to fixed-backbone to

RosettaDesign, the authors claim that the SEF captures energy contributions that favor native sequences. The authors note that the SEF approach cannot treat packing in the same level as physics-based approaches, but seems to do a better job in capturing topology-related features, especially for β-strand containing topologies. Four well-folded de novo proteins for three different targets were obtained and two were structurally resolved validating the promising approach.

In *2014* the Baker lab applied CPD for the design of hyper-stable helical bundles [113]. Specifically, using Rosetta along with parametric backbone generation an antiparallel, monomeric untwisted three-helix bundle with 80-residue helices (18-residue repeat) was designed as well as an antiparallel right-handed monomeric four-helix bundle and a parallel left-handed five-helix bundle. While the classical coiled-coil structure is considered as a side-chain 'knobs-into-holes' structure, here the focus was on the less-appreciated contribution of backbone strain. Within the coiled-coil Crick parameters, a change of 2° in the helical twist and the coupled supercoil parameter can dictate the coiled coil twisting or lack of it. Within RosettaDesign, finer grid searches were undertaken in the vicinity of these parameters, yielding optimized designs. The resulting designs denatured only in >95 °C with 0.4–1.1 Å RMSD between the crystallographically resolved structures and the designs.

In *2014* Woolfson applied CPD for designing water-soluble α-helical barrels [114]. These are coiled-coils with more than four helices which form a central cavity. Within the *abcdefg* heptad repeat of coiled coils positions *gade* determine the oligomer state. As such, these positions were the focus of the design with specific positions relating to the requested coiled coil type. A bZIP scoring function was used to assess the fitness score of the homo-oligomer. Sequential rules were applied to reduce the set to be sampled and then Coiled Coil Builder (*CCBuilder*) was applied to construct the requested full-atom models. This includes the SOCKET knobs-into-holes packing assessment. Next, a genetic algorithm was applied to optimize radius, pitch, and inter-helical rotational offset. The designed pentamer, hexamer, and heptamer coiled coil were resolved crystallographically with RMSDs of 0.67–1.77 Å between the design and the actual structure.

In *2014* Negron and Keating combined the CLASSY [75] multi-state CPD and the distance-scaled, finite-gas reference (DFIRE [163]) state potential for de novo CPD of three coiled coils consisting three orthogonal antiparallel homodimers [115]. The heptad repeat coiled coil structure enabled the multi-state design scheme to provide a partition function between the stability and the specificity gap; allowing for the design of novel and experimentally prove 43-residue peptides folding into specific antiparallel homodimers. As such, a synthetic coiled-coil toolkit is provided for modular synthetic biology applications.

In *2014* the Schief lab collaborated with Baker and others to apply CPD for the important cause of epitope-focused vaccine design [116]. Their 27-author study focused on inducing potent neutralizing antibodies to small and stable CPD scaffolds which present a respiratory syncytial virus (RSV) epitope. The fold-from-loops (FFL) CPD Rosetta protocol starts by identifying a functional motif (epitope), which in this case was a helix-turn-helix motif in the RSV Fusion (F) glycoprotein, as identified from an antigen-antibody crystal structure. The epitope was placed on a target topology along with distance restraints of the scaffold, a thermally stable three-helix bundle. Then, ab initio folding was applied to build diverse backbone conformations consistent with the target topology. Successful low-resolution designs were subjected to an all-atom sequence design in which functional motif side chains were recovered followed by three rounds of sequence design and full-atom optimization. Last but not least, the 40,000 successful designs were evaluated by structural metrics and 8 designs were subjected to human-guided sequence design to correct potential flaws. These included replacing surface residues outside the epitope with the original template residues and designing larger hydrophobic residues at selected positions. One of the designs also underwent resurfacing (described above). The successful design induced neutralizing antibodies and was recognized by an existing antibody against the epitope.

In *2014* the DeGrado lab joined forces with three other labs, applying CPD for a de novo TM Zn^{2+}-transporting four-helix bundle [117]. The protein was named *ROCKER*. The first shell of the metal binding was inspired by a previous di-manganese four helix bundle while the second shell was adapted from that soluble structure for the TM milieu. A stochastic search over the helix-bundle Crick parameters was applied for a D2-symmetric anti-parallel tetrameric coiled-coil. A design alphabet was guided by the membrane depth (using the Ez potential [164]) and functional requirements of the different regions. Rotameric self and pair energies were compuated with a van der Waals radii reduced to 90 % of their size with the optimal rotameric conformation searched using a DEE/A* algorithm. 1008 resulting sequences had a preference for an asymmetric state, excluding the transporter from being filled with two ions. To confirm an asymmetric rather than symmetric conformation, each of these sequences was subjected to the two-state free-energy comparison evaluator algorithm VALOCIDY (Valuation of Local Configuration Integral with Dynamics [165]) using independent MD trajectories. The protein was extensively characterized structurally and functionally, confirming the CPD models.

In *2014* Baker and coworkers applied CPD for reducing immunogenicity by removing T-cell epitopes [118]. As proteins represent the fastest-growing class of pharmaceuticals, their deimmunization

is of growing need. MHC-II-binding short-sequence epitopes have been characterized. Herein, a sliding window of 15-residues was searched using a support vector machine (SVM) for T-cell epitopes. These were searched and potential epitope sites were redesigned without losing structure, stability, and function. As the deimmunization scores favor negatively charged residues, a net charge constraint was added. First, they computationally recapitulated a previous deimmunization effort. Second, the method was experimentally validated on the superfolder green fluorescent protein (sfGFP) by redesigning the top four predicted H-2-IAb epitopes. The deimmunized protein designs failed to isolate T cells in mice while maintaining function. Third, 5 mutations were aimed at removing 3 epitopes in the toxin domain of the cancer therapeutic HA22, a potential drug for refractory cell leukemia. Two of these mutants lost 80 % of the cytotoxic effect while other mutants displayed increased effect.

In *2014* Zhang, Tame and coworkers applied CPD for the design of a self-assembling sixfold perfectly symmetric β-propeller protein [119]. Visual examination of 174 β-propeller proteins was applied to choose the most visually symmetric protein for design. Therein, ancestor reconstruction of one of the six blades was applied followed by reverse engineering of a 6-blade protein. The process included docking of the blades and side-chain design in which essential inter-blade interacting residues were left as is. The actual design was experimentally proven to have an excellent 0.68 Å-backbone RMSD to the designed model.

In *2014* the Andre lab designed a leucine-rich repeat from the ribonuclease inhibitor family with predefined geometry [120]. Designated software was utilized to determine the length, curvature, and twist geometrical features. The protocol first defined the desired protein geometry. Second, a library of structures of individual repeats was compiled from crystal structures of selected repeat proteins. Third, self-compatible repeats capable of symmetrical assembly were selected. Fourth, the inter-repeat interface was optimized by cycles of docking and sequence optimization. Fifth, consecutive repeats were connected by loops. Last, capping was added to most N- and C-terminal repeats. A five double-repeat protein was confirmed to fold into a novel ring for the cap-less design and to a well-defined repeat protein when the caps were included.

5 CPD Failed Efforts and Retractions

Description of achievements and challenges of CPD cannot be complete without mentioning cases in which CPD publications were retracted. Naturally, published science highlights success stories rather than failures. Nevertheless, in some cases the failed attempt to repeat a published study results in exposing an

erroneous or disputed scientific publication. The need to analyze and understand failed efforts was highlighted by Mayo [100] in his description of an iterative design cycle: "*Proteins from failed computational design efforts are typically discarded without comment or investigation into the cause of failure. This situation is unfortunate, because valuable information is lost when successful designs are reported. Without detailed computational and/or experimental analysis of failed designs, flaws in the design procedure cannot be identified and remedied.*"

The field of protein design had suffered from several such incidents, partly as the proof of the output protein is not always straightforward. The resulting retracted publications may be due to innocent mistakes, insufficient validation or potentially even cheating in reporting the research. This section aims to present key retractions without getting into the details underlying the retractions. Rather, such retractions remind us of the caution required in reporting CPD studies and the need to unequivocally validate the result of the CPD process.

In *2008* Dwyer, Looger, and Hellinga retracted [166] their *2004 Science* [167] publication which attempted to describe the first computational enzyme design, a triose phosphate isomerase (TIM) in a computationally redesigned ribose-binding protein. The retraction states that this is following a report that the provided clones that were supposed to be clones of the designed enzyme were actually clones of *wild-type* TIM impurity. In addition, a *JMB* computational enzyme design publication by the same group was retracted [168]. Following these retractions questions arose [169, 170] including over the validity of a *2003 Nature* paper describing computational redesign of ligand-binding specificities [171] and a *2004 PNAS* paper describing the CPD of receptors for an organophosphate surrogate of the nerve agent soman [172]. Notably, these papers were not retracted. Importantly, Hellinga has acknowledged responsibility for the two retractions and asked his university to hold an inquiry regarding them [173].

Unfortunately, retractions in the field of protein design are not limited to CPD. For example, following cross-contamination, in *2002* Fersht and coworkers have retracted [174] their *Nature* paper [175] on the directed evolution of new catalytic activity using the α/β-barrel scaffold.

In summary, these retractions following irreproducible results and the heated debate that followed should remind us of the special care required in experimentally characterizing and confirming that the CPD product is indeed the designed protein.

6 Concluding Remarks: Future Challenges

Many aspects of CPD has been reviewed in the past [121, 122, 176–182], yet a chronological case-study review of the field is presented here for the first time. The field of CPD has undergone a tremendous leap forward in the three decades in which it exists. CPD demonstrates the ability to design functional and extremophile complex proteins with great precision using a wide array of tailored methods as well as imported methods from other fields. Taken together, it seems that the achievements and challenges of the CPD field reflect that of the broader structural bioinformatics and computational biophysics [183] field.

Some of the pending challenges include:

1. Accessibility to the general relevant scientific community. Thus far, the main efforts in the field of CPD were not distributed among a large community but rather clustered in a small number of labs (*see* Table 3 for list of main labs and software packages). Often, the CPD software packages are used solely '*in-house*' and not utilized by the general community, even if the software is open-source. CPD requires multidisciplinary know-how in structural biology, biophysics, biochemistry, software engineering, and a general nontrivial combination of theory and experiment. As with other fields, it is expected that with time more and more scientists will apply CPD for their research and consequently use software developed by others.
2. Integration of knowledge-based and energy-based methods: Ideally, all design algorithms will rely on physics to address the enthalpic and entropic energetic contributions. Yet, within the complex protein milieu and within the foreseeable future of computer power, such a description is not practical in high resolution. Currently, it seems that each design lab selects a different method of integrating knowledge-based know-how into the design—from selection of hydrophobic or helix-forming amino acids to use of known structural motifs or structural fragments. A systematic and comparative analysis of the different design schemes may help determine better guidelines on this aspect.
3. Systematic differential approach towards different proteins levels of organization, different protein regions, and the relationships between such regions. While often the design is split to solvent-exposed and buried regions, the adaption of the CPD algorithm to the local milieu of the target site is still not optimized.
4. Assessment of electrostatics and solvation effects: Coupled to the previous item, the local dielectric milieu and long-range

Table 3
List of main labs discussed in this chapter and their CPD software packages. Notably, only the main software packages for CPD are described here. Further, only the main and most known software of each lab is mentioned with some of the specific case-studies of each lab not necessarily using that software. Additional software for different aspects of CPD are described throughout the book. For a brief description of the main early-stage software packages see [184]

	Year	Lab	Software	Publication of main software	Lab publications discussed in the chapter
1.	1985–	DeGrado	ProtCad for few studies but most CPD case-studies with numerous other tools.	[160]	[7, 9, 22, 23, 29, 38, 39, 41, 42, 47, 48, 51, 54, 55, 61, 65, 70, 78, 81, 85, 92, 103, 104, 106, 117, 123, 127, 144, 147, 164]
2.	1987	Ponder	PROPAK	[126]	
3.	1991	Hellinga	DEZYMER	[125]	[11, 125, 166, 167, 171–173]
4.	1995	Desjarlais	Repacking of Core (ROC) & SPA (Sequence Prediction Algorithm)	[15, 128]	[15–17, 22, 44, 128, 185]
5.	1996, 1997	Mayo	PDA (Protein Design Automation) and ORBIT (Optimization of Rotamers by Iterative Techniques), Other programs include Phoenix, Triad, Faster	[18, 141]	[18–20, 27, 31, 32, 60, 64, 80, 83, 100, 101, 141]
6.	2000	Baker	Rosetta (including RosettaDesign, RosettaMatch etc.)	[133, 134, 150]	[25, 36, 43, 45, 46, 49, 50, 52, 53, 58, 71–73, 76, 79, 86, 87, 95, 98, 109–111, 113, 116, 118, 133, 134, 138–140, 142, 149, 150]
7.	2000	Kuhlman	RosettaDesign	[134]	[25, 43, 46, 62, 82, 88, 134]
8.	2001	Saven	SCADS (Statistical Computationally Assisted Design Strategy)	[143]	[41, 51, 54–56, 68, 70, 78, 90, 94, 103, 107, 121, 143, 146, 164]
9.	2001	Serrano	PERLA (protein Engineering Rotamer Library Algorithm)	[135]	[24, 26–30, 57, 135, 136, 182]
10.	2014	Chen & Liu	Statistical energy function and boosted by experimental selection for foldability	[112]	[112]

electrostatic interactions are still not sufficiently modeled within CPD software.

5. Integration of thermal plasticity and functional dynamics: While a generalization, the incorporation of dynamics into the design scheme is still not done, despite the hard-wired dynamic functional profile of every protein as e.g. depicted by quick Gaussian network models.
6. Negative design: Negative design, defined as a design aimed at avoiding unwanted conformations or functions, must be an explicit part of computational design. Since the 1991 thioredoxin redesign [11] and the betadoublet, a β-sandwich de novo design [14], the negative design aspect has been in the forefront of the field. While the importance of negative design is well acknowledged since early days of CPD [185], it is still not explicitly integrated into design algorithms. In this respect, the positive-design scheme explicitly or implicitly regards a reference state which can often be considered as a negative design element. However, too often insufficient emphasis is given to the definition of the reference state.
7. Systematic integration of experimental design approaches: the theoretical rational design is moving towards integration with experimental semi-rational design approaches such as directed evolution. Yet, currently the number of designs benefiting from the combination of approaches is still small. Moreover, there is no systematic protocol for combining the two approaches or even for reporting the stage to which each approach has advanced the target design.
8. Objective cross-assessment of methods: To date, there has not been an objective cross-assessment of the different available methods, as done for e.g. structure prediction via the Critical Assessment of Structure Prediction (CASP) competition [186] which is running since 1995. Therein, the community is given a mutual target to be submitted to assessors who are not part of the competitors thus enabling objective analysis of achievements and challenges in a method comparative manner. Without such a community-wide objective assessment the comparative analysis of CPD methods is often challenging relying solely on reports by the respective authors for each tool. Consequently, the identification of advantages and disadvantages of each method and the cross-dissemination of knowledge is hampered.
9. Definition of the reference state: In many cases the scoring function consists of scoring the gap between the desired state and the nondesired, e.g. denatured one. However, the reference state is still not sufficiently defined, let alone divided between protein and cellular regions.

10. *In vivo* CPD: Many designs are not stable and prone to aggregation [111]. As seen from the case-studies presented, the vast majority of designs were not characterized within an *in vivo* setting, which is the ultimate natural environment of proteins.

Each of the above items deserves a separate chapter. Yet, after highlighting some of the pending challenges, it is important to emphasize that the hierarchical approach to CPD has advanced in all levels—from large scaffold searches in the growing PDB to quantum-mechanical optimization of enzymatic catalytic sites. In parallel the richness in knowledge-based and physics-based methodology sets the stage to comparative analysis of methods and the dissemination of methods from the method creators to the general community of protein scientists.

References

1. Fischer E (1966) In: Nobelstiftelsen (ed) Nobel lectures, chemistry 1901–1921, vol 1. Elsevier, Amsterdam, p 21–35
2. Anfinsen CB, Harrington WF, Hvidt A, Linderstrom-Lang K, Ottesen M, Schellman J (1989) Studies on the structural basis of ribonuclease activity. 1955. Biochim Biophys Acta 1000:200–201
3. Drexler KE (1981) Molecular engineering: an approach to the development of general capabilities for molecular manipulation. Proc Natl Acad Sci U S A 78(9):5275–5278
4. Pabo C (1983) Molecular technology. Designing proteins and peptides. Nature 301 (5897):200
5. Jaramillo A, Wernisch L, Hery S, Wodak SJ (2002) Folding free energy function selects native-like protein sequences in the core but not on the surface. Proc Natl Acad Sci U S A 99(21):13554–13559. doi:10.1073/pnas.212068599
6. Wernisch L, Hery S, Wodak SJ (2000) Automatic protein design with all atom force-fields by exact and heuristic optimization. J Mol Biol 301(3):713–736. doi:10.1006/jmbi.2000.3984
7. DeGrado WF, Prendergast FG, Wolfe HR Jr, Cox JA (1985) The design, synthesis, and characterization of tight-binding inhibitors of calmodulin. J Cell Biochem 29(2):83–93. doi:10.1002/jcb.240290204
8. Craik CS, Largman C, Fletcher T, Roczniak S, Barr PJ, Fletterick R, Rutter WJ (1985) Redesigning trypsin: alteration of substrate specificity. Science 228(4697):291–297
9. Vonderviszt F, Matrai G, Simon I (1986) Characteristic sequential residue environment of amino acids in proteins. Int J Pept Protein Res 27(5):483–492
10. Hecht MH, Richardson JS, Richardson DC, Ogden RC (1990) De novo design, expression, and characterization of Felix: a four-helix bundle protein of native-like sequence. Science 249(4971):884–891
11. Hellinga HW, Caradonna JP, Richards FM (1991) Construction of new ligand binding sites in proteins of known structure. II. Grafting of a buried transition metal binding site into Escherichia coli thioredoxin. J Mol Biol 222(3):787–803
12. Wilson C, Mace JE, Agard DA (1991) Computational method for the design of enzymes with altered substrate specificity. J Mol Biol 220(2):495–506
13. Hurley JH, Baase WA, Matthews BW (1992) Design and structural analysis of alternative hydrophobic core packing arrangements in bacteriophage T4 lysozyme. J Mol Biol 224 (4):1143–1159
14. Quinn TP, Tweedy NB, Williams RW, Richardson JS, Richardson DC (1994) Beta-doublet: de novo design, synthesis, and characterization of a beta-sandwich protein. Proc Natl Acad Sci U S A 91(19):8747–8751
15. Desjarlais JR, Handel TM (1995) De novo design of the hydrophobic cores of proteins. Protein Sci 4(10):2006–2018. doi:10.1002/pro.5560041006
16. Lazar GA, Desjarlais JR, Handel TM (1997) De novo design of the hydrophobic core of ubiquitin. Protein Sci 6(6):1167–1178. doi:10.1002/pro.5560060605
17. Johnson EC, Lazar GA, Desjarlais JR, Handel TM (1999) Solution structure and dynamics

of a designed hydrophobic core variant of ubiquitin. Structure 7(8):967–976

18. Dahiyat BI, Mayo SL (1997) De novo protein design: fully automated sequence selection. Science 278(5335):82–87
19. Dahiyat BI, Sarisky CA, Mayo SL (1997) De novo protein design: towards fully automated sequence selection. J Mol Biol 273 (4):789–796. doi:10.1006/jmbi.1997.1341
20. Malakauskas SM, Mayo SL (1998) Design, structure and stability of a hyperthermophilic protein variant. Nat Struct Biol 5(6):470–475
21. Harbury PB, Plecs JJ, Tidor B, Alber T, Kim PS (1998) High-resolution protein design with backbone freedom. Science 282 (5393):1462–1467
22. Bryson JW, Desjarlais JR, Handel TM, DeGrado WF (1998) From coiled coils to small globular proteins: design of a native-like three-helix bundle. Protein Sci 7 (6):1404–1414. doi:10.1002/pro.5560070617
23. Walsh ST, Cheng H, Bryson JW, Roder H, DeGrado WF (1999) Solution structure and dynamics of a de novo designed three-helix bundle protein. Proc Natl Acad Sci U S A 96 (10):5486–5491
24. Domingues H, Cregut D, Sebald W, Oschkinat H, Serrano L (1999) Rational design of a GCN4-derived mimetic of interleukin-4. Nat Struct Biol 6(7):652–656. doi:10.1038/10706
25. Kuhlman B, O'Neill JW, Kim DE, Zhang KY, Baker D (2001) Conversion of monomeric protein L to an obligate dimer by computational protein design. Proc Natl Acad Sci U S A 98(19):10687–10691. doi:10.1073/pnas.181354398
26. Lopez de la Paz M, Lacroix E, Ramirez-Alvarado M, Serrano L (2001) Computer-aided design of beta-sheet peptides. J Mol Biol 312(1):229–246. doi:10.1006/jmbi.2001.4918
27. Bolon DN, Mayo SL (2001) Enzyme-like proteins by computational design. Proc Natl Acad Sci U S A 98(25):14274–14279. doi:10.1073/pnas.251555398
28. Keating AE, Malashkevich VN, Tidor B, Kim PS (2001) Side-chain repacking calculations for predicting structures and stabilities of heterodimeric coiled coils. Proc Natl Acad Sci U S A 98(26):14825–14830. doi:10.1073/pnas.261563398
29. Summa CM, Rosenblatt MM, Hong JK, Lear JD, DeGrado WF (2002) Computational de novo design, and characterization of an A(2)B (2) diiron protein. J Mol Biol 321 (5):923–938
30. Ventura S, Vega MC, Lacroix E, Angrand I, Spagnolo L, Serrano L (2002) Conformational strain in the hydrophobic core and its implications for protein folding and design. Nat Struct Biol 9(6):485–493. doi:10.1038/nsb799
31. Shifman JM, Mayo SL (2002) Modulating calmodulin binding specificity through computational protein design. J Mol Biol 323(3):417–423
32. Shifman JM, Mayo SL (2003) Exploring the origins of binding specificity through the computational redesign of calmodulin. Proc Natl Acad Sci U S A 100(23):13274–13279. doi:10.1073/pnas.2234277100
33. Hayes RJ, Bentzien J, Ary ML, Hwang MY, Jacinto JM, Vielmetter J, Kundu A, Dahiyat BI (2002) Combining computational and experimental screening for rapid optimization of protein properties. Proc Natl Acad Sci U S A 99(25):15926–15931. doi:10.1073/pnas.212627499
34. Filikov AV, Hayes RJ, Luo P, Stark DM, Chan C, Kundu A, Dahiyat BI (2002) Computational stabilization of human growth hormone. Protein Sci 11(6):1452–1461. doi:10.1110/ps.3500102
35. Luo P, Hayes RJ, Chan C, Stark DM, Hwang MY, Jacinto JM, Juvvadi P, Chung HS, Kundu A, Ary ML, Dahiyat BI (2002) Development of a cytokine analog with enhanced stability using computational ultrahigh throughput screening. Protein Sci 11(5):1218–1226. doi:10.1110/ps.4580102
36. Chevalier BS, Kortemme T, Chadsey MS, Baker D, Monnat RJ, Stoddard BL (2002) Design, activity, and structure of a highly specific artificial endonuclease. Mol Cell 10 (4):895–905
37. Ogata K, Jaramillo A, Cohen W, Briand JP, Connan F, Choppin J, Muller S, Wodak SJ (2003) Automatic sequence design of major histocompatibility complex class I binding peptides impairing CD8+ T cell recognition. J Biol Chem 278(2):1281–1290. doi:10.1074/jbc.M206853200
38. Slovic AM, Summa CM, Lear JD, DeGrado WF (2003) Computational design of a water-soluble analog of phospholamban. Protein Sci 12(2):337–348. doi:10.1110/ps.0226603
39. Slovic AM, Stayrook SE, North B, Degrado WF (2005) X-ray structure of a water-soluble analog of the membrane protein phospholamban: sequence determinants defining the topology of tetrameric and pentameric coiled coils. J Mol Biol 348(3):777–787. doi:10.1016/j.jmb.2005.02.040

40. Havranek JJ, Harbury PB (2003) Automated design of specificity in molecular recognition. Nat Struct Biol 10(1):45–52. doi:10.1038/nsb877
41. Calhoun JR, Kono H, Lahr S, Wang W, DeGrado WF, Saven JG (2003) Computational design and characterization of a monomeric helical dinuclear metalloprotein. J Mol Biol 334(5):1101–1115
42. Calhoun JR, Liu W, Spiegel K, Dal Peraro M, Klein ML, Valentine KG, Wand AJ, DeGrado WF (2008) Solution NMR structure of a designed metalloprotein and complementary molecular dynamics refinement. Structure 16 (2):210–215. doi:10.1016/j.str.2007.11.011
43. Kuhlman B, Dantas G, Ireton GC, Varani G, Stoddard BL, Baker D (2003) Design of a novel globular protein fold with atomic-level accuracy. Science 302(5649):1364–1368. doi:10.1126/science.1089427
44. Kraemer-Pecore CM, Lecomte JT, Desjarlais JR (2003) A de novo redesign of the WW domain. Protein Sci 12(10):2194–2205. doi:10.1110/ps.03190903
45. Dantas G, Kuhlman B, Callender D, Wong M, Baker D (2003) A large scale test of computational protein design: folding and stability of nine completely redesigned globular proteins. J Mol Biol 332(2):449–460
46. Dantas G, Corrent C, Reichow SL, Havranek JJ, Eletr ZM, Isern NG, Kuhlman B, Varani G, Merritt EA, Baker D (2007) High-resolution structural and thermodynamic analysis of extreme stabilization of human procarboxypeptidase by computational protein design. J Mol Biol 366(4):1209–1221. doi:10.1016/j.jmb.2006.11.080
47. Di Costanzo L, Wade H, Geremia S, Randaccio L, Pavone V, DeGrado WF, Lombardi A (2001) Toward the de novo design of a catalytically active helix bundle: a substrate-accessible carboxylate-bridged dinuclear metal center. J Am Chem Soc 123 (51):12749–12757
48. Kaplan J, DeGrado WF (2004) De novo design of catalytic proteins. Proc Natl Acad Sci U S A 101(32):11566–11570. doi:10.1073/pnas.0404387101
49. Kortemme T, Joachimiak LA, Bullock AN, Schuler AD, Stoddard BL, Baker D (2004) Computational redesign of protein-protein interaction specificity. Nat Struct Mol Biol 11(4):371–379. doi:10.1038/nsmb749
50. Joachimiak LA, Kortemme T, Stoddard BL, Baker D (2006) Computational design of a new hydrogen bond network and at least a 300-fold specificity switch at a protein-protein interface. J Mol Biol 361(1):195–208. doi:10.1016/j.jmb.2006.05.022
51. Slovic AM, Kono H, Lear JD, Saven JG, DeGrado WF (2004) Computational design of water-soluble analogues of the potassium channel KcsA. Proc Natl Acad Sci U S A 101 (7):1828–1833. doi:10.1073/pnas.0306417101
52. Korkegian A, Black ME, Baker D, Stoddard BL (2005) Computational thermostabilization of an enzyme. Science 308 (5723):857–860. doi:10.1126/science.1107387
53. Bolon DN, Grant RA, Baker TA, Sauer RT (2005) Specificity versus stability in computational protein design. Proc Natl Acad Sci U S A 102(36):12724–12729. doi:10.1073/pnas.0506124102
54. Nanda V, Rosenblatt MM, Osyczka A, Kono H, Getahun Z, Dutton PL, Saven JG, Degrado WF (2005) De novo design of a redox-active minimal rubredoxin mimic. J Am Chem Soc 127(16):5804–5805. doi:10.1021/ja050553f
55. Cochran FV, Wu SP, Wang W, Nanda V, Saven JG, Therien MJ, DeGrado WF (2005) Computational de novo design and characterization of a four-helix bundle protein that selectively binds a nonbiological cofactor. J Am Chem Soc 127(5):1346–1347. doi:10.1021/ja044129a
56. Swift J, Wehbi WA, Kelly BD, Stowell XF, Saven JG, Dmochowski IJ (2006) Design of functional ferritin-like proteins with hydrophobic cavities. J Am Chem Soc 128 (20):6611–6619. doi:10.1021/ja057069x
57. van der Sloot AM, Tur V, Szegezdi E, Mullally MM, Cool RH, Samali A, Serrano L, Quax WJ (2006) Designed tumor necrosis factor-related apoptosis-inducing ligand variants initiating apoptosis exclusively via the DR5 receptor. Proc Natl Acad Sci U S A 103 (23):8634–8639. doi:10.1073/pnas.0510187103
58. Ashworth J, Havranek JJ, Duarte CM, Sussman D, Monnat RJ Jr, Stoddard BL, Baker D (2006) Computational redesign of endonuclease DNA binding and cleavage specificity. Nature 441(7093):656–659. doi:10.1038/nature04818
59. Lazar GA, Dang W, Karki S, Vafa O, Peng JS, Hyun L, Chan C, Chung HS, Eivazi A, Yoder SC, Vielmetter J, Carmichael DF, Hayes RJ, Dahiyat BI (2006) Engineered antibody Fc variants with enhanced effector function. Proc Natl Acad Sci U S A 103

(11):4005–4010. doi:10.1073/pnas.0508123103

60. Huang PS, Love JJ, Mayo SL (2007) A de novo designed protein protein interface. Protein Sci 16(12):2770–2774. doi:10.1110/ps.073125207
61. Yin H, Slusky JS, Berger BW, Walters RS, Vilaire G, Litvinov RI, Lear JD, Caputo GA, Bennett JS, DeGrado WF (2007) Computational design of peptides that target transmembrane helices. Science 315 (5820):1817–1822. doi:10.1126/science.1136782
62. Hu X, Wang H, Ke H, Kuhlman B (2007) High-resolution design of a protein loop. Proc Natl Acad Sci U S A 104 (45):17668–17673. doi:10.1073/pnas.0707977104
63. Liu S, Liu S, Zhu X, Liang H, Cao A, Chang Z, Lai L (2007) Nonnatural protein-protein interaction-pair design by key residues grafting. Proc Natl Acad Sci U S A 104 (13):5330–5335. doi:10.1073/pnas.0606198104
64. Shah PS, Hom GK, Ross SA, Lassila JK, Crowhurst KA, Mayo SL (2007) Full-sequence computational design and solution structure of a thermostable protein variant. J Mol Biol 372(1):1–6. doi:10.1016/j.jmb.2007.06.032
65. Bender GM, Lehmann A, Zou H, Cheng H, Fry HC, Engel D, Therien MJ, Blasie JK, Roder H, Saven JG, DeGrado WF (2007) De novo design of a single-chain diphenylporphyrin metalloprotein. J Am Chem Soc 129 (35):10732–10740. doi:10.1021/ja071199j
66. Lippow SM, Wittrup KD, Tidor B (2007) Computational design of antibody-affinity improvement beyond in vivo maturation. Nat Biotechnol 25(10):1171–1176. doi:10.1038/nbt1336
67. Potapov V, Reichmann D, Abramovich R, Filchtinski D, Zohar N, Ben Halevy D, Edelman M, Sobolev V, Schreiber G (2008) Computational redesign of a protein-protein interface for high affinity and binding specificity using modular architecture and naturally occurring template fragments. J Mol Biol 384 (1):109–119. doi:10.1016/j.jmb.2008.08.078
68. Butts CA, Swift J, Kang SG, Di Costanzo L, Christianson DW, Saven JG, Dmochowski IJ (2008) Directing noble metal ion chemistry within a designed ferritin protein. Biochemistry 47(48):12729–12739. doi:10.1021/bi8016735
69. Reynolds KA, Hanes MS, Thomson JM, Antczak AJ, Berger JM, Bonomo RA, Kirsch JF, Handel TM (2008) Computational redesign of the SHV-1 beta-lactamase/beta-lactamase inhibitor protein interface. J Mol Biol 382(5):1265–1275. doi:10.1016/j.jmb.2008.05.051
70. McAllister KA, Zou H, Cochran FV, Bender GM, Senes A, Fry HC, Nanda V, Keenan PA, Lear JD, Saven JG, Therien MJ, Blasie JK, DeGrado WF (2008) Using alpha-helical coiled-coils to design nanostructured metalloporphyrin arrays. J Am Chem Soc 130 (36):11921–11927. doi:10.1021/ja800697g
71. Jiang L, Althoff EA, Clemente FR, Doyle L, Rothlisberger D, Zanghellini A, Gallaher JL, Betker JL, Tanaka F, Barbas CF 3rd, Hilvert D, Houk KN, Stoddard BL, Baker D (2008) De novo computational design of retro-aldol enzymes. Science 319(5868):1387–1391. doi:10.1126/science.1152692
72. Rothlisberger D, Khersonsky O, Wollacott AM, Jiang L, DeChancie J, Betker J, Gallaher JL, Althoff EA, Zanghellini A, Dym O, Albeck S, Houk KN, Tawfik DS, Baker D (2008) Kemp elimination catalysts by computational enzyme design. Nature 453 (7192):190–195. doi:10.1038/nature06879
73. Murphy PM, Bolduc JM, Gallaher JL, Stoddard BL, Baker D (2009) Alteration of enzyme specificity by computational loop remodeling and design. Proc Natl Acad Sci U S A 106(23):9215–9220. doi:10.1073/pnas.0811070106
74. Yosef E, Politi R, Choi MH, Shifman JM (2009) Computational design of calmodulin mutants with up to 900-fold increase in binding specificity. J Mol Biol 385(5):1470–1480. doi:10.1016/j.jmb.2008.09.053
75. Grigoryan G, Reinke AW, Keating AE (2009) Design of protein-interaction specificity gives selective bZIP-binding peptides. Nature 458 (7240):859–864. doi:10.1038/nature07885
76. Thyme SB, Jarjour J, Takeuchi R, Havranek JJ, Ashworth J, Scharenberg AM, Stoddard BL, Baker D (2009) Exploitation of binding energy for catalysis and design. Nature 461 (7268):1300–1304. doi:10.1038/nature08508
77. Chen CY, Georgiev I, Anderson AC, Donald BR (2009) Computational structure-based redesign of enzyme activity. Proc Natl Acad Sci U S A 106(10):3764–3769. doi:10.1073/pnas.0900266106
78. Fry HC, Lehmann A, Saven JG, DeGrado WF, Therien MJ (2010) Computational design and elaboration of a de novo heterotetrameric alpha-helical protein that selectively binds an emissive abiological (porphinato)

zinc chromophore. J Am Chem Soc 132 (11):3997–4005. doi:10.1021/ja907407m

79. Ashworth J, Taylor GK, Havranek JJ, Quadri SA, Stoddard BL, Baker D (2010) Computational reprogramming of homing endonuclease specificity at multiple adjacent base pairs. Nucleic Acids Res 38(16):5601–5608. doi:10.1093/nar/gkq283
80. Chica RA, Moore MM, Allen BD, Mayo SL (2010) Generation of longer emission wavelength red fluorescent proteins using computationally designed libraries. Proc Natl Acad Sci U S A 107(47):20257–20262. doi:10.1073/pnas.1013910107
81. Korendovych IV, Senes A, Kim YH, Lear JD, Fry HC, Therien MJ, Blasie JK, Walker FA, Degrado WF (2010) De novo design and molecular assembly of a transmembrane diporphyrin-binding protein complex. J Am Chem Soc 132(44):15516–15518. doi:10.1021/ja107487b
82. Jha RK, Leaver-Fay A, Yin S, Wu Y, Butterfoss GL, Szyperski T, Dokholyan NV, Kuhlman B (2010) Computational design of a PAK1 binding protein. J Mol Biol 400(2):257–270. doi:10.1016/j.jmb.2010.05.006
83. Allen BD, Nisthal A, Mayo SL (2010) Experimental library screening demonstrates the successful application of computational protein design to large structural ensembles. Proc Natl Acad Sci U S A 107(46):19838–19843. doi:10.1073/pnas.1012985107
84. Frey KM, Georgiev I, Donald BR, Anderson AC (2010) Predicting resistance mutations using protein design algorithms. Proc Natl Acad Sci U S A 107(31):13707–13712. doi:10.1073/pnas.1002162107
85. Grigoryan G, Kim YH, Acharya R, Axelrod K, Jain RM, Willis L, Drndic M, Kikkawa JM, DeGrado WF (2011) Computational design of virus-like protein assemblies on carbon nanotube surfaces. Science 332(6033):1071–1076. doi:10.1126/science.1198841
86. Fleishman SJ, Whitehead TA, Ekiert DC, Dreyfus C, Corn JE, Strauch EM, Wilson IA, Baker D (2011) Computational design of proteins targeting the conserved stem region of influenza hemagglutinin. Science 332(6031):816–821. doi:10.1126/science.1202617
87. Karanicolas J, Corn JE, Chen I, Joachimiak LA, Dym O, Peck SH, Albeck S, Unger T, Hu W, Liu G, Delbecq S, Montelione GT, Spiegel CP, Liu DR, Baker D (2011) A de novo protein binding pair by computational design and directed evolution. Mol Cell 42 (2):250–260. doi:10.1016/j.molcel.2011.03.010
88. Stranges PB, Machius M, Miley MJ, Tripathy A, Kuhlman B (2011) Computational design of a symmetric homodimer using beta-strand assembly. Proc Natl Acad Sci U S A 108 (51):20562–20567. doi:10.1073/pnas.1115124108
89. Correia BE, Ban YE, Friend DJ, Ellingson K, Xu H, Boni E, Bradley-Hewitt T, Bruhn-Johannsen JF, Stamatatos L, Strong RK, Schief WR (2011) Computational protein design using flexible backbone remodeling and resurfacing: case studies in structure-based antigen design. J Mol Biol 405(1):284–297. doi:10.1016/j.jmb.2010.09.061
90. Diaz JE, Lin CS, Kunishiro K, Feld BK, Avrantinis SK, Bronson J, Greaves J, Saven JG, Weiss GA (2011) Computational design and selections for an engineered, thermostable terpene synthase. Protein Sci 20 (9):1597–1606. doi:10.1002/pro.691
91. Xu F, Zahid S, Silva T, Nanda V (2011) Computational design of a collagen A:B:C-type heterotrimer. J Am Chem Soc 133 (39):15260–15263. doi:10.1021/ja205597g
92. Korendovych IV, Kulp DW, Wu Y, Cheng H, Roder H, DeGrado WF (2011) Design of a switchable eliminase. Proc Natl Acad Sci U S A 108(17):6823–6827. doi:10.1073/pnas.1018191108
93. Merski M, Shoichet BK (2012) Engineering a model protein cavity to catalyze the Kemp elimination. Proc Natl Acad Sci U S A 109 (40):16179–16183. doi:10.1073/pnas.1208076109
94. Cui T, Mowrey D, Bondarenko V, Tillman T, Ma D, Landrum E, Perez-Aguilar JM, He J, Wang W, Saven JG, Eckenhoff RG, Tang P, Xu Y (2012) NMR structure and dynamics of a designed water-soluble transmembrane domain of nicotinic acetylcholine receptor. Biochim Biophys Acta 1818(3):617–626. doi:10.1016/j.bbamem.2011.11.021
95. Khare SD, Kipnis Y, Greisen P Jr, Takeuchi R, Ashani Y, Goldsmith M, Song Y, Gallaher JL, Silman I, Leader H, Sussman JL, Stoddard BL, Tawfik DS, Baker D (2012) Computational redesign of a mononuclear zinc metalloenzyme for organophosphate hydrolysis. Nat Chem Biol 8(3):294–300. doi:10.1038/nchembio.777
96. Azoitei ML, Ban YE, Julien JP, Bryson S, Schroeter A, Kalyuzhniy O, Porter JR, Adachi Y, Baker D, Pai EF, Schief WR (2012) Computational design of high-affinity epitope scaffolds by backbone grafting of a linear epitope. J Mol Biol 415(1):175–192. doi:10.1016/j.jmb.2011.10.003

97. Kapp GT, Liu S, Stein A, Wong DT, Remenyi A, Yeh BJ, Fraser JS, Taunton J, Lim WA, Kortemme T (2012) Control of protein signaling using a computationally designed GTPase/GEF orthogonal pair. Proc Natl Acad Sci U S A 109(14):5277–5282. doi:10.1073/pnas.1114487109
98. Koga N, Tatsumi-Koga R, Liu G, Xiao R, Acton TB, Montelione GT, Baker D (2012) Principles for designing ideal protein structures. Nature 491(7423):222–227. doi:10.1038/nature11600
99. Fallas JA, Hartgerink JD (2012) Computational design of self-assembling register-specific collagen heterotrimers. Nat Commun 3:1087. doi:10.1038/ncomms2084
100. Privett HK, Kiss G, Lee TM, Blomberg R, Chica RA, Thomas LM, Hilvert D, Houk KN, Mayo SL (2012) Iterative approach to computational enzyme design. Proc Natl Acad Sci U S A 109(10):3790–3795. doi:10.1073/pnas.1118082108
101. Blomberg R, Kries H, Pinkas DM, Mittl PR, Grutter MG, Privett HK, Mayo SL, Hilvert D (2013) Precision is essential for efficient catalysis in an evolved Kemp eliminase. Nature 503(7476):418–421. doi:10.1038/nature12623
102. Grzyb J, Xu F, Nanda V, Luczkowska R, Reijerse E, Lubitz W, Noy D (2012) Empirical and computational design of iron-sulfur cluster proteins. Biochim Biophys Acta 1817 (8):1256–1262. doi:10.1016/j.bbabio.2012.02.001
103. Lanci CJ, MacDermaid CM, Kang SG, Acharya R, North B, Yang X, Qiu XJ, DeGrado WF, Saven JG (2012) Computational design of a protein crystal. Proc Natl Acad Sci U S A 109(19):7304–7309. doi:10.1073/pnas.1112595109
104. Reig AJ, Pires MM, Snyder RA, Wu Y, Jo H, Kulp DW, Butch SE, Calhoun JR, Szyperski T, Solomon EI, DeGrado WF (2012) Alteration of the oxygen-dependent reactivity of de novo Due Ferri proteins. Nat Chem 4 (11):900–906. doi:10.1038/nchem.1454
105. Dagliyan O, Shirvanyants D, Karginov AV, Ding F, Fee L, Chandrasekaran SN, Freisinger CM, Smolen GA, Huttenlocher A, Hahn KM, Dokholyan NV (2013) Rational design of a ligand-controlled protein conformational switch. Proc Natl Acad Sci U S A 110(17):6800–6804. doi:10.1073/pnas.1218319110
106. Fry HC, Lehmann A, Sinks LE, Asselberghs I, Tronin A, Krishnan V, Blasie JK, Clays K, DeGrado WF, Saven JG, Therien MJ (2013) Computational de novo design and characterization of a protein that selectively binds a highly hyperpolarizable abiological chromophore. J Am Chem Soc 135 (37):13914–13926. doi:10.1021/ja4067404
107. Perez-Aguilar JM, Xi J, Matsunaga F, Cui X, Selling B, Saven JG, Liu R (2013) A computationally designed water-soluble variant of a G-protein-coupled receptor: the human mu opioid receptor. PLoS One 8(6), e66009. doi:10.1371/journal.pone.0066009
108. Lerner MB, Matsunaga F, Han GH, Hong SJ, Xi J, Crook A, Perez-Aguilar JM, Park YW, Saven JG, Liu R, Johnson AT (2014) Scalable production of highly sensitive nanosensors based on graphene functionalized with a designed G protein-coupled receptor. Nano Lett 14(5):2709–2714. doi:10.1021/nl5006349
109. Procko E, Hedman R, Hamilton K, Seetharaman J, Fleishman SJ, Su M, Aramini J, Kornhaber G, Hunt JF, Tong L, Montelione GT, Baker D (2013) Computational design of a protein-based enzyme inhibitor. J Mol Biol 425(18):3563–3575. doi:10.1016/j.jmb.2013.06.035
110. Tinberg CE, Khare SD, Dou J, Doyle L, Nelson JW, Schena A, Jankowski W, Kalodimos CG, Johnsson K, Stoddard BL, Baker D (2013) Computational design of ligand-binding proteins with high affinity and selectivity. Nature 501(7466):212–216. doi:10.1038/nature12443
111. Strauch EM, Fleishman SJ, Baker D (2014) Computational design of a pH-sensitive IgG binding protein. Proc Natl Acad Sci U S A 111(2):675–680. doi:10.1073/pnas.1313605111
112. Xiong P, Wang M, Zhou X, Zhang T, Zhang J, Chen Q, Liu H (2014) Protein design with a comprehensive statistical energy function and boosted by experimental selection for foldability. Nat Commun 5:5330. doi:10.1038/ncomms6330
113. Huang PS, Oberdorfer G, Xu C, Pei XY, Nannenga BL, Rogers JM, DiMaio F, Gonen T, Luisi B, Baker D (2014) High thermodynamic stability of parametrically designed helical bundles. Science 346 (6208):481–485. doi:10.1126/science.1257481
114. Thomson AR, Wood CW, Burton AJ, Bartlett GJ, Sessions RB, Brady RL, Woolfson DN (2014) Computational design of water-soluble alpha-helical barrels. Science 346 (6208):485–488. doi:10.1126/science.1257452

115. Negron C, Keating AE (2014) A set of computationally designed orthogonal antiparallel homodimers that expands the synthetic coiled-coil toolkit. J Am Chem Soc 136 (47):16544–16556. doi:10.1021/ja507847t
116. Correia BE, Bates JT, Loomis RJ, Baneyx G, Carrico C, Jardine JG, Rupert P, Correnti C, Kalyuzhniy O, Vittal V, Connell MJ, Stevens E, Schroeter A, Chen M, Macpherson S, Serra AM, Adachi Y, Holmes MA, Li Y, Klevit RE, Graham BS, Wyatt RT, Baker D, Strong RK, Crowe JE Jr, Johnson PR, Schief WR (2014) Proof of principle for epitope-focused vaccine design. Nature 507(7491):201–206. doi:10.1038/nature12966
117. Joh NH, Wang T, Bhate MP, Acharya R, Wu Y, Grabe M, Hong M, Grigoryan G, DeGrado WF (2014) De novo design of a transmembrane Zn(2)(+)-transporting four-helix bundle. Science 346(6216):1520–1524. doi:10.1126/science.1261172
118. King C, Garza EN, Mazor R, Linehan JL, Pastan I, Pepper M, Baker D (2014) Removing T-cell epitopes with computational protein design. Proc Natl Acad Sci U S A 111 (23):8577–8582. doi:10.1073/pnas.1321126111
119. Voet AR, Noguchi H, Addy C, Simoncini D, Terada D, Unzai S, Park SY, Zhang KY, Tame JR (2014) Computational design of a self-assembling symmetrical beta-propeller protein. Proc Natl Acad Sci U S A 111 (42):15102–15107. doi:10.1073/pnas.1412768111
120. Ramisch S, Weininger U, Martinsson J, Akke M, Andre I (2014) Computational design of a leucine-rich repeat protein with a predefined geometry. Proc Natl Acad Sci U S A 111 (50):17875–17880. doi:10.1073/pnas.1413638111
121. Samish I, MacDermaid CM, Perez-Aguilar JM, Saven JG (2011) Theoretical and computational protein design. Annu Rev Phys Chem 62:129–149. doi:10.1146/annurev-physchem-032210-103509
122. Pantazes RJ, Grisewood MJ, Maranas CD (2011) Recent advances in computational protein design. Curr Opin Struct Biol 21 (4):467–472. doi:10.1016/j.sbi.2011.04.005
123. O'Neil KT, DeGrado WF (1985) A predicted structure of calmodulin suggests an electrostatic basis for its function. Proc Natl Acad Sci U S A 82(15):4954–4958
124. Weiner SJ, Kollman PA, Case DA, Singh UC, Ghio C, Alagona G, Profeta S, Weiner P (1984) A new force field for molecular mechanical simulation of nucleic acids and proteins. J Am Chem Soc 106(3):765–784
125. Hellinga HW, Richards FM (1991) Construction of new ligand binding sites in proteins of known structure. I. Computer-aided modeling of sites with pre-defined geometry. J Mol Biol 222(3):763–785
126. Ponder JW, Richards FM (1987) Tertiary templates for proteins. Use of packing criteria in the enumeration of allowed sequences for different structural classes. J Mol Biol 193 (4):775–791
127. Bryson JW, Betz SF, Lu HS, Suich DJ, Zhou HX, O'Neil KT, DeGrado WF (1995) Protein design: a hierarchic approach. Science 270 (5238):935–941
128. Raha K, Wollacott AM, Italia MJ, Desjarlais JR (2000) Prediction of amino acid sequence from structure. Protein Sci 9(6):1106–1119. doi:10.1110/ps.9.6.1106
129. Desmet J, De Maeyer M, Hazes B, Lasters I (1992) The dead-end elimination theorem and its use in protein side-chain positioning. Nature 356(6369):539–542
130. Samish I (2009) Search and sampling in structural bioinformatics. In: Bourne P, Gu J (eds) Structural bioinformatics. Wiley, Hoboken, NJ, pp 207–236
131. Dunbrack RL Jr, Karplus M (1993) Backbone-dependent rotamer library for proteins. Application to side-chain prediction. J Mol Biol 230(2):543–574. doi:10.1006/jmbi.1993.1170
132. Tuffery P, Etchebest C, Hazout S, Lavery R (1991) A new approach to the rapid determination of protein side chain conformations. J Biomol Struct Dyn 8(6):1267–1289. doi:10.1080/07391102.1991.10507882
133. Bowers PM, Strauss CE, Baker D (2000) De novo protein structure determination using sparse NMR data. J Biomol NMR 18 (4):311–318
134. Kuhlman B, Baker D (2000) Native protein sequences are close to optimal for their structures. Proc Natl Acad Sci U S A 97 (19):10383–10388
135. Fisinger S, Serrano L, Lacroix E (2001) Computational estimation of specific side chain interaction energies in alpha helices. Protein Sci 10(4):809–818. doi:10.1110/ps.34901
136. Kortemme T, Ramirez-Alvarado M, Serrano L (1998) Design of a 20-amino acid, three-stranded beta-sheet protein. Science 281 (5374):253–256
137. Debler EW, Ito S, Seebeck FP, Heine A, Hilvert D, Wilson IA (2005) Structural origins of

efficient proton abstraction from carbon by a catalytic antibody. Proc Natl Acad Sci U S A 102(14):4984–4989. doi:10.1073/pnas.0409207102

138. Khersonsky O, Rothlisberger D, Wollacott AM, Murphy P, Dym O, Albeck S, Kiss G, Houk KN, Baker D, Tawfik DS (2011) Optimization of the in-silico-designed kemp eliminase KE70 by computational design and directed evolution. J Mol Biol 407 (3):391–412. doi:10.1016/j.jmb.2011.01.041
139. Khersonsky O, Rothlisberger D, Dym O, Albeck S, Jackson CJ, Baker D, Tawfik DS (2010) Evolutionary optimization of computationally designed enzymes: Kemp eliminases of the KE07 series. J Mol Biol 396 (4):1025–1042. doi:10.1016/j.jmb.2009.12.031
140. Khersonsky O, Kiss G, Rothlisberger D, Dym O, Albeck S, Houk KN, Baker D, Tawfik DS (2012) Bridging the gaps in design methodologies by evolutionary optimization of the stability and proficiency of designed Kemp eliminase KE59. Proc Natl Acad Sci U S A 109(26):10358–10363. doi:10.1073/pnas.1121063109
141. Dahiyat BI, Mayo SL (1996) Protein design automation. Protein Sci 5(5):895–903. doi:10.1002/pro.5560050511
142. Kortemme T, Baker D (2002) A simple physical model for binding energy hot spots in protein-protein complexes. Proc Natl Acad Sci U S A 99(22):14116–14121. doi:10.1073/pnas.202485799
143. Kono H, Saven JG (2001) Statistical theory for protein combinatorial libraries. Packing interactions, backbone flexibility, and the sequence variability of a main-chain structure. J Mol Biol 306(3):607–628. doi:10.1006/jmbi.2000.4422
144. Lombardi A, Summa CM, Geremia S, Randaccio L, Pavone V, DeGrado WF (2000) Retrostructural analysis of metalloproteins: application to the design of a minimal model for diiron proteins. Proc Natl Acad Sci U S A 97(12):6298–6305
145. Dunbrack RL Jr, Cohen FE (1997) Bayesian statistical analysis of protein side-chain rotamer preferences. Protein Sci 6 (8):1661–1681. doi:10.1002/pro.5560060807
146. Fu X, Kono H, Saven JG (2003) Probabilistic approach to the design of symmetric protein quaternary structures. Protein Eng 16 (12):971–977. doi:10.1093/protein/gzg132
147. Zhang SQ, Kulp DW, Schramm CA, Mravic M, Samish I, DeGrado WF (2015) The membrane- and soluble-protein helix-helix interactome: similar geometry via different interactions. Structure 23(3):527–541. doi:10.1016/j.str.2015.01.009
148. Smith MD, Zanghellini A, Grabs-Rothlisberger D (2014) Computational design of novel enzymes without cofactors. Methods Mol Biol 1216:197–210. doi:10.1007/978-1-4939-1486-9_10
149. Zanghellini A, Jiang L, Wollacott AM, Cheng G, Meiler J, Althoff EA, Rothlisberger D, Baker D (2006) New algorithms and an in silico benchmark for computational enzyme design. Protein Sci 15(12):2785–2794. doi:10.1110/ps.062353106
150. Richter F, Leaver-Fay A, Khare SD, Bjelic S, Baker D (2011) De novo enzyme design using Rosetta3. PLoS One 6(5), e19230. doi:10.1371/journal.pone.0019230
151. Frushicheva MP, Cao J, Chu ZT, Warshel A (2010) Exploring challenges in rational enzyme design by simulating the catalysis in artificial kemp eliminase. Proc Natl Acad Sci U S A 107(39):16869–16874. doi:10.1073/pnas.1010381107
152. Georgiev I, Lilien RH, Donald BR (2008) The minimized dead-end elimination criterion and its application to protein redesign in a hybrid scoring and search algorithm for computing partition functions over molecular ensembles. J Comput Chem 29 (10):1527–1542. doi:10.1002/jcc.20909
153. Schneidman-Duhovny D, Inbar Y, Polak V, Shatsky M, Halperin I, Benyamini H, Barzilai A, Dror O, Haspel N, Nussinov R, Wolfson HJ (2003) Taking geometry to its edge: fast unbound rigid (and hinge-bent) docking. Proteins 52(1):107–112. doi:10.1002/prot.10397
154. Fiser A, Do RK, Sali A (2000) Modeling of loops in protein structures. Protein Sci 9 (9):1753–1773. doi:10.1110/ps.9.9.1753
155. Ofek G, Guenaga FJ, Schief WR, Skinner J, Baker D, Wyatt R, Kwong PD (2010) Elicitation of structure-specific antibodies by epitope scaffolds. Proc Natl Acad Sci U S A 107 (42):17880–17887. doi:10.1073/pnas.1004728107
156. Canutescu AA, Dunbrack RL Jr (2003) Cyclic coordinate descent: a robotics algorithm for protein loop closure. Protein Sci 12 (5):963–972. doi:10.1110/ps.0242703
157. Smith CA, Kortemme T (2011) Predicting the tolerated sequences for proteins and protein interfaces using RosettaBackrub flexible

backbone design. PLoS One 6(7), e20451. doi:10.1371/journal.pone.0020451

158. Mandell DJ, Coutsias EA, Kortemme T (2009) Sub-angstrom accuracy in protein loop reconstruction by robotics-inspired conformational sampling. Nat Methods 6 (8):551–552. doi:10.1038/nmeth0809-551
159. Roy A, Kucukural A, Zhang Y (2010) I-TASSER: a unified platform for automated protein structure and function prediction. Nat Protoc 5(4):725–738. doi:10.1038/nprot.2010.5
160. Summa CM, Levitt M, Degrado WF (2005) An atomic environment potential for use in protein structure prediction. J Mol Biol 352 (4):986–1001. doi:10.1016/j.jmb.2005.07.054
161. Antonkine ML, Maes EM, Czernuszewicz RS, Breitenstein C, Bill E, Falzone CJ, Balasubramanian R, Lubner C, Bryant DA, Golbeck JH (2007) Chemical rescue of a site-modified ligand to a [4Fe-4S] cluster in PsaC, a bacterial-like dicluster ferredoxin bound to Photosystem I. Biochim Biophys Acta 1767(6):712–724. doi:10.1016/j.bbabio.2007.02.003
162. Kulp DW, Subramaniam S, Donald JE, Hannigan BT, Mueller BK, Grigoryan G, Senes A (2012) Structural informatics, modeling, and design with an open-source Molecular Software Library (MSL). J Comput Chem 33 (20):1645–1661. doi:10.1002/jcc.22968
163. Yang Y, Zhou Y (2008) Ab initio folding of terminal segments with secondary structures reveals the fine difference between two closely related all-atom statistical energy functions. Protein Sci 17(7):1212–1219. doi:10.1110/ps.033480.107
164. Schramm CA, Hannigan BT, Donald JE, Keasar C, Saven JG, Degrado WF, Samish I (2012) Knowledge-based potential for positioning membrane-associated structures and assessing residue-specific energetic contributions. Structure 20(5):924–935. doi:10.1016/j.str.2012.03.016
165. Grigoryan G (2013) Absolute free energies of biomolecules from unperturbed ensembles. J Comput Chem 34(31):2726–2741. doi:10.1002/jcc.23448
166. Dwyer MA, Looger LL, Hellinga HW (2008) Retraction. Science 319(5863):569. doi:10.1126/science.319.5863.569b
167. Dwyer MA, Looger LL, Hellinga HW (2004) Computational design of a biologically active enzyme. Science 304(5679):1967–1971. doi:10.1126/science.1098432
168. Allert M, Dwyer MA, Hellinga HW (2007) Local encoding of computationally designed enzyme activity. J Mol Biol 366(3):945–953. doi:10.1016/j.jmb.2006.12.002
169. (2008) Negative results. Nature 453 (7193):258. doi:10.1038/453258b
170. Hayden EC (2009) Key protein-design papers challenged. Nature 461(7266):859. doi:10.1038/461859a
171. Looger LL, Dwyer MA, Smith JJ, Hellinga HW (2003) Computational design of receptor and sensor proteins with novel functions. Nature 423(6936):185–190. doi:10.1038/nature01556
172. Allert M, Rizk SS, Looger LL, Hellinga HW (2004) Computational design of receptors for an organophosphate surrogate of the nerve agent soman. Proc Natl Acad Sci U S A 101 (21):7907–7912. doi:10.1073/pnas.0401309101
173. Hellinga HW (2008) In the wake of two retractions, a request for investigation. Nature 454(7203):397. doi:10.1038/454397b
174. Altamirano MM, Blackburn JM, Aguayo C, Fersht AR (2002) Retraction. Directed evolution of new catalytic activity using the alpha/beta-barrel scaffold. Nature 417 (6887):468. doi:10.1038/417468a
175. Altamirano MM, Blackburn JM, Aguayo C, Fersht AR (2000) Directed evolution of new catalytic activity using the alpha/beta-barrel scaffold. Nature 403(6770):617–622. doi:10.1038/35001001
176. Feldmeier K, Hocker B (2013) Computational protein design of ligand binding and catalysis. Curr Opin Chem Biol 17 (6):929–933
177. Wijma HJ, Janssen DB (2013) Computational design gains momentum in enzyme catalysis engineering. FEBS J 280 (13):2948–2960. doi:10.1111/febs.12324
178. Khare SD, Fleishman SJ (2013) Emerging themes in the computational design of novel enzymes and protein-protein interfaces. FEBS Lett 587(8):1147–1154. doi:10.1016/j.febslet.2012.12.009
179. Davey JA, Chica RA (2012) Multistate approaches in computational protein design. Protein Sci 21(9):1241–1252. doi:10.1002/pro.2128
180. Tiwari MK, Singh R, Singh RK, Kim IW, Lee JK (2012) Computational approaches for rational design of proteins with novel functionalities. Comput Struct Biotechnol J 2, e201209002. doi:10.5936/csbj.201209002
181. Senes A (2011) Computational design of membrane proteins. Curr Opin Struct Biol 21(4):460–466. doi:10.1016/j.sbi.2011.06.004

182. Verschueren E, Vanhee P, van der Sloot AM, Serrano L, Rousseau F, Schymkowitz J (2011) Protein design with fragment databases. Curr Opin Struct Biol 21 (4):452–459. doi:10.1016/j.sbi.2011.05.002

183. Samish I, Bourne PE, Najmanovich RJ (2015) Achievements and challenges in structural bioinformatics and computational biophysics. Bioinformatics 31(1):146–150. doi:10.1093/bioinformatics/btu769

184. Rosenberg M, Goldblum A (2006) Computational protein design: a novel path to future protein drugs. Curr Pharm Des 12(31):3973–3997

185. Desjarlais JR, Lazar GA (2003) Negative design for improved therapeutic proteins. Trends Biotechnol 21(10):425–427. doi:10.1016/S0167-7799(03)00205-1

186. Moult J, Pedersen JT, Judson R, Fidelis K (1995) A large-scale experiment to assess protein structure prediction methods. Proteins 23(3):ii–v. doi:10.1002/prot.340230303

Chapter 3

Production of Computationally Designed Small Soluble- and Membrane-Proteins: Cloning, Expression, and Purification

Barsa Tripathy and Rudresh Acharya

Abstract

This book chapter focuses on expression and purification of computationally designed small soluble proteins and membrane proteins that are ordinarily difficult to express in good amounts for experiments. Over-expression of such proteins can be achieved by using the solubility tag such as maltose binding protein (MBP), Thioredoxin (Trx), and Gultathione-S-transferase (GST) fused to the protein of interest. Here, we describe and provide the protocols for cloning, expression and purification of such proteins using the solubility tag.

Key words Cloning, Protein expression and purification, Designed proteins, Small proteins, Solubility tags

1 Introduction

Insights into protein chemistry and advancement in the field of computational biology have led to evolution of the field of protein designing. The last decade has witnessed many impressive designed, in silico proteins: water-soluble proteins [1–4], water-soluble analogue of membrane protein [5], single pass membrane proteins [6, 7]. These proteins are often small (~24–100 amino acid residues) with directed simple functions. In the future it is likely that computational design will not only advance the design of soluble proteins, but also design of membrane proteins, that will venture into the avenue of multiple span single chain proteins [~100 residues] to perform complex functions. Due to the difficulties in over expressing such designed proteins, the experimental characterizations become challenging. Even, naturally occurring small proteins will encounter the same fate when tried to express heterologously. These proteins either prove toxic to the cells or go into inclusion bodies due to over expression and aggregation. Such proteins when purified result in meager yields to be used for

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_3,

characterizations. This forms the bottleneck in the transition of in silico work into in vitro work. To overcome these challenges, one can screen for suitable host bacterial strain [8, 9], and further, the protein of interest can be over-expressed by fusing with solubility tags such as Maltose binding protein (MBP), Thioredoxin (TrX), and Glutathione-S-transferase (GST) as well as solubility enhancing tags including Small ubiquitin-like modifier (SUMO) protein and Halo Tag. Several of these tags were successful in producing the protein of interest at good yields, and the detailed usage of the tags has been extensively reported in several articles [10–15]. In this chapter, we will discuss the generalized strategies, and provide protocols that can be used successfully to obtain good quantities of small sized proteins by using the solubility tags.

2 Materials

All the solutions were prepared using Autoclaved Milli-Q water. All necessary precautions were taken to avoid contamination.

2.1 Cloning

2.1.1 Restriction Digestion

1. Insert.
2. Expression Vector (pMALc5X for MBP tag and pET42a for GST tag).
3. Restriction Enzymes.
4. Calf Intestinal Phosphatase (CIP).
5. Water Bath.

2.1.2 For DNA Gel Electrophoresis and Gel Elution

1. Agarose
2. 1X TAE: 40 mM Tris-acetate, 1 mM EDTA pH 8.0.

   First make 1000 ml of stock of 50X TAE buffer as follows: Make a concentrated (50X) stock solution of TAE by weighing out 242 g Tris base (FW = 121.14) and dissolving in approximately 750 mL Milli-Q water. Carefully add 57.1 ml glacial acid and 100 mL of 0.5 M EDTA (pH 8.0) and adjust the solution to a final volume of 1 L. This stock solution can be stored at room temperature. The pH of this buffer is not adjusted and should be about 8.5.

   Now to make 1000 ml of 1X TAE from 50X TAE stock solutions, 20 ml stock of 50X is taken and 980 ml of Milli-Q water is added to it.
3. 10 % Ethidium Bromide
4. Gel Elution (Commercially available kit)

2.1.3 Ligation

1. T4 Ligase
2. Thermocycler Machine

2.2 Competent Cell Preparation

1. LEMO 21 cell glycerol stock (New England Biolabs) (for alternative competent cells *see* **Note 2**).
2. 100 ml LB Agar
   To make 100 ml of LB Agar, weigh 4 g of LB Agar powder and dissolve in 60 ml of Milli-Q water. Adjust volume to 100 ml and then autoclave it. After autoclave is complete, pour 25 ml into each of the petri plates, and let them solidify. Store plates at 4 °C for further use.
3. 50 ml LB Broth
   To make 50 ml of LB Broth, weigh 1.25 g of LB Broth powder and dissolve it in 35 ml of Milli-Q water. Adjust volume to 50 ml and then autoclave it.
4. 200 ml LB Broth
5. 0.1 M $CaCl_2$
   First make a stock of 1 M $CaCl_2$ by weighing 14.7 g of $CaCl_2$ (FW = 147) and dissolve it in 80 ml of autoclaved Milli-Q water. Adjust volume to 100 ml.
   To make 0.1 M of $CaCl_2$, take 5 ml of 1 M $CaCl_2$ add 45 ml of autoclaved Milli-Q water. Filter sterilize it and save it in a sterile bottle and store at 4 °C. This solution should be prepared freshly every time a batch of competent cells has to be prepared.
6. 0.1 M $CaCl_2$+ 10 % (v/v) Glycerol
   From the stock of 1 M $CaCl_2$ take 5 ml and add 5 ml of Glycerol to it. Adjust volume to 50 ml by adding autoclaved Milli-Q water. Filter sterilize it and save it in a sterile bottle and store at 4 °C. This solution should be prepared freshly every time a batch of competent cells has to be prepared.
7. Liquid Nitrogen.
8. Shaker Incubator.

2.3 Transformation

1. Selection Plates.
   Selection plates contain LB Agar and the required antibiotic. To make selection plates, prepare and autoclave LB Agar. After autoclave, when the agar cools down to an extent that the temperature of the flask is bearable by cheek (cheek test), add antibiotic to it as per prescribed concentration, mix well and pour 25 ml into each plate. After it solidifies, store plates at 4 °C for further use.
2. L Shaped Spreader.
3. 50 ml LB Broth.

2.4 Screening

1. Taq Polymerase.
2. PCR Master Mix.
3. Insert Specific Forward and Reverse Primers.
4. Plasmid isolation (commercially available kit).

2.5 Protein Expression

1. 200 ml LB Broth.
2. Specific Antibiotics.
3. 0.5 M Rhamnose.
4. 1 M Isopropyl β-D-1-thiogalactopyranoside (IPTG).
5. UV–Vis Spectrophotometer.

2.6 SDS PAGE

1. 30 % Acrylamide/Bis-acrylamide Solution.
2. Tris–Cl Buffer, pH 8.8 : 1.5 M Tris–HCl, pH 8.8.
3. Tris–Cl Buffer, pH 6.8 : 0.5 M Tris–HCl, pH 6.8.
4. Sodium Dodecyl Sulfate (SDS) : 10 % (w/v) in water.
5. Ammonium per sulfate (APS) : 10 % (w/v) in water.
6. *N*,*N*,*N*′,*N*′-tetramethyl-ethylenediamine (TEMED).
7. SDS PAGE Running Buffer : 0.025 M Tris pH 8.3, 0.192 M Glycine, 0.1 % SDS.
8. Gel Loading Buffer 5X : 0.3 M Tris–HCl pH 6.8, 0.1 % (w/v) Bromophenol Blue, 10 % (w/v) SDS, 25 % (v/v) β-Mercaptoethanol, 45 % (v/v) Glycerol.
9. Gel Staining Solution: 0.25 % (w/v) Coomassie brilliant blue, 10ml Acetic acid, 40ml water, 50ml methanol.
10. Gel De-staining solution: 10 ml Acetic acid, 40 ml Methanol, 50 ml Water.

2.7 Cell Lysis

1. 5X Native Purification Buffer: 250 mM NaH_2PO_4 pH 8.0, 0.5 M NaCl.

   Prepare 1 L of 5X Native Purification Buffer. To 900 ml of autoclaved Milli-Q water, add 34.5 g of NaH_2PO_4, and 29.2 g of NaCl. Adjust pH to 8 and bring final volume to 1 L. Further to make 100 ml of 1X Native purification buffer, 80 ml of Milli-Q water is added to 20 ml of 5X Native Purification buffer.
2. EDTA/EGTA free protease Inhibitor Cocktail.
3. 100 mM PMSF.

   To prepare 100 mM PMSF, weigh 174.19 mg of PMSF (FW = 174.19) and dissolve in 10 ml of isopropanol. Keep inverting and tapping till all the PMSF crystals completely dissolve.
4. Lysis Buffer

   The basic composition of the lysis buffers used for different tags is the same, which is 1X Native Purification Buffer. For both MBP and GST tag, the same composition is used.
5. Tip Sonicator.

2.8 Protein Purification

1. 10 ml purification column.
2. Wash Buffer.
   The composition of wash buffer varies with the solubility tag used. For GST tag, 50 mM Tris pH 7.4, 0.25 M NaCl, 1 mM EDTA is used. For MBP tag, the 1X native purification buffer can be used as wash buffer.
3. Elution Buffer.
   Elution buffer composition also varies along with the solubility tag. For GST tag, to the wash buffer 33 mM of reduced glutathione is added. To prepare GST elution buffer, weigh 20.28 mg (FW = 307.32) of reduced glutathione and add to 2 ml of wash buffer.
   For MBP tag, 10 mM Maltose is required in the Elution buffer. To prepare MBP elution buffer, add 7.2 mg of Maltose to 2 ml of wash buffer for MBP Tag.
4. Reduced Glutathione solution.
5. Maltose.

3 Methods

3.1 Reverse Translation

De novo designed proteins, or proteins for which it is very difficult to get the full length DNA, reverse translation serves as an extremely helpful tool. The amino acid sequence of the protein can be reverse translated to the corresponding DNA sequence by using the web-based application by Helix Systems called "DNA-Works" [16]. The DNA sequence can be optimized for codon bias so as to obtain high expression. DNAWorks also generates oligos corresponding to the DNA sequence, which when assembled through the PCR-based method can give rise to the complete DNA sequence.

3.2 Construct Design

The simplest of the construct design would be the insert (obtained by using DNAWorks by Helix Systems) flanked at its 5′ and 3′ ends by two different restriction enzyme sites, which would assist in directional cloning. Those restriction enzyme sites should be present in the multiple cloning site (MCS) of the vector as well. A few (~6) bases should be added at both the 5′ and 3′ ends of the construct to increase the efficiency of cleavage by the restriction enzymes. Most of the expression vectors carry a protease site that allows the protein of interest to be cleaved from the solubility tag. There is a chance that after the cleavage with protease, some of the amino acid residues of the cleavage site remain attached to the protein of interest, which might be undesirable. To avoid this problem, incorporating an additional protease site such as that of Factor Xa (cleaves at the C terminal of the protease site, and leaves

no stray residues attached to the protein of interest) at the 5′ end of the insert results in a clean release of only the protein of interest. Additionally, a hexa-histidine (6×His) tag can also be inserted in between the 5′ end restriction site and protease site to assist in affinity-based protein purification after solubility tag cleavage.

3.3 Cloning

3.3.1 Restriction Digestion

The designed construct (herefrom referred to as insert) is doubly digested using two different enzymes. Similarly, the expression vector of our choice is doubly digested using the same enzymes that were used for digesting the insert.

Recipe for restriction digestion

	Insert	Expression Vector
Vector	1 μg	1 μg
Buffer	1X	1X
Enzyme 1	1 Unit	2–5 Units
Enzyme 2	1 Unit	2–5 Units
Water	Volume adjust	Volume adjust
Total	30 μL	30 μL

1. The digestion mixes are left at 37 °C in a water bath for 3.5 hours.
2. After completion of 3.5 hours, a 1–2 unit of Calf intestinal phosphate is added to the expression vector (*see* **Note 1**). It is incubated at 37 °C on a water bath for another 30 min. The insert digestion mix is left undisturbed.
3. After restriction digestion is complete the digestion mixes are run on a 1 % agarose gel.
4. The bands corresponding to insert and linearized expression vector are eluted from the gel.
5. The concentrations are measured using a spectrophotometer. These concentrations are used in calculations required in the next step of ligation.

3.3.2 Ligation

The following formula is used to calculate the amount of insert required given the amount of vector (100 ng).

$$\text{Amount of Insert} = \frac{\text{ng of vector} \times \text{kb size of insert}}{\text{kb size of vector}} \times \frac{\text{molar ratio insert}}{\text{Vector}}.$$

The following recipe is used for ligation when using T4 ligase.

	Insert + vector	Vector (−ve control)
Vector	100 ng	100 ng
Insert	As per formula	–
Ligation Buffer	1X	1X
T4 Ligase	1 Unit	1 Unit
Water	Volume adjustment	Volume adjustment
Total	10 μL	10 μL

The ligation mix is kept at room temperature (25 °C) for 15 min.

3.4 Competent Cell Preparation

Competent cell preparation is started 4 days prior to cloning experiment.

DAY1: Required *E. coli* strains are streaked on fresh LB Agar plates and incubated overnight at 37 °C.

DAY2: Preparation of 0.1 M $CaCl_2$ and 0.1 M $CaCl_2$ + 10 % glycerol solutions.

Setting up of 5 ml primary culture (picking up a single colony from plate and inoculating the broth) and overnight incubation at 37 °C.

DAY3:

1. Secondary culture is set up (200 mL LB broth in 1 L flask) by adding 1 % (v/v) of primary culture.
2. Incubation at 37 °C with shaking till OD reaches 0.4.
3. Cells are harvested by pouring culture into four 50 mL falcon tubes and centrifuging at 1500 × *g* at 4 °C for 10 min.
4. Supernatant is discarded and pellet is washed with 5 mL of pre-chilled 0.1 M $CaCl_2$ per tube, till the pellet resuspends. Centrifugation is repeated as above. The supernatant is carefully discarded.
5. The subsequent steps are done on ice. 5 ml of pre-chilled 0.1 M $CaCl_2$ is added to the pellet and the cells are suspended and left to incubate for 40 min. Centrifugation is repeated as above but for only 5 min.
6. Supernatant is discarded and 2 mL of 0.1 M $CaCl_2$ + 10 % glycerol solution is added to each tube. The cells are suspended by swirling the contents of the tubes and are finally poured into one.
7. The tube is allowed to sit on ice or kept in cold room overnight.

DAY4: 100 (in numbers) of 1.5 mL microcentrifuge tubes are pre-chilled on ice. The cells in the falcon tube are resuspended and 100 μL is aliquot into each microcentrifuge tube. All the tubes are flash frozen and stored at −80 °C for future usage.

3.5 Transformation and Screening

1. Two microcentrifuge tubes containing 100 μL of competent cells (DH5α) are taken out from −80 °C and kept in ice for 10 min.
2. All the 10 μL of the ligation mix − insert+vector and only vector are added to the separate tubes containing competent cells.
3. The cells are then left to equilibrate with the DNA for 20–30 min.
4. The cells are then given heat shock at 42 °C for 60 s by putting the tubes in water bath set at the said temperature.
5. The tubes are taken out and kept in ice for 2 min and then 1 ml of LB broth is added to both tubes.
6. The tubes are then kept in shaker incubator at 37 °C for 1 h.
7. After 1 h, the cells are centrifuged, 900 μL of the supernatant is discarded, and the pellet is resuspended in the remaining supernatant and is plated on appropriate selection plates (depending upon antibiotic resistant gene carried by the expression vector) and left overnight at 37 °C.
8. Screening

The plates are checked for the appearance of colonies on the plates.

1. A small amount of inoculum from each of these colonies (from the insert + vector plate) is then streaked into another plate.
2. Colony PCR is performed to verify the ligation of the insert using primers specific for that insert.

Protocol for colony PCR is as follows:

Small amount of bacterial inoculum from the colonies are smeared into PCR tubes (depending on number of colonies to be screened) and to each of the tubes the following recipe is added.

2X PCR Mix	1X
Forward and reverse primer mix [10 μM]	1 μM
Taq polymerase	1–3 Units
Water	Adjust volume
Total	10 μL

The colonies that give positive results are further validated by isolating the plasmids and performing restriction digestion using the same restriction enzymes used for cloning (following the restriction digestion protocol 3.3.1).

DNA sequencing is ultimately done to confirm the insert sequence.

3.6 Protein Expression

1. Transform (*see* **step 4** of **section 3.5**) the plasmid carrying the insert into LEMO21 cells (New England Biolabs) (*see* **Note 2**). The cells are plated and the plate is left for overnight incubation at 37 °C. A 20ml primary culture is set up the following day.
2. A 1 L secondary culture is set up the next morning. To the culture, antibiotics (*see* **Note 2**) are added along with 2 mM Rhamnose (*see* **Note 3**). 1% (v/v) primary culture is added to the secondary culture and it is kept in a shaker incubator till OD reaches 0.4–0.6.
3. At this OD, induction is done with 1 M IPTG such that the working concentration of IPTG is 0.4 mM.
4. Post induction, the culture is kept back in shaker incubator for 4 hours at 37 °C.
5. After incubation, the cells are harvested by centrifuging at 1500 × *g* for 5 min. The cell pellets are stored at −80 °C for further use.

3.7 Cell Lysis

1. 50 μL of EDTA/EGTA free Protease Inhibitor cocktail is added to the harvested cells after thawing.
2. The pellet is resuspended in 10 ml of Lysis Buffer containing 1 mM working PMSF. It is left to incubate at ice for 30 min.
3. At 40 % amplitude, and a 5 s ON and 10 s OFF pulse, sonication is done for approximately 5 min. 5 μL of the lysate is then stored to run on a SDS-PAGE gel later.
4. The remaining lysate is centrifuged at 15000 × *g* for 1 h to pellet down cellular debris. The pellet is stored at −20 °C, after taking out 5 μL of supernatant and a speck of pellet for running on a SDS-PAGE gel. Ideally, most of the protein should be present in the supernatant fraction.
5. However if the protein is still trapped in the pellet fraction a different strategy is used to solubilize it. To the pellet, 5 ml of Lysis buffer as mentioned earlier is added and the pellet is resuspended. Now sonication is repeated for 5 min. 0.3 % (v/v) SDS, 3 % (v/v) Triton X-100, and 30 mM CHAPS (v/v) are added (*see* **Note 4**) [17] and left for overnight incubation on a nutating mixer. The following morning the lysate is centrifuged at 15000 × *g* for 1 h again. 5 μL of supernatant and a speck of pellet is saved for running on a gel and the remaining is stored at −20 °C. This step should solubilize most of the protein trapped in the pellet.
6. After confirming the presence of protein in the supernatant fractions by running the samples on a 12 % acrylamide gel, protein purification is followed.

3.8 Protein Purification

3.8.1 Column Preparation

1. 1.5 ml of resin (50% ethanol slurry) (*see* **Note 5**) is pipetted into a 10 ml purification column [18]. The resin is allowed to settle down either by gravity or by centrifugation at low RPM ($<1500 \times g$).
2. The supernatant is allowed to flow through the column and to the resin 6 ml of water is added to wash the resin, by inverting and tapping the resin few times. The resin is allowed to settle and the supernatant is allowed to flow through the column.
3. 6 ml of lysis buffer is added next to equilibrate the resin. After the resin settles down the supernatant is allowed to flow through the column. This step is repeated twice. The column is now ready for use.

3.8.2 Binding, Washing, and Elution

1. All 10 ml of Supernatant containing the soluble protein is added to the resin and left for overnight binding on a nutating mixer at 4 °C (*see* **Note 6**).
2. The next day the column is fixed on a stand and the resin is allowed to settle. The supernatant is allowed to flow through the column and is collected in a separate tube. This flow through constitutes the unbound fraction. 5 μL of this unbound fraction is kept aside to run on a gel and the remaining is stored at −20 °C.
3. Next, to the column 8 ml of Wash buffer is added and the resin is allowed to settle down. The flow through collected is labeled as Wash fraction 1 and stored at −20 °C. 5 μL of this wash fraction 1 is kept aside to run on a gel. This process is repeated thrice.
4. Finally to elute the protein, 0.5 ml of Elution buffer is added to the resin and is allowed to stand for a minute. The eluted fraction is then collected in a 1.5 ml microcentrifuge tube separately labeled as Elute Fraction 1. Three more elute fractions are collected by following the same procedure.
5. All the collected fractions are run on a 12 % gel to check for presence of the protein of interest. The fractions containing the protein of interest are pooled together and concentrated using appropriate molecular weight cut off spin concentrator. Dialysis against an appropriate storage buffer is followed afterwards to get rid of the remaining glutathione/maltose. As per the need, the solubility tag could be cleaved by specific protease (Factor Xa) action and further protein of interest is purified using ion exchange chromatography/affinity chromatography (in case 6 × His tag is present in the construct *see* **section 3.2** Construct Design).

4 Notes

1. Calf Intestinal Phosphate helps to chew the phosphate overhangs thereby inhibiting self ligation of vectors and facilitating directional cloning of insert.
2. These cells are best suited to produce proteins that are toxic to the cell and cannot be expressed in BL21(DE3) cells. After transformation, these cells have to be plated on an agar plate containing two antibiotics - chloramphenicol (for pLEMO) and Ampicillin (for pMALc5X) / Kanamycin (for pET42a). In parallel, protein over-expression with other competent cells such as C41, C43, Rosetta could also be tried.
3. Rhamnose helps in tuning the protein expression and is specific for LEMO 21 cells.
4. Alternatively 10 % SDS and 10 % Triton X-100 can also be used to solubilize the protein trapped in the pellet.
5. For MBP tag Amylose resin is used and for GST tag, Glutathione resin is used.
6. When using GST tag, adding 1 mM of Dithiothreitol (DTT) to the lysate while it is kept for binding, increases specific binding of GST to column.

Acknowledgments

This work was funded by Ramalingaswami fellowship, Department of Biotechnology, India (R.A.) and National Institute of Science Education and Research start up funds. We would like to thank National Institute of Science Education and Research for providing the infrastructure and facilities.

References

1. Lombardi A, Summa CM, Geremia S, Randaccio L, Pavone V, DeGrado WF et al (2000) Retrostructural analysis of metalloproteins: application to the design of a minimal model for diiron proteins. Proc Natl Acad Sci U S A 97(12):6298–6305
2. Koga N, Tatsumi-Koga R, Liu G, Xiao R, Acton TB, Montelione GT, Baker D et al (2012) Principles for designing ideal protein structures. Nature 491(7423):222–227
3. Thomson AR, Wood CW, Burton AJ, Bartlett GJ, Sessions RB, Brady RL, Woolfson DN et al (2014) Computational design of water-soluble α-helical barrels. Science 346(6208):485–488
4. Bender GM, Lehmann A, Zou H, Cheng H, Fry HC, Engel D, Therien MJ, Blasie JK, Roder H, Saven JG, DeGrado WF et al (2007) De novo design of a single-chain diphenylporphyrin metalloprotein. J Am Chem Soc 129(35):10732–10740
5. Slovic AM, Kono H, Lear JD, Saven JG, DeGrado WF et al (2004) Computational design of water-soluble analogues of the potassium channel KcsA. Proc Natl Acad Sci U S A 101(7):1828–1833
6. Korendovych IV, Senes A, Kim YH, Lear JD, Fry HC, Therien MJ, Blasie JK, Walker FA, Degrado WF et al (2010) De novo design and

molecular assembly of a transmembrane diporphyrin-binding protein complex. J Am Chem Soc 132(44):15516–15518

7. Joh NH, Wang T, Bhate MP, Acharya R, Wu Y, Grabe M, Hong M, Grigoryan G, DeGrado WF et al (2014) De novo design of a transmembrane zn^{2+}-transporting four-helix bundle. Science 346(6216):1520–1524
8. Bernaudat F, Frelet-Barrand A, Pochon N, Dementin S, Hivin P et al (2011) Heterologous expression of membrane proteins: choosing the appropriate host. PLoS One 6(12): e29191. doi:10.1371/journal.pone.0029191
9. Gopal GJ, Kumar A (2013) Strategies for the production of recombinant protein in Escherichia coli. Protein J 32(6):419–425
10. Hu J, Qin H, Gao FP, Cross TA et al (2011) A systemic assessment of mature MBP in membrane protein production: over expression, membrane targeting and purification. Protein Expr Purif 80(1):34–40
11. Harper S, Speicher DW (2011) Purification of proteins fused to glutathione s-transferase. Methods Mol Biol 681:259–280
12. Terpe K (2003) Overview of tag protein fusions: from molecular and biochemical fundamentals to commercial systems. Appl Microbiol Biotechnol 60:523–533
13. Malhotra A (2009) Tagging for protein expression. Methods Enzymol 463:239–258
14. Young CL, Britton ZT, Robinson AS (2012) Recombinant protein expression and purification: a comprehensive review of affinity tags and microbial applications. Biotechnol J 5:620–634
15. Zhao X, Li G, Liang S (2013) Several affinity tags commonly used in chromatographic purification. J Anal Methods Chem 2013: 581093
16. Hoover DM, Lubkowski J (2002) DNAWorks: an automated method for designing oligonucleotides for PCR-based gene synthesis. Nucleic Acids Res 30(10):e43
17. Tao H, Liu W, Simmons BN, Harris HK, Cox TC, Massiah MA et al (2010) Purifying natively folded proteins from inclusion bodies using sarkosyl, Triton X-100, and CHAPS. Biotechniques 48(1):61–64
18. Invirogen Ni-NTA Purification system manual, Gräslund S, Nordlund P, Weigelt J, Hallberg BM et al (2008) Protein production and purification. Nat Methods 5(2):135–146

Chapter 4

Deterministic Search Methods for Computational Protein Design

Seydou Traoré, David Allouche, Isabelle André, Thomas Schiex, and Sophie Barbe

Abstract

One main challenge in Computational Protein Design (CPD) lies in the exploration of the amino-acid sequence space, while considering, to some extent, side chain flexibility. The exorbitant size of the search space urges for the development of efficient exact deterministic search methods enabling identification of low-energy sequence-conformation models, corresponding either to the global minimum energy conformation (GMEC) or an ensemble of guaranteed near-optimal solutions. In contrast to stochastic local search methods that are not guaranteed to find the GMEC, exact deterministic approaches always identify the GMEC and prove its optimality in finite but exponential worst-case time. After a brief overview on these two classes of methods, we discuss the grounds and merits of four deterministic methods that have been applied to solve CPD problems. These approaches are based either on the Dead-End-Elimination theorem combined with A^* algorithm (DEE/A^*), on Cost Function Networks algorithms (CFN), on Integer Linear Programming solvers (ILP) or on Markov Random Fields solvers (MRF). The way two of these methods (DEE/A^* and CFN) can be used in practice to identify low-energy sequence-conformation models starting from a pairwise decomposed energy matrix is detailed in this review.

Key words Exact combinatorial optimization, Global minimum energy conformation, Near-optimal solutions, Dead-end-elimination, Cost function network, Integer linear programming, Markov random field

1 Introduction

Computational Protein Design (CPD) seeks to identify amino-acid sequences that fold into stable known three-dimensional (3D) scaffolds and possess desired biophysical and functional properties. Achieving this goal requires facing several challenges. During the CPD process, amino-acid residues in the protein sequence are replaced by other possible amino acid types to find beneficial combined mutations for the targeted properties. Beyond the sequence identity, one has also to consider the conformational flexibility of the biomolecular system which follows from degrees of freedom

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_4,

around chemical bonds. The search space defined by both sequence identity and conformation grows exponentially with the number of considered mutations and becomes quickly out of reach of computational approaches. In this regard, the conformational search space is usually discretized using a set of side-chain conformations defined by their inner dihedral angles, which are called rotamers [1]. These low-energy side-chain conformations are derived from statistical analysis of high-resolution crystal structures in the Protein Data Bank [2]. Additionally, an assumption of modest protein backbone conformational flexibility is generally made. Numerous CPD methods consider a fixed protein backbone or a limited set of small changes. However, despite these simplifications, the size of the search space is still excessively large. Hence, efficient mehods are necessary to both evaluate sequence-conformation candidates based on their energy and to search through the sequence-conformation space a model of GMEC. In practice, an ensemble of near-optimal solutions is also desirable.

The most basic CPD problem defined by a fixed backbone with a corresponding set of positions and a rotamer library is formulated as an optimization problem that consists in choosing combinations of rotamers at designable specified positions such that the energy-based objective function is minimized. The energetic assessment of any combination of rotamers requires computationally efficient energy functions while being sufficiently accurate to discriminate between multiple sequence-conformation models. Energy functions used in CPD have been reformulated in such a way that the terms are pairwise decomposable [3]. From this formulation, the energy of a given protein sequence-conformation model, defined for each residue by a choice of one specific amino acid with an associated rotamer, can be written as:

$$E = E_{\emptyset} + \sum_{i} E(i_r) + \sum_{i} \sum_{j>i} E(i_r, j_s) \tag{1}$$

where E is the potential energy of the protein, $E_{\emptyset}$ is a constant energy contribution capturing interactions between fixed parts of the model, $E(i_r)$ is the energy contribution of rotamer r at position i capturing internal interactions or interactions with fixed regions, and $E(i_r, j_s)$ is the pairwise interaction energy between rotamer r at position i and rotamer s at position j [4]. This pairwise decomposition makes the CPD problem more amenable to computational optimization procedures. First, all the energy terms can be precomputed for each amino acid/rotamer (or $E(i_r, j_s)$ pair) independently of each other and stored in an energy matrix. Hence, once a specific rotamer has been chosen at each mutable amino-acid residue, the energy of a model can be quickly computed as the above-defined pairwise sum. Finally, to assess the fitness of the models, an appropriate objective function has to be appropriately defined with respect to the design purpose. Typically, to assess protein stability,

a reference energy term is incorporated into the $E(i_r)$ term without changing the form of the pairwise sum to take into account the unfolded protein state. The rigid backbone discrete rotamer CPD problem consists thus in identifying at each position i a pair from a subset D_i of all (amino-acid, rotamer) such that the overall energy E is minimized. In practice, based on knowledge of the molecular system and specific design goals, each position can be fixed (D_i is a singleton), flexible (all pairs in D_i have the same amino-acid type), or mutable (the general situation).

The main trend over the last decade is to extend this already difficult task to incorporate more and more flexibility to alleviate the inaccuracy resulting from the simplifications introduced in the modeling of the design problem. As an illustration, recent CPD approaches allow for consideration of continuous rotamers [5], flexible backbones or backbone ensembles [6], or both [7].

Despite its apparent simplicity, the rigid backbone discrete rotamer CPD problem as defined above has been proven to be NP-hard [8]. Even more, the problem has been shown hard to approximate [9]. For these reasons, stochastic local search methods based on Monte Carlo simulated annealing [10, 11], genetic algorithms [12], and many other algorithms [13–15] have been extensively developed to handle practical CPD optimization problems. These methods have a random component, may give a different answer for each run, and offer only asymptotic convergence. The general strategy of Monte Carlo simulated annealing methods (such as implemented into the well-known Rosetta modeling suite [16]) is to iteratively propose a random rotamer substitution (either the same amino acid or a new one) at a randomly picked residue and then decide whether or not the proposed modification should be accepted according to the Metropolis criterion [17]. A rotamer substitution is always accepted if it lowers the energy of the model while the acceptance or rejection of a modification that increases the energy is based on Bolzmann's relationship between probability and energy differences at a given temperature for the system. The substitution is accepted with Boltzmann probability or rejected otherwise. The system is slowly cooled throughout the run. The high initial temperature allows large jumps between local energy minima in the energy landscape and its reduction along the run gradually decreases the probability that move to a higher energy will be accepted. As a stochastic local search procedure, finding the GMEC is not guaranteed in finite time and the routine may end up trapped in local minima. To try to circumvent this, multiple independent runs are performed (each with a predefined number of steps) to cover, as well as possible, a rugged energy landscape. Genetic algorithms (such as implemented in EGAD [18]) are related in some aspects to Monte Carlo approaches. The main differences are that genetic algorithms work on a population of models throughout the run and mimic

genetic recombination and mutations to create new models from parents. The population dynamics of genetic algorithms make larger changes than Monte Carlo methods and thus, can more rapidly overcome energy barriers. However, each cycle is computationally more expensive than in the Monte Carlo method. The general procedure of genetic algorithm-based methods can be described as follows: a population of M models is generated beforehand. It defines the parent models for the next evolutionary process. Parent models are mutated with a given probability distribution associated with rotamers and N best mutants are chosen for recombination. A tournament selection technique (where N mutated models are picked at random) is applied to generate the new population of models. The model with the lowest energy is allowed to continue to the next generation. This selection step is repeated M times to produce the whole population of the next generation that will continue to the next round of mutation, recombination, and selection. The overall procedure is repeated until population equilibrium is reached. As in a Monte Carlo simulated annealing method, a "heating and cooling" process can be simulated by varying the number of models N, thus tailoring the pressure of selection. Initial low N values allow a broad population distribution and then, high N values restreint the variability of the population after each generation. This process is repeated to enhance the probability of finding lower minima.

These stochastic local search approaches have the advantage of providing a best known model at any time; however, they neither guarantee to find the GMEC nor a bounded energetic distance to the optimal solution. Moreover, the accuracy of stochastic methods also degrades as problem size increases [11]. In contrast, exact deterministic methods are able to get rid of these deficiencies. Since they can provably solve the problem to optimality, they ensure that when a discrepancy is found between computational and experimental results, the only possible culprit lies in the CPD model, and not in the optimization algorithm. This guaranty is fundamental in design cycles that go through modeling, solving, protein synthesis, and experimental evaluation. For a long time available deterministic methods have been extremely time-consuming, thus preventing their use to handle complex CPD problems. However, their advantages have motivated the recent development of more efficient deterministic approaches that are able to control the exponential explosion on increasingly large design sizes.

In this chapter, we present four exact search methods for the rigid backbone discrete rotamer problem, either based on the Dead-End-Elimination theorem combined with A^* algorithm (DEE/A^*), on Cost Function Networks algorithms (CFN), on Integer Linear Programming solvers (ILP) or on Markov Random Fields solvers (MRF). We then provide practical details to solve

GMEC-based protein design problems as well as to enumerate near-optimal solutions using two of these methods [3], the DEE/*A**, a well-established method in the CPD field and the more recent CFN method. Using the size of the sequence-conformation space as a proxy to the hardness of the problems for these methods, recent experiments [19] on 35 designs of increasing sizes showed that within 100 h, DEE/*A** was able to tackle 18 problems with sizes up to 10^{88} but choked on some problems with size 10^{47}. Instead, CFN algorithms were able to solve 30 problems, with sizes up to 10^{94} and started to choke only on problems of size 10^{61} (*see* Fig. 1).

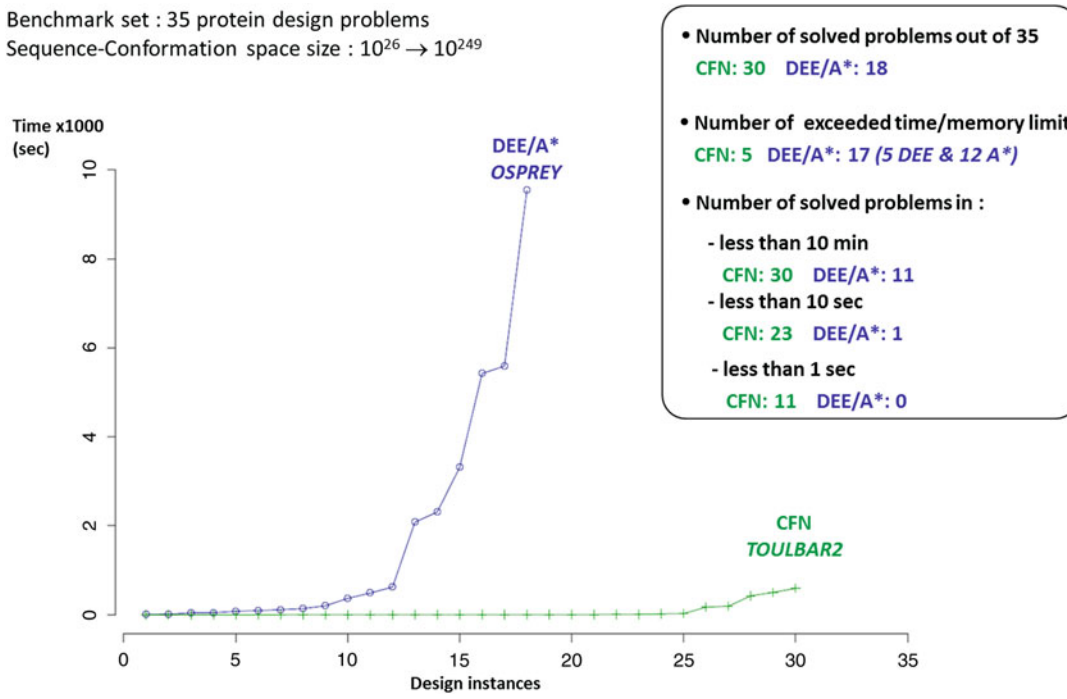


Fig. 1 CPU-time for solving the GMEC using DEE/A* (*osprey*) and CFN (*toulbar2*). The graph shows the number of Computational Protein Design instances solved to optimality by DEE/A* (in *blue*) and CFN (in *green*) (*X*-axis) as a function of time allowed for solving each problem (*Y*-axis). The performance of the algorithms was examined using a benchmark set of 30 CPD instances. This set comprises protein structures derived from the PDB which were chosen for the high resolution of their 3D structures and their distribution of sizes and types. Diverse sizes of sequence-conformation combinatorial spaces ranging from 10^{26} to 10^{249} were considered, varying by the number of mutable residues, the number of alternative amino acid types at each position, and the number of conformations for each amino acid (the *Penultimate* rotamer library was used). All computations (*toulbar2* and *osprey*) were performed on a single core of an AMD Operon 6176 at 2.3 GHz, 128 GB of RAM, and a 100 h time-out

2 Methods

2.1 DEE/A*-Based Search Method

DEE/A^* is the most widespread exact method in the CPD field. The two steps involved in this framework can be summarized as follows: (1) a preprocessing to reduce the search space, until a fixpoint is reached and (2) the application of a search algorithm to extract the optimum from the remaining space. The preprocessing step mainly relies on the so-called Dead-End Elimination (DEE) theorem [4, 20] and the A^* algorithm is the most applied search strategy by exact CPD solvers [21, 22].

DEE is a dominance analysis technique. The rotamer r at position i (denoted by i_r) is removed if there exists another rotamer u at the same position such that [4] :

$$E(i_r) + \sum_{j \neq i} \min_s E(i_r, j_s) \; > \; E(i_u) + \sum_{j \neq i} \max_s E(i_u, j_s) \qquad (2)$$

This criterion, referred to as the Desmet criterion, guarantees that the energy of any given conformation with rotamer r can be lowered if we substitute u for r, when such a rotamer exists. The Desmet criterion has later been improved by the Goldstein criterion that compares directly the energies of each rotamer within an identical conformational context [23].

$$E(i_r) - \; E(i_u) + \sum_{j \neq i} \min_s \left[\; E(i_r, j_s) - \; E(i_u, j_s) \; \right] \; > 0 \qquad (3)$$

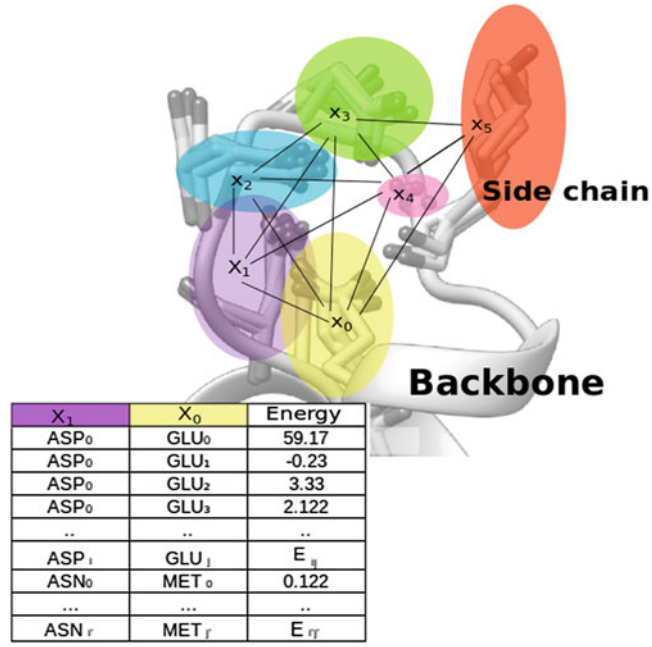
These two properties and various extensions of the DEE theorem define the polynomial time algorithms that prune dominated values [24–26].

However, although DEE has become a commonly used method in CPD, it is an incomplete algorithm: that is, it cannot solve all CPD instances. Therefore, DEE preprocessing is often followed by an A^* search that expands an energy sorted sequence-conformation tree. Thence, the first complete sequence-conformation reached by an A^* search is the GMEC and the following solutions are discovered in an increasing energy order [22]. But, unfortunately, CPD is NP-hard and the search problem may become intractable for A^* when the DEE preprocessing step does not reduce the search space sufficiently: the search becomes either too slow or memory demanding.

The DEE/A* method is available for example in *osprey*, a well-known program in the CPD field [27, 28] (*see* **Note 1**).

2.2 CFN-Based Search Method

The CPD optimization problem, in its pairwise-decomposed form, can be easily formulated as a Cost Function Network optimization problem (CFN), also known as a Weighted Constraint Satisfaction Problem (WCSP) (*see* Fig. 2).



X_1	X_0	Energy
ASP_0	GLU_0	59.17
ASP_0	GLU_1	-0.23
ASP_0	GLU_2	3.33
ASP_0	GLU_3	2.122
..	..	..
ASP_i	GLU_j	E_{ij}
ASN_0	MET_0	0.122
...	...	..
ASN_r	MET_t	E_{rt}

Fig. 2 Modeling of computational protein design problem (based on rigid backbone and discrete rotamers) as a Cost Function Network. Each variable amino acid residue is represented by a variable *X* (*highlighted using different colors*). The set of rotamers available to the residue defines the domain of the variable *X*. Each interaction energy term between pair of rotamers is represented as a cost function

A CFN P is defined by a set of variables that are each involved in a set of local cost functions [29]. Formally, a CFN P is a triple $P = (X, D, C)$ where $X = \{1, 2, .., n\}$ is a set of n variables. Each variable $i \in X$ has a discrete domain $D_i \in D$ that defines the set of values that it can take. A set of local cost functions C defines a network over X. Each cost function $c_S \in C$ is defined over a subset of variables $S \subseteq X$ (called its scope), has a domain $\prod_{i \in S} D_i$ and takes integer values in {0, 1, 2,.., k}. The cost k represents a maximum tolerable cost, and can be infinite or set to a finite upper bound. Values or pairs of values that are forbidden by a cost function are simply mapped to k. The global cost of a complete assignment A is defined as the sum of all cost functions on this assignment (or k if this sum is larger than k). The WCSP defined by P consists in finding an assignment of all variables that minimizes this global cost. Notice that it is usually assumed that C contains one constant cost function, with an empty scope, denoted as $c_\emptyset$. Since all cost functions in a CFN are nonnegative, this constant cost function $c_\emptyset \in C$ defines a lower bound on the optimization problem.

It is straightforward to map the CPD problem to the CFN model. Every nonrigid residue i is represented by a variable i and

the set of (amino acid, rotamer) pairs available to the residue defines its domain D_i. Then, each energy interaction term in E is represented as a cost function. The constant term E_c is captured as the constant cost function with empty scope ($c_\emptyset$) and terms $E(i_r)$ and $E(i_r, j_s)$ are represented by unary and binary cost functions involving the variables i and j of the corresponding residues. The mapping of energy terms to positive integers is done by shifting and scaling according to desired precision. The nonnegativity of cost functions is enforced by simply subtracting the minimum of every cost function from its cost table. Such operations preserve the set of optimal solutions. The joint cost distribution defined by the corresponding CFN is then equal to the energy, up to a known constant shift. The optimal solution of the CFN is an assignment that corresponds to a GMEC for the CPD problem (when stability is the objective function). When the maximum number of available rotamers over all residues is d, the resulting binary CFN takes space in $O(n^2d^2)$.

The fundamental processing technique in CFN optimization is the so-called Local Consistency filtering instead of dominance analysis by DEE. Enforcing Local Consistency can reformulate an initial CFN into an *equivalent* CFN, with the same variables and scopes but possibly smaller domains (value deletion) and an increased lower bound $c_\emptyset$ (lower bounding). By equivalent, we mean that the new CFN will assign the same cost to any complete assignment. This is obtained by the exclusive and repeated application of local transformations of the CFN that shift cost (or energy) between cost functions of intersecting scopes until a given *local consistency* property is satisfied. Many of these local consistency properties and associated polynomial time enforcing algorithms have been defined [30–32]. Depending on the locality of the property, which may apply to one variable, one cost function or more, they are called Node, Arc, or higher order consistencies. As an example, the node consistency of a variable i with associated cost function c_i requires that d_i contains at least one value v such that $c_i(v) = 0$ and no value w such that that $c_\emptyset + c_i(w) \geq k$ (the forbidden cost). Equivalently, this means that there is at least one value that does not increase cost locally and no value that would lead to intolerable costs. If a variable does not satisfy these properties, then by deleting values and shifting costs to $c_\emptyset$, the variable can be made node consistent. The amount of pruning therefore increases with smaller values of the upper bound k. Arc consistencies are defined similarly but are significantly more involved (*see* e.g. [30–32]). Since they preserve equivalence, local consistency algorithms are naturally incremental. This means they are not only useful as a preprocessing mechanism but can also be very cheap to maintain during search, usually within an exhaustive Depth First Branch and Bound (DFBB) algorithm, which ensures that the solution at the end of the search is the optimum. As search progresses, local consistency enforcing algorithms increasingly simplify

the initial problem and strengthen the lower bound that is used to prune during DFBB. Thence, the enforcing of local consistency properties may lead to pruning during search and provide heuristics to dynamically guide the search.

The very good performance of the CFN-based approaches as available in the *toulbar2* software [33, 34] (winner of the UAI Inference Challenge in 2010 and 2014) on CPD problems has been shown in recent publications [19, 35, 36] (*see* **Note 2**).

2.3 ILP-Based Search Method

The rigid backbone discrete rotamer CPD problem can also be represented as a zero/one linear program (01LP) problem [19, 35, 36] using the usual translation from CFN to ILP initially proposed in [37], which has later been proposed for CPD in [38]. A 01LP is defined by a linear criteria and a set of linear constraints on Boolean variables. For every value/rotamer i_r of the variable/residue i, one Boolean variable d_{i_r} is introduced. d_{i_r} indicates whether the rotamer i_r is used ($d_{i_r} = 1$) or not ($d_{i_r} = 0$). In order to enable the expression of the energy as a linear function of variables, an extra Boolean variable $p_{i_r j_s}$ is introduced for every pair of rotamers (i_r, j_s), capturing the fact that this pair of rotamers is used. The energy can then be expressed directly as the linear function to be minimized (the constant term can be ignored as it cannot change the optimal solution):

$$\sum_{i,r} E(i_r) \cdot d_{i_r} + \sum_{i,r,j,s} E(i_r, j_s) \cdot p_{i_r j_s} \tag{4}$$

Additional constraints enforce that exactly one rotamer is selected for each variable position and that a pair is used if, and only if, the corresponding values are used. Then, finding a GMEC reduces to the following 01LP:

$$\min \sum_{i,r} E(i_r) \cdot d_{i_r} + \sum_{i,r,j,s} E(i_r, j_s) \cdot p_{i_r j_s} \tag{5}$$

such that:

$$\sum_{r} d_{i_r} = 1 (\forall i) \tag{6}$$

$$\sum_{s} p_{i_r j_s} = d_{i_r} (\forall i, r, j) \tag{7}$$

The resulting ILP contains $O(n^2 d^2)$ variables and $O(n^2 d)$ constraints. Note that since the objective function is nonlinear, it is fundamentally impossible to express it in 01LP without introducing a quadratic number of variables. Hence, this 01LP model cannot be improved significantly in size.

This type of model can be handled by various ILP solvers such as IBM ILOG *cplex* (*see* **Note 3**).

2.4 MRF-Based Search Method

The CPD problem can also be formulated as a probabilistic graphical model [19, 39], such as a Markov random field [40]. In this formalism, a concise description of a joint distribution of probabilities over a set of variables is obtained through a factorization in local terms, involving only few variables. For terms involving at most two variables, if vertices represent variables and edges represent terms, a factorization can be represented as a graph, hence the name of graphical models. The same idea is used for concisely describing a cost distribution in Cost Function Networks.

A discrete Markov Random Field (MRF) can be defined as a pair (X, Φ) where $X = \{1, \ldots, n\}$ is a set of n random variables and Φ is a set of potential functions. Each variable $i \in X$ has a finite domain D_i of values that can be assigned to it. A potential function $\phi_S \in \Phi$ with scope $S \subseteq X$ is a function $\phi_S : D_S \to \mathbb{R}$. A MRF implicitly defines a nonnormalized probability distribution over X. The probability of a given tuple t is defined as:

$$P(t) = \frac{\exp\left(-\sum_{\phi_S \in \Phi} \phi_S(t[S])\right)}{Z} \tag{8}$$

where Z is a normalizing constant (the partition function).

From the sole point of view of optimization, the problem of finding an assignment of maximum probability, also known as the maximum a posteriori (MAP) assignment in a MRF or a minimum cost solution of a CFN, is equivalent by monotonicity of the exp() function. Only technical differences remain: CFNs are restricted to nonnegative and usually integer costs. Being focused on optimization, CFNs also emphasize the existence of a possibly finite upper bound k that can be exploited for pruning.

The CPD problem can therefore directly be modeled as the MAP problem in a MRF exactly as earlier described for CFN, using additive potentials to capture energies (*see* for example [41]).

These models can be solved using MAP-MRF solvers such as *daoopt* [33, 34] (winner of the Pascal Inference Challenge in 2011) (*see* **Note 4**) or the recent version of the *mplp* [34] solver (*see* **Note 5**).

3 Practical Procedure

In this section, we describe procedures to solve the GMEC identification problem with the DEE/A^* CPD-dedicated package, *osprey* version 2.0 [27, 28] (*see* **Note 1**) and the CFN solver, *toulbar2* version 0.9.6 (*see* **Note 2**) from the energy matrix precomputed for a protein design problem. In addition to the identification of the GMEC, both methods can also enumerate an ensemble of suboptimal solutions within a given energy interval, which can be of

interest for the experimental construction of rational protein mutant libraries. The procedures to generate these suboptimal sequence ensembles are also explained hereafter. Notice that the *toulbar2* CFN solver has been shown to outperform the DEE/*A** approach by several orders of magnitude for the GMEC identification and also for producing a set of suboptimal solutions [19, 35, 36] (*see* Fig. 1). In practice, this latter step has been found unattainable using DEE/*A** in numerous CPD cases [36]. All computational scripts mentioned, as well as the CPD instance handled in the following example, have been made available to the scientific community (in the archive *SpeedUp2* at the following address: http://genoweb.toulouse.inra.fr/~tschiex/CPD/SpeedUp2.tgz). They assume the use of a Linux/Unix environment using a *sh (bash)* shell.

Before using any of these exact deterministic optimization methods, the pairwise decomposed energy matrix needs to be computed and stored. This can be achieved using the patched and compiled version of *osprey 2.0* [27, 28], available in the *Osprey2.0* directory of the *SpeedUp2* archive, which works under most 64 bits Linux systems with Java (6 or above) installed. The result is a binary matrix file that will be later used to generate the input for *toulbar2* solver. The command line for computing a pairwise energy matrix is:

```
java -cp Osprey2.0/src:Osprey2.0/src/mpiJava/lib/classes -Xmx2G
KStar -t 5 -c inp/KStar.cfg computeEmats inp/System.cfg inp/DEE.
cfg >out/matrix.out 2>&1 < /dev/null
mv dat/matrixEMmin_COM dat/matrixEMmin_COM.dat
```

The *KStar.cfg* file contains parameters to define the force field, the weights of energy terms, and the path to the rotamers library. The *System.cfg* file defines the input pdb model (parameter *pdbName*) as well as the variable residues (parameter *strandMut0*: list of pdb residue number with *strand0* that indicates the range of considered residues from the chain *index 0* and the suffix *0* that is an index on the molecular chain). The list of amino acids allowed at each *i*th variable residue is defined by the *resAllowedx_y* parameters from the DEE.cfg file (x is the chain number, and y is the *i*th variable residue defined at strandMut0 for $x = 0$). More details can be found in *osprey* user manual (available in the *SpeedUp2* archive as well as at the following URL http://www.cs.duke.edu/donaldlab/osprey.php).

3.1 DEE/A*-Based Optimization Using *osprey*

1. The previously generated binary file can be handled internally by *osprey*. The GMEC identification can be accomplished by the following command line that produces a sequence-conformation file in the *conf_info* directory:

   ```
   java -cp Osprey2.0/src:Osprey2.0/src/mpiJava/lib/classes -
   Xmx2G KStar -c inp/KStar.cfg doDEE inp/System.cfg inp/DEE.
   cfg >doDEE.out 2>&1 < /dev/null
   ```

2. For the generation of near-optimal solutions, it is necessary to modify the *initEw* parameter from file *inp/DEE.cfg*. This parameter defines the interval within which near-optimal solutions are enumerated. Simply setting its value to 0.5 for example will cause the previous command line to enumerate all solutions within 0.5 kcal/mol of the GMEC.

3.2 CFN-Based Optimization Using toulbar2

Alternatively, the open source *CFN* solver *toulbar2* can be used to identify the GMEC or generate suboptimal solutions of the CPD problem. By default, *toulbar2* maintains Existential Directional Arc Consistency [42] for incremental lower bounding, dynamic value ordering (based on minimum unary cost), and a variable ordering heuristics (based on the median energy of terms involving a given residue following preprocessing) combined with last conflict heuristics [43]. To use the *toulbar2* solver, it is necessary to generate a specific text file format defining a WCSP problem beforehand (.wcsp file).

1. The translation of the energy matrix into a CFN model can be accomplished by the command line below. An additional text matrix file is generated, which is used thereafter to translate solutions into the osprey sequence-conformation file format:

```
java -cp Osprey2.0/src/:Osprey2.0/src/mpiJava/lib/classes/
KStar -c inp/KStar.cfg writeWcsp inp/System.cfg inp/DEE.cfg
>writeWCSP.out
```

2. The CFN-based optimization using *toulbar2* can be performed by *scripts/run_toulbar2.sh*. The first step in this script is to perform the computation of the GMEC, followed by the extraction of the solution from the output and its translation into *osprey* conformation file by the script *scripts/sol2conf.pl*).

```
name=matrixEMmin
./bin/toulbar2 dat/$name.wcsp -l=3 -m -d: -s > out/$name.
wcsp.opt.out
grep -A 1 "New solution" out/$name.wcsp.opt.out|tail -1 |sed -
re "s/^/ solution:/" > out/$name.wcsp.opt.sol
perl scripts/sol2conf.pl -mat=dat/$name.quick -tbsol=out/
matrixEMmin.wcsp.opt.sol
```

The file *out/$name.wcsp.opt.sol* contains the solution found by *toulbar2*. The corresponding *osprey* conformation file is generated at *conf_file/$name.wcsp.opt.sol.conf.sorted*

The second step in the scripts is the computation of the suboptimal ensemble. The cost of the GMEC is used to define the upper bound below which suboptimal solutions are enumerated. The threshold from the GMEC energy is controlled by *ew* (0.5 kcal/mol in this example).

```
ew=`bc -l <<<" 0.5 * 10^8"|awk '{printf "%d",$0}'` #kcal.mol-1
lb=`egrep "^Optimum:" out/${name}.wcsp.opt.out|awk '{print
$2}'` # lowerbound
ub=`bc -l <<<" $lb + $ew"` # upperbound
./bin/toulbar2 dat/$name.wcsp -d: -m -a -s -ub=$ub >out/
$name.wcsp.enum 2>&1
perl scripts/sol2conf.pl -mat=dat/$name.quick -tbsol=out/
$name.wcsp.enum -useq
```

The *sol2conf.pl* script can be restricted to just produce the best conformation for each sequence by using the *useq* flag. The associated *fasta* file reporting sequences, energies, and the number of occurrences for each sequence is also written.

The translation of the generated conformations to pdb structures files using *osprey* is performed by the following script. Its argument is the conformation file. A single pdb file is generated into the pdbs subdirectory for each line of the conformation file.

```
bash scripts/genstruct.sh conf_info/$name.wcsp.conf.sorted
```

4 Conclusion

The development of computational methods to guide the design of novel proteins has come a long way in the last decade. Considerable efforts have been accomplished to better account for many essential aspects of the protein design problem going from a more realistic physical modeling of the problem, the quantum modeling of the reaction transition state, the treatment of limited molecular flexibility, the development of more accurate energy functions, and a more efficient optimization of the combinatorial sequence-conformation space.

Regarding this latter area, exact deterministic methods have shown to be very efficient to search the CPD sequence-conformation space to provably identify the lowest-energy solution. In particular, we presented here three alternative exact deterministic solvers, based on Cost Function Network algorithms (CFN), Integer Linear Programming solvers (ILP), and Markov Random Fields solvers (MRF), which have yet been little applied to CPD but have demonstrated their ability to handle highly complex CPD problems, thus offering novel computational solutions. In particular, the CFN-based methods have led to tremendous improvements compared to the CPD commonly used DEE/*A** algorithm (*see* Fig. 1). CFN methods not only enable quickly identifying the GMEC solution but they are capable of enumerating all suboptimal solutions within a threshold of the optimum, which is often out of reach of DEE/*A** algorithm. This information is of particular use for the rational construction of focused protein sequence libraries. New CFN algorithmic developments

targeted at CPD may even be able to push the computational barrier to more complex design problems, either in terms of size or definition (e.g., multistate). While restricted to fixed backbone/rotamer-based designs, CFN-based methods also have the capacity to replace DEE/*A** in all existing deterministic CPD algorithms that rely on the optimization on precomputed energy matrices, including those targeted at continuous rotamers [5], flexible backbones or backbone ensembles [6], or both [7].

With continued development of methods that address the points mentioned above, we are optimistic that further improvements will help to increase reliability and accuracy of CPD methods, which can have an impact on the development of proteins and catalysts for biotechnologies and nanotechnologies.

5 Notes

1. The *osprey* open source CPD-dedicated software is available at http://www.cs.duke.edu/donaldlab/osprey.php/.
2. *toulbar2* is an international collaborative CFN solver development. It was the winning solver of the UAI Probabilistic Inference Challenge in 2010 and 2014 and it finished second in the 2011 PASCAL Probabilistic Inference Challenge (PIC) in the "MAP" category. All sources are available on the git repository at http://mulcyber.toulouse.inra.fr/projects/toulbar2. Specific CPD extensions are available in the "cpd" branch.
3. IBM ILOG *cplex* is free for academics as described on the dedicated IBM academic initiative web site at http://www-01.ibm.com/software/websphere/products/optimization/academic-initiative/.
4. *daoopt* is the winning solver of the 2011 PASCAL Probabilistic Inference Challenge (PIC) in the "MAP" category. It can be downloaded at: https://github.com/lotten/daoopt. The distributed version of *daoopt* is not the same as the PIC challenge version. It lacks the Dual Decomposition bound strengthening component [33] that relies on private code. This solver relies on Stochastic Local Search for finding initial solutions followed by depth-first AND/OR search [44] and mini-bucket lower bounds [45] for pruning. Mini-bucket lower bounds require space and time in $O(d^i)$ (where is a user-controlled parameter).
5. The sources for the recent version 2 of the *mplp* (Message Passing Linear Programming) implementation can be downloaded at http://cs.nyu.edu/~dsontag/. This solver uses a Message Passing based bound and duality theory to identify optimal solutions of a MAP-MRF problem through successive

tightening of subsets of variables. The message passing used in *mplp* defines reparametrizations of the underlying MRF. These reparametrizations are similar to the reformulations done by local consistencies in CFN [30, 46]. The solver is unique in all the solvers considered in that it never uses branching but only increasingly strong inference by applying reparametrizations to set of variables that initially contain only pairwise potentials, reasoning on stars [47], and are incrementally enlarged to include several potentials and strengthen the corresponding bound [34, 48].

Acknowledgments

This work has been funded by a grant from INRA and the Region Midi-Pyrénées and the "Agence Nationale de la Recherche," references ANR 10-BLA-0214 and ANR-12-MONU-0015-03. We thank the Computing Center of Region Midi-Pyrénées (CALMIP, Toulouse, France) and the GenoToul Bioinformatics Platform of INRA-Toulouse for providing computing resources and support.

References

1. Shapovalov MV, Dunbrack RL Jr (2011) A smoothed backbone-dependent rotamer library for proteins derived from adaptive kernel density estimates and regressions. Structure 19 (6):844–858. doi:10.1016/j.str.2011.03.019
2. Bernstein FC, Koetzle TF, Williams GJ, Meyer EF Jr, Brice MD, Rodgers JR, Kennard O, Shimanouchi T, Tasumi M (1977) The Protein Data Bank. A computer-based archival file for macromolecular structures. Eur J Biochem 80 (2):319–324
3. Boas FE, Harbury PB (2007) Potential energy functions for protein design. Curr Opin Struct Biol 17(2):199–204. doi:10.1016/j.sbi.2007.03.006
4. Desmet J, De Maeyer M, Hazes B, Lasters I (1992) The dead-end elimination theorem and its use in protein side-chain positioning. Nature 356(6369):539–542
5. Gainza P, Roberts KE, Donald BR (2012) Protein design using continuous rotamers. PLoS Comput Biol 8(1), e1002335
6. Georgiev I, Donald BR (2007) Dead-end elimination with backbone flexibility. Bioinformatics 23(13):185–194
7. Ma H, Keedy DA, Donald BR (2013) Dead-end elimination with perturbations (DEEPer): a provable protein design algorithm with continuous sidechain and backbone flexibility. Proteins 81(1):18–39. doi:10.1002/prot.24150
8. Pierce NA, Winfree E (2002) Protein design is NP-hard. Protein Eng 15(10):779–782. doi:10.1093/protein/15.10.779
9. Chazelle B, Kingsford C, Singh M (2004) A semidefinite programming approach to side chain positioning with new rounding strategies. Informs J Comput 16(4):380–392
10. Kuhlman B, Baker D (2000) Native protein sequences are close to optimal for their structures. Proc Natl Acad Sci U S A 97 (19):10383–10388
11. Voigt CA, Gordon DB, Mayo SL (2000) Trading accuracy for speed: a quantitative comparison of search algorithms in protein sequence design. J Mol Biol 299(3):789–803. doi:10.1006/jmbi.2000.3758
12. Raha K, Wollacott AM, Italia MJ, Desjarlais JR (2000) Prediction of amino acid sequence from structure. Protein Sci 9(6):1106–1119. doi:10.1110/ps.9.6.1106
13. Ogata K, Jaramillo A, Cohen W, Briand J, Conan F, Wodak S (2003) Automatic sequence design of MHC class-I binding peptides impairing CD8+ T cell recognition. J Biol Chem 278:1281

14. Allen BD, Mayo SL (2006) Dramatic performance enhancements for the FASTER optimization algorithm. J Comput Chem 27 (10):1071–1075
15. Desmet J, Spriet J, Lasters I (2002) Fast and accurate side-chain topology and energy refinement (FASTER) as a new method for protein structure optimization. Proteins 48(1):31–43. doi:10.1002/prot.10131
16. Leaver-Fay A, Tyka M, Lewis SM, Lange OF, Thompson J, Jacak R, Kaufman K, Renfrew PD, Smith CA, Sheffler W, Davis IW, Cooper S, Treuille A, Mandell DJ, Richter F, Ban YE, Fleishman SJ, Corn JE, Kim DE, Lyskov S, Berrondo M, Mentzer S, Popovic Z, Havranek JJ, Karanicolas J, Das R, Meiler J, Kortemme T, Gray JJ, Kuhlman B, Baker D, Bradley P (2011) ROSETTA3: an object-oriented software suite for the simulation and design of macromolecules. Methods Enzymol 487:545–574. doi:10.1016/B978-0-12-381270-4.00019-6
17. Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, Teller E (1953) Equation of state calculations by fast computing machines. J Chem Phys 21:1087
18. Chowdry AB, Reynolds KA, Hanes MS, Voorhies M, Pokala N, Handel TM (2007) An object-oriented library for computational protein design. J Comput Chem 28 (14):2378–2388. doi:10.1002/jcc.20727
19. Allouche D, André I, Barbe S, Davies J, de Givry S, Katsirelos G, O'Sullivan B, Prestwich S, Schiex T, Traoré S (2014) Computational protein design as an optimization problem. Artif Intell 212:59–79. doi:10.1016/j.artint.2014.03.005
20. Dahiyat BI, Mayo SL (1996) Protein design automation. Protein Sci 5(5):895–903
21. Leach AR, Lemon AP (1998) Exploring the conformational space of protein side chains using dead-end elimination and the A* algorithm. Proteins 33(2):227–239
22. Georgiev I, Lilien RH, Donald BR (2008) The minimized dead-end elimination criterion and its application to protein redesign in a hybrid scoring and search algorithm for computing partition functions over molecular ensembles. J Comput Chem 29(10):1527–1542
23. Goldstein RF (1994) Efficient rotamer elimination applied to protein side-chains and related spin glasses. Biophys J 66 (5):1335–1340
24. Pierce NA, Spriet JA, Desmet J, Mayo SL (2000) Conformational splitting: a more powerful criterion for dead-end elimination. J Comput Chem 21(11):999
25. Looger LL, Hellinga HW (2001) Generalized dead-end elimination algorithms make large-scale protein side-chain structure prediction tractable: implications for protein design and structural genomics. J Mol Biol 307 (1):429–445. doi:10.1006/jmbi.2000.4424
26. Georgiev I, Lilien RH, Donald BR (2006) Improved pruning algorithms and divide-and-conquer strategies for dead-end elimination, with application to protein design. Bioinformatics 22(14):E174–E183. doi:10.1093/bioinformatics/btl220
27. Chen C-Y, Georgiev I, Anderson AC, Donald BR (2009) Computational structure-based redesign of enzyme activity. Proc Natl Acad Sci 106(10):3764–3769
28. Gainza P, Roberts KE, Georgiev I, Lilien RH, Keedy DA, Chen CY, Reza F, Anderson AC, Richardson DC, Richardson JS, Donald BR (2013) Osprey: protein design with ensembles, flexibility, and provable algorithms. Methods Enzymol 523:87–107. doi:10.1016/B978-0-12-394292-0.00005-9
29. Schiex T, Fargier H, Verfaillie G (1995) Valued constraint satisfaction problems: hard and easy problems. Int Joint Conf Artif Intell 14:631–639
30. Cooper M, Schiex T (2004) Arc consistency for soft constraints. Artif Intell 154(1):199–227
31. Larrosa J, Schiex T (2004) Solving weighted CSP by maintaining arc consistency. Artif Intell 159(1):1–26
32. Cooper M, Givry Sd, Schiex T (2006) The quest for the best arc consistent closure in weighted CSP. In: 8th International CP-06 workshop on preferences and soft constraints, Nantes, France
33. Otten L, Dechter R (2012) Anytime {AND/OR} depth-first search for combinatorial optimization. Artif Intell Commun 25(3):211–227
34. Sontag D, Choe DK, Li Y (2012) Efficiently searching for frustrated cycles in {MAP} inference. AUAI Press, Corvallis, OR, pp 795–804
35. Allouche D, Traoré S, André I, de Givry S, Katsirelos G, Barbe S, Schiex T (2012) Computational protein design as a cost function network optimization problem CP 2012
36. Traoré S, Allouche D, André I, de Givry S, Katsirelos G, Schiex T, Barbe S (2013) A new framework for computational protein design through cost function network optimization. Bioinformatics. doi:10.1093/bioinformatics/btt374
37. Koster AMCA, van Hoesel SPM, Kolen AWJ (1999) Solving frequency assignment problems via tree-decomposition. Electron Notes Discrete Math 3:102

38. Kingsford CL, Chazelle B, Singh M (2005) Solving and analyzing side-chain positioning problems using linear and integer programming. Bioinformatics (Oxford) 21 (7):1028–1036. doi:10.1093/bioinformatics/bti144
39. Zhou Y, Wu Y, Zeng J (2015) Computational protein design using AND/OR branch-and-bound search. In: Przytycka TM (ed) Research in computational molecular biology, vol 9029, Lecture notes in computer science. Springer, New York, NY, pp 354–366. doi:10.1007/978-3-319-16706-0_36
40. Khoury GA, Smadbeck J, Kieslich CA, Floudas CA (2014) Protein folding and \emph{de novo} protein design for biotechnological applications. Trends Biotechnol 32(2):99–109
41. Yanover C, Meltzer T, Weiss Y (2006) Linear programming relaxations and belief propagation—an empirical study. J Mach Learn Res 7:1887–1907
42. De Givry S, Heras F, Zytnicki M, Larrosa J (2005) Existential arc consistency: getting closer to full arc consistency in weighted CSPs. In: IJCAI'05 proceedings of the 19th international joint conference on Artificial intelligence
43. Lecoutre C, Saïs L, Tabary S, Vidal V (2009) Reasoning from last conflict(s) in constraint programming. Artif Intell 173:1592–1614
44. Dechter R, Mateescu R (2007) {AND/OR} search spaces for graphical models. Artif intell 171(2):73–106
45. Dechter R, Rish I (2003) Mini-buckets: a general scheme for bounded inference. J ACM 50 (2):107–153
46. Schiex T (2000) Valued constraint networks. In: Proceedings of the 6th conference on principles and practice of constraint programming
47. Globerson A, Jaakkola TS (2007) Fixing max-product: convergent message passing algorithms for MAP LP-relaxations. In: NIPS'07 Proceedings of the 20th international conference on neural information processing systems, pp 553–560
48. Sontag D, Meltzer T, Globerson A, Weiss Y, Jaakkola T (2008) Tightening {LP} relaxations for {MAP} using message-passing. AUAI Press, Corvallis, OR, pp 503–510

Chapter 5

Geometric Potentials for Computational Protein Sequence Design

Jie Li and Patrice Koehl

Abstract

Computational protein sequence design is the rational design based on computer simulation of new protein molecules to fold to target three-dimensional structures, with the ultimate goal of designing novel functions. It requires a good understanding of the thermodynamic equilibrium properties of the protein of interest. Here, we consider the contribution of the solvent to the stability of the protein. We describe implicit solvent models, focusing on approximations of their nonpolar components using geometric potentials. We consider the surface area (SA) model in which the nonpolar solvation free energy is expressed as a sum of the contributions of all atoms, assumed to be proportional to their accessible surface areas (ASAs). We briefly review existing numerical and analytical approaches that compute the ASA. We describe in more detail the alpha shape theory as it provides a unifying mathematical framework that enables the analytical calculations of the surface area of a macromolecule represented as a union of balls.

Key words Protein structure, Solvation free energy, Accessible surface area, Delaunay triangulation

1 Introduction

Proteins, the end products of the processing of the information contained in the genome of any organism, are the biological molecules whose chemical activities regulate most cellular processes. It is fascinating to see how nature has arranged simple atoms in such a way as to facilitate a myriad of activities. This fascination has led many scientists to design and create their own customized proteins to perform prescribed functions, defining a research field of their own, protein design, also called protein engineering. In this protocol, we cover the computational efforts associated with this field, focusing on the implementation of geometric potentials as a support to the task of identifying protein sequences that are compatible with a given scaffold. We note that those potentials have broader impacts in the general field of molecular simulations.

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_5,

Protein design requires a good understanding of the relationship between a protein sequence and its structure. Recent progress in genomics and structural genomics has led to an explosion in the amount of experimental data available on proteins. There are currently (as of March 2015) more than 90 million protein sequences available in the UniProt database [1] and more than 100,000 protein structures in the Protein Data Bank [2]. The large gap however between those two numbers, and the difficulties encountered while trying to decipher the relationship between a protein sequence and its structure from those data, has led to the development of many modeling initiatives to shed lights on these connections [3]. Probably, the most famous is the study of the protein-folding problem—the "holy grail" for the structural biology community that focuses on proteins. Its elusive goal is to predict the detailed three-dimensional structure of a protein from its sequence. This "holy grail" is still considered out of reach [4–6], although significant progress has been made recently for the prediction of small, globular proteins [7–9]. Interestingly, the difficulties encountered in trying to solve the protein-folding problem have led to the development of an alternative route in which the quest is reformulated as searching for protein sequences that fold into a given stable conformation. This is the inverse folding problem [10, 11], whose successes have paved the way for efficient and successful computer-based protein sequence design (for a nonexhaustive list of recent successes in designing small proteins as well as large nano-assemblies, *see* refs. 12–18).

All modeling investigations that consider the structure of a protein require an understanding of the thermodynamic equilibrium properties of the protein, which are usually derived from a sampling of its free energy surface. The "state" of a protein structure usually corresponds to a point or patch on this surface, with the native state usually associated with a large patch, also referred to as basin. Protein-folding studies are mostly interested in the structure of this basin, while computational protein design studies focus on how this basin changes as the sequence of the protein is changed.

The stability of the native state of a protein is measured as the difference $\Delta G(P)$ in free energy between its native state, N, and a reference, usually unfolded state, U:

$$\Delta G_{\mathrm{U}\to\mathrm{N}}(P) = G_{\mathrm{N}}(P) - G_{\mathrm{U}}(P) \tag{1}$$

Note that "P" here refers to the "solvated" protein, i.e., accounts for the protein and its surrounding solvent and ionic environment; G refers to the Gibbs free energy of the system. A typical computational protein sequence design experiment starts from a known protein structure template N and tests the "compatibility" of many sequences for this template, searching for sequences that are

both stable (positive design) and specific (negative design) to the structure N. Two putative sequences P_0 and P_1 for N are compared based on their stability, as defined by Eq. 1:

$$\Delta\Delta G_{\mathrm{U}\rightarrow\mathrm{N}}(P_0 \rightarrow P_1) = \Delta G_{\mathrm{U}\rightarrow\mathrm{N}}(P_1) - \Delta G_{\mathrm{U}\rightarrow\mathrm{N}}(P_0) \qquad (2)$$

To compute the free energy G of a protein, which is required in Eqs. 1 and 2, we need to compute its internal energy U and entropy S. In theory, the laws of quantum mechanics fully define the energetics U of a molecule. In practice, however, only the simplest system such as the hydrogen atom can be solved exactly, and modelers of large molecular systems such as proteins must rely on approximations. While some simulations remain anchored in quantum mechanics [19, 20], most computational protein design studies rely on a space-filling representation of the molecule, in which atoms are represented as hard spheres that interact through empirical or semiempirical "molecular force fields" [21]. In addition, computational protein design usually relies on implicit solvent models that reduce the protein-solvent interactions to their mean-field characteristics, which are expressed as a function of the protein degrees of freedom alone. These models represent the solvent as a dielectric continuum that mimics the solvent-solute interactions, including their nonpolar components (vdW contacts and the entropic effects of creating a cavity in the solvent) and their polar components (mostly through screening of electrostatics interactions). This protocol focuses on approximations of the nonpolar component using geometric potentials.

Eisenberg and McLachlan [22] computed the nonpolar part of the free energy of solvation as the sum of the contributions from all atoms of a protein P. The contribution of one atom is computed as the product of its accessible surface area, ASA [23], with a surface tensor factor referred to as Atomic Solvation Parameter, or ASP:

$$W_{\mathrm{np}}(P) = \sum_i \mathrm{ASP}_i \times \mathrm{ASA}_i \qquad (3)$$

ASP is positive for nonpolar atoms and negative for polar atoms. This model, referred to as SA (for Surface Area), is supported indirectly by the observed linearity between the Gibbs free energy and the surface area for transferring small compounds from nonaqueous liquids to water. Similarly, the free energy of solvation correlates with the sum of the transfer free energies of the constituent atomic groups. SA has become the method of choice for computing the hydrophobic effects on proteins. It is interesting to recall that W_{np} accounts for cavity formation in water as well as the vdW interactions between the protein and the solvent molecules. The latter occurs within the first hydration shell around the protein, and therefore is expected to be proportional to the accessible surface area of the protein. Cavity formation, on the other hand, is

proportional to the volume of the protein. This apparent contradiction between a surface area model and a volume model is part of the debate on the geometric nature of the nonpolar solvation energy. Lum, Chandler and Weeks have unified these two models by showing that W_{np} scales with the volume of the solute for small solutes, and is proportional to the surface area for large solutes [24]. Their theory of hydrophobicity adds to the validation of the surface area model for proteins.

The original approach of Lee and Richards computed the accessible surface area of a protein by first cutting the molecule with a set of parallel planes [25]. The intersection of a plane with an atom is a circle that can be partitioned into accessible arcs on the boundary and occluded arcs in the interior. The accessible surface area of an atom is then the sum of the contributions of all its accessible arcs. Shrake and Rupley proposed an alternative approach based on numerical integration of the surface area using a Monte Carlo method [26]. Implementations of their method include applications of lookup tables [27], vectorized algorithms [28], and parallel algorithms [29]. The surface area computed by numerical integration however lacks accuracy. To improve the accuracy of numerical methods, analytical approximations to the accessible surface area were developed by treating multiple overlaps probabilistically [30, 31] or ignoring them altogether [32]. Better analytical methods describe the molecule as a geometric union of spheres, and analytically compute the surface area [33–36]. Yet another approach uses the inclusion–exclusion formula [37] and applies a theorem, which states that overlaps of order five and above can always be reduced to overlaps of order four or below [38]. Doing the reduction correctly and efficiently is a difficult task. An exact solution was later obtained by using the Alpha Shape Theory of Edelsbrunner [39], which is the basis of the method described below [40, 41].

2 Materials

To compute the surface-area-based solvation free energy of a protein (Eq. 3) requires knowledge of the coordinates of all atoms of the protein, a program to compute accessible surface area, and the values of the Atomic Solvation Parameters.

2.1 Atomic Coordinates

The solvation free energy given by Eq. 3 can only be computed if the 3D structure of the protein is known. If this structure has been elucidated experimentally, it is made available freely in the Protein Data Bank, PDB, accessible at www.rcsb.org [2]. In the database, it is identified with a 4-character tag that can be recovered using their search engine. The PDB file contains the information needed, namely the X, Y, and Z coordinates of all atoms that were identified

Table 1
Atomic groups in proteins and their vdW radii and atomic solvation parameters

Atomic group [44]	Radii (Å) [45]	Atomic solvation parameters (kcal/Å^2) [49]
C3H0	1.76	36.0
C3H1	1.76	36.0
C4H1	1.87	36.0
C4H2	1.87	36.0
C4H3	1.87	36.0
N3H0	1.50	8.1
N3H1	1.65	8.1
N3H2	1.65	8.1
N4H3	1.50	−46.0
O1H0	1.40	−5.0
O2H1	1.40	8.1
S2H0	1.85	44.0
S2H1	1.85	44.0

experimentally (*see* **Note 1**). If the structure has been generated using a software resource for molecular simulation package, the 3D coordinates of its atoms will be automatically available.

2.2 Atomic Radii

Each atom in the protein is assigned a radius, usually taken to correspond to its vdW radius. The vdW radii of individual atoms have been well documented [42, 43]. Within proteins, however, the positions of hydrogen atoms are not generally known. This means that hydrogen atoms are usually subsumed into the "heavy" atoms to which they are covalently linked, creating atomic groups. The radius for an atomic group, such as the methyl group (–CH3), applies to the group as a whole. Several sets of radii for atomic groups are available in the literature, but there are appreciable differences among them (for review, *see* ref. 44). We list in Table 1 the different chemical groups and the radii we recommend, as defined by Chothia [45].

2.3 Software Resources for Computing Accessible Surface Area

The different programs currently available differ in the methodologies they use and can be divided into two groups, those that rely on numerical integration, and those that apply an analytical method (see the discussion above). Table 2 lists the most common of those

Table 2
Standard packages for computing accessible surface areas of proteins

Package	Availability	Comments
ASV	petitjeanmichel.free.fr/itoweb.petitjean.spheres.html	Exact analytical method [58] *Free for academic use*
Msroll	biohedron.drupalgardens.com	Exact analytical method [33] *Free for academic use*
Naccess	www.bioinf.manchester.ac.uk/naccess	Numerical method *Free for academic use*
PDBREMIX	boscoh.github.io/pdbremix	Includes pdbasa (numerical method) *Opensource*
POPS	http://mathbio.nimr.mrc.ac.uk/wiki/Software	Analytical method based on approximate probabilistic formula [59] *Opensource (GPL)*
UnionBall	Contact author: koehl@cs.ucdavis.edu	Exact analytical method [40] *Opensource (LGPL)*

This list is far from exhaustive. Note that many modeling software resources include their own implementation of a numerical or analytical method for computing the accessible surface area

software resources, providing information on how to access them. Our own analytical implementation based on the Alpha Shape theory is listed (UnionBall; [40]).

2.4 Atomic Solvation Parameters

Atomic solvation parameters (ASPs) are scaling factors that relate surface areas to solvation energies. Eisenberg and McLachlan [22] developed the surface-area-based solvation free energy model that proposed to compute the ASPs from the experimental free energies of transfer of analogs of amino acids from an hydrophobic environment (n-octanol) to an hydrophilic environment (water) [46]. They showed that only five classes of atoms are needed to obtain a good fit between free energies computed from Eq. 1 and the corresponding experimental free energies of transfer. The corresponding ASP values, however, were deemed to be incorrect, as the experimental transfer free energy values need to be corrected to account for size and contact effects [47, 48]. We advocate the use of the corresponding corrected values, as derived for example in ref. 49. Those values are given in Table 1.

3 Methods

UnionBall is our software package that implements the Alpha Shape theory for computing the accessible surface area and volume of a union of balls [40]; its origins lie in the Alpha Shape package

[50, 51]. UnionBall takes as input a set of balls B_i in space, each specified by the coordinates of its center z_i and its radius r_i. In the case of a protein, the coordinates of the centers c_i are extracted from the corresponding PDB file (*see* Subheading 2), while the radii r_i are computed as the sum of the vdW radii corresponding to the atom types (*see* Table 1) and the radius R_w of a probe, usually set to 1.4 Å to correspond to the radius of a water molecule (*see* **Note 2**).

The computation is performed through three successive tasks, namely (a) Construct the weighted Delaunay triangulation for the balls, (b) Extract the dual complex, and (c) Compute the accessible surface area of each atom using a reduced Inclusion–exclusion equation that maps to the simplices of the dual complex. This process is illustrated in 2D in Fig. 1. The three subsections below provide the details needed to implement this procedure.

3.1 Delaunay Triangulation of a Union of Balls

Our implementation of the Delaunay triangulation is based on the randomized incremental algorithm described in ref. 52. Following the paper's recommendations, we use a minimalist approach to store the triangulation in a linear array of tetrahedrons.

For each tetrahedron, we store the indices of its four vertices, the indices of the four neighboring tetrahedrons, and the position of the opposite vertex in the vertex list of each neighboring tetrahedron. For each vertex, we use four double-precision real numbers for the coordinates and the radius of the corresponding sphere. The triangles and edges are implicit in this representation. We start the procedure with an "infinite" tetrahedron defined by adding four additional balls with their centers at "infinity" (in practice far enough so that the centers of all balls fall inside the corresponding "infinite" tetrahedron). The triangulation is then constructed incrementally, by adding one ball at a time (*see* **Note 3**).

Let N be the number of balls, and let D_i be the Delaunay triangulation of the four balls at infinity together with B_1, B_2, …, B_i.

The algorithm proceeds by iterating three steps:

For i from 1 to N do

1. *Find tetrahedron t in D_{i-1} that contains the center c_i of ball B_i.*
2. *Add c_i to decompose t into four tetrahedrons.*
3. *Flip locally non-Delaunay triangles attached to c_i.*

End.

The first step is implemented using the jump-and-walk technique proposed by Mücke and colleagues [53]. Note that in this step, the ball may be discarded if it is found to be *redundant*. A ball B_i is deemed to be redundant if it is fully included inside the union of other balls. **Step 3** follows the algorithm proposed by Edelsbrunner and Shah [52]. A flip in this step replaces two tetrahedrons

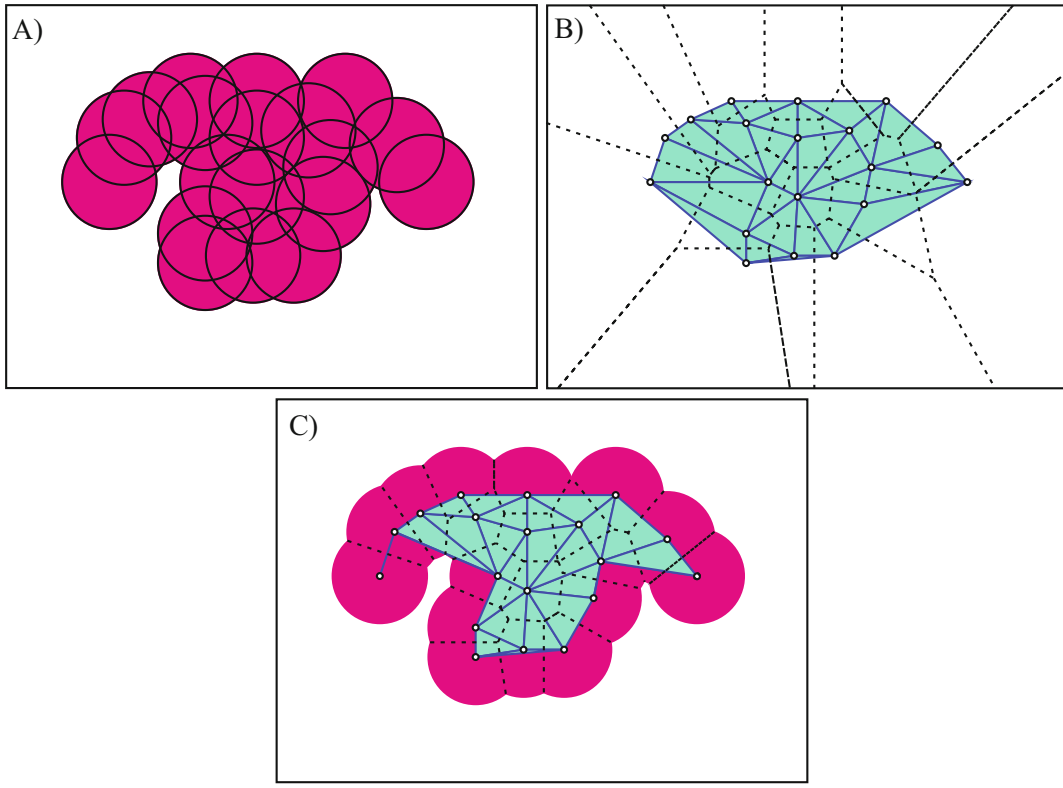


Fig. 1 Voronoi decomposition, Delaunay triangulation, and dual complex of a set of disks in the plane. Given a finite set of disks (**a**), the Voronoi diagram decomposes the plane into regions, one per disk, such that any point in the region V_i assigned to disk B_i is closer to that disk than to any other disk, where the distance from a point M to the disk B_i is defined as $d(M,B_i)^2 = d(M,z_i)^2 - r_i^2$, where z_i and r_i are the center and radius of B_i, respectively. (**b**) The boundaries of those regions are shown as *dashed lines*. The dual Delaunay triangulation is obtained by drawing edges between the centers of the *circles* corresponding to neighboring Voronoi regions. (**c**) We restrict the Voronoi diagram to within the portion of the plane covered by the disks and get a decomposition of the union into convex regions. To draw the dual complex of the disks we limit ourselves to edges and triangles between centers whose corresponding restricted Voronoi regions have a nonempty common intersection

by three or three tetrahedrons by two. The fact that any arbitrary ordering of the flips will successfully repair the Delaunay triangulation is nontrivial but has been established by Edelsbrunner and Shah [52].

Once all the balls have been inserted, we remove all the tetrahedrons that have at least one vertex corresponding to one of the balls placed at infinity.

The final Delaunay complex **DT** is fully defined by the list of the tetrahedrons it contains. Each tetrahedron includes four facets, six edges, and four vertices. The complete list of tetrahedrons, facets, edges, and vertices defines the *simplices* of **DT**. Note that most facets, edges, and vertices are shared by two or more tetrahedrons. Finally, the collection of facets that only belong to one tetrahedron in **DT** forms the convex hull of the set of centers of the balls.

3.2 Generating the Dual Complex of a Union of Balls

The Voronoi diagram is the dual of the Delaunay complex **DT**. It divides the whole space into convex regions, V_i, one per ball B_i in the union. The Voronoi region V_i associated with the ball B_i consists of all points that are at least as close to the center of B_i as to any other balls in the union, as illustrated in Fig. 1. It is a convex polyhedron obtained as the common intersection of finitely many closed half-spaces, one per ball B_j, such that the line segment joining the centers of B_i and B_j belongs to **DT**. It follows that the Voronoi regions decompose the union of balls B_i into convex regions of the form $B_i \cap V_i$ (*see* Fig. 1). Computing the surface area of the union of balls can then be reformulated as computing the surface areas of all convex regions $B_i \cap V_i$, which is a much simpler problem, as those regions do not overlap. In addition, the convex region $B_i \cap V_i$ is fully defined by the ball B_i and its neighboring balls B_j such that the Voronoi region V_j has a common facet with V_i within the union of balls. Those balls B_j are readily identified as the line segment joining the centers of B_i and B_j forms an edge in the dual complex **K**, a subset of the Delaunay triangulation **DT**, defined below.

Given the Delaunay triangulation **DT** of the centers of the balls in the union, we identify first all simplices in **DT** that are *critical*. We call S a *critical* simplex of **DT** if the balls defining S have a nonempty common intersection. Detailed expressions for the geometric tests that establish if two, three, or four balls intersect or not can be found in [50, 54]. The dual complex $\mathbf{K} \subset \mathbf{DT}$ is then defined as the list of all critical simplices in **DT**. Note that the simplices of **DT** that do not belong to **K** are also interesting, as they define the cavities and pockets within the union of balls [55–57].

3.3 Computing the Individual Accessible Surface Areas of the Balls

A simplex S in the dual complex **K** can be interpreted abstractly as a collection of balls with a nonempty intersection, one ball if it is a vertex, two if it is an edge, etc. As such, it makes sense to speak about $A(S)$, the surface area of the intersection of the balls that define S. The core result of the Alpha Shape theory of Edelsbrunner [39] is that the surface area of a union of balls can be expressed exactly as an inclusion–exclusion formula over all simplices in the corresponding dual complex **K**:

$$A\left(\bigcup_i B_i\right) = \sum_{S \in K} (-1)^{dim(S)} A(S) \tag{4}$$

Here, $dim(S) = card(S) - 1$, i.e., the number of balls in S minus 1. This result overcomes past difficulties by implicitly reducing higher-order to lower-order overlaps. An added advantage of Eq. 4 is that the balls in each term form a unique geometric configuration so that the analytic calculation of the surface area can be done without case analysis.

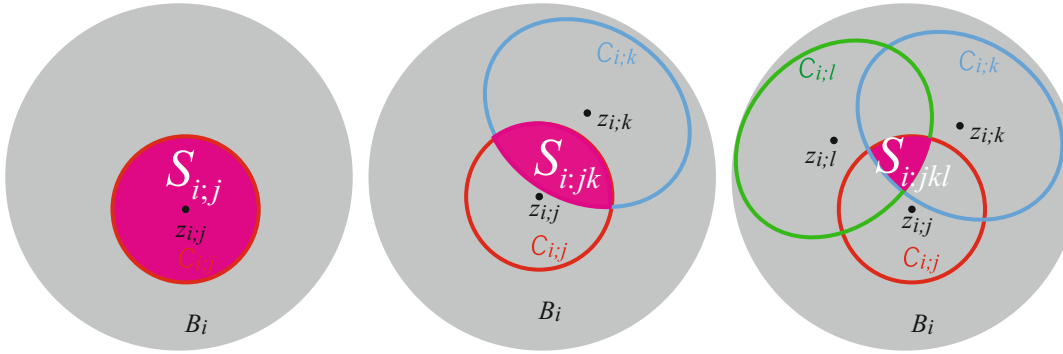


Fig. 2 Intersection of two (*left*), three (*center*), and four (*right*) spheres viewed on the flattened surface of a ball B_i

One way to arrive at this formula is to consider a ball B_i with center z_i and radius r_i, and to observe that its contribution to the total area of the union of balls is the area of the entire ball, $4\pi r_i^2$, minus the portion covered by caps of the form $B_i \cap B_j$, such that $z_i z_j$ forms an edge of the dual complex **K**. The surface area of this portion is computed as the sum of the surface area of each cap, minus the portion covered by the intersection of three caps of the form $B_i \cap B_j \cap B_k$ such that $z_i z_j z_k$ forms a triangle of the dual complex **K**. Finally, the surface area of this portion is the area of the intersection of three caps, minus the portion covered by the intersection of four caps of the form $B_i \cap B_j \cap B_k \cap B_l$ such that $z_i z_j z_k z_l$ forms a tetrahedron of the dual complex **K**. The key to the success of the Alpha Shape theory is that no additional higher terms need to be considered. The whole procedure is illustrated in Fig. 2.

Finally, we note that detailed expressions for the surface areas of the intersections of two, three, and four balls can be found in ref. 40.

3.4 Computing the Nonpolar Contribution to the Solvation Free Energy

The nonpolar part of the solvation free energy of the protein is computed as a weighted sum of the accessible surface areas of all its representing balls (*see* Eq. 2), where the weights are the Atomic Solvation Parameters, defined in Table 1.

4 Notes

1. Unfortunately, PDB files can be difficult to process and it is expected that you do a significant amount of preprocessing prior to using the information they contain. As part of this preprocessing, you should consider at least the following points. (a) *Identify the chain(s) you are interested in*. The PDB file may contain information about a protein complex, while you may be only interested in one subunit. Note that each subunit is

identified with a chain label in the PDB file. Reversely, the PDB file may contain the information about a protein in a monomeric form, while you are interested in the biologically relevant multimeric form. PDB files usually contain information about the mathematical operations that need to be performed to generate the multimer, but it is left to you to perform those operations. (b) *Setting a rule for missing atoms.* Experimental structures may not be complete, as part of the structure may be too flexible to be observed, such as flexible loops, or the terminal groups of long amino acids at the surface of the proteins. You may ignore those missing atoms, or decide to use a modeling program to generate their possible location. (c) *Dealing with alternate configurations.* In addition to missing atoms, the PDB file may contain multiple conformations for some parts of the molecule. These multiple conformations, mostly observed for side-chains, relate to ambiguities in the experimental data. Usually, an occupancy factor is provided for each conformation and it is usually best to select the conformation with the highest factor. (d) *NMR structures: using the average model?* NMR spectroscopy provides indirect measurements on the protein structure of interest, usually a set of short-range interatomic distances. Many modeling techniques generate a collection of models for the structure that are compatible with those distances, as well as an average structure based on this collection. Both are usually provided in the PDB. It is strongly recommended to use one of the models instead of the average structure, as the latter is a simple geometric mean of the models that often has poor stereochemistry.

2. There is no real consensus in computational biology as to which surface of the union of balls representing a protein best relates to the physical properties of the molecule. Three models are widely used, namely, the *van der Waals surface*, the *molecular surface*, and the *solvent accessible surface*, with the latter usually preferred for computing solvation free energies. Lee and Richards [25] defined the *solvent accessible surface* of a molecule as the loci of the center of a probe sphere with radius R_w as it rolls over the van der Waals surface. The value of R_w is usually set to 1.4 Å as it approximates the size of a water molecule. It can be shown that the accessible surface is also the boundary of the union of balls $\cup B_w$, where B_w are "hydrated" balls representing the atoms, i.e., the balls whose vdW radii have been increased by R_w. Note that values for R_w vary from 1.2 to 1.8 Å in the literature.
3. The standard algorithm for building the Delaunay triangulation of a set of balls proceeds incrementally, by adding one ball at a time. Before starting the construction, the balls are re-indexed with a random permutation of the order in which they appear in the input file. The randomization preprocessing in this

algorithm guarantees an expected theoretical running time of $O(Nlog(N) + N^2)$ in the worst case, where N is the number of balls [52]. In practice, however, a very different behavior is observed for a very large dataset. Inherent to their nature, randomized algorithms access the data structures they maintain randomly, and random access works poorly with memory hierarchies available on modern computers. Virtual memory operating systems cache recently used data in memory, under the assumption that they are more likely to be used again soon. Randomized algorithms violate this assumption; they consequently perform poorly as the data structure exceeds the cache size. A simple solution is to insert points in an order that improves locality. Interestingly, the order in which atoms are stored in a PDB file is inherently local. In most cases, two consecutive atoms either belong to the same amino acid or to two sequential amino acids that are in contact. The construction of the Delaunay triangulation for a protein is therefore significantly faster if the order of the atoms is not randomized [40].

Acknowledgment

Patrice Koehl acknowledges support from the Ministry of Education of Singapore through Grant Number: MOE2012-T3-1-008.

References

1. The UniProt Consortium (2015) UniProt: a hub for protein information. Nucleic Acids Res 43:D204–D212
2. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H (2000) The Protein Data Bank. Nucleic Acids Res 28:235–242
3. Dror RO, Dirks RM, Grossman JP, Xu H, Shaw DE (2012) Biomolecular simulation: a computational microscope for molecular biology. Annu Rev Biophys 41:429–452
4. Koehl P, Levitt M (1999) A brighter future for protein structure prediction. Nat Struct Biol 6:108–111
5. Friesner RA, Abel R, Goldfeld DA, Miller EB, Murrett CS (2013) Computational methods for high resolution prediction and refinement of protein structures. Curr Opin Struct Biol 23:177–184
6. Jothi A (2012) Principles, challenges and advances in ab initio protein structure prediction. Protein Pept Lett 19:1194–1204
7. Lindorff-Larsen K, Piana S, Dror RO, Shaw DE (2011) How fast-folding proteins fold. Science 334:517–520
8. Piana S, Lindorff-Larsen K, Shaw DE (2013) Atomic-level description of ubiquitin folding. Proc Natl Acad Sci U S A 110:5915–5920
9. Nguyen H, Maier J, Huang H, Perrone V, Simmerling C (2014) Forlding simulations for proteins with diverse topologies are accessible in days with a physics-based force field and implicit solvent. J Am Chem Soc 136:13959–13962
10. Drexler KE (1981) Molecular engineering: an approach to the development of general capabilities for molecular manipulation. Proc Natl Acad Sci U S A 78:5275–5278
11. Pabo C (1983) Designing proteins and peptides. Nature (London) 301:200
12. Khoury GA, Smadbeck J, Kieslich CA, Floudas CA (2014) Protein folding and de novo protein design for biotechnological applications. Trends Biotechnol 32:99–109
13. Kiss G, Celebi-Olcum N, Moretti R, Baker D, Houk KN (2013) Computational enzyme design. Angew Chem 52:5700–5725
14. Pantazes RJ, Grisewood MJ, Maranas CD (2011) Recent advances in computational protein design. Curr Opin Struct Biol 21:467–472

15. Strauch EM, Fleishman SJ, Baker D (2014) Computational design of a pH-sensitive IgG binding protein. Proc Natl Acad Sci U S A 111:675–680
16. Damborsky J, Brezovsky J (2014) Computational tools for designing and engineering enzymes. Curr Opin Chem Biol 19:8–16
17. King NP, Bale JB, Sheffler W, McNamara DE, Gonen S, Gonen T, Yeates TO, Baker D (2014) Accurate design of co-assembling multi-component protein nanomaterials. Nature (London) 510:103–108
18. Lai Y-T, Reading E, Hura GL, Tsai K-L, Laganowsky A, Asturias FJ, Tainer JA, Robinson CV, Yeates TO (2014) Structure of a designed protein cage that self-assembles into a highly porous cube. Nat Chem 6:1065–1071
19. Gogonea V, Suarez D, van der Vaart A, Merz KM Jr (2001) New developments in applying quantum mechanics to proteins. Curr Opin Struct Biol 11:217–223
20. Raha K, Peters MB, Wang B, Yu N, Wollacott AM, Westerhoff LM, Merz KM Jr (2007) The role of quantum mechanics in structure-based drug design. Drug Discov Today 12:725–731
21. Li Z, Yang Y, Zhan J, Dai L, Zhou Y (2013) Energy functions in de novo protein design: current challenges and future prospects. Annu Rev Biophys 42:315–335
22. Eisenberg D, McLachlan A (1986) Solvation energy in protein folding and binding. Nature (London) 319:199–203
23. Richards FM (1977) Areas; volumes; packing; and protein-structure. Annu Rev Biophys Bioeng 6:151–176
24. Lum K, Chandler D, Weeks JD (1999) Hydrophobicity at small and large length scales. J Phys Chem B 103:4570–4577
25. Lee B, Richards FM (1971) Interpretation of protein structures: estimation of static accessibility. J Mol Biol 55:379–400
26. Shrake A, Rupley J (1973) Environment and exposure to solvent of protein atoms in lyzozyme and insulin. J Mol Biol 79:351–371
27. Legrand SM, Merz KM (1993) Rapid approximation to molecular-surface area via the use of Boolean logic and look-up tables. J Comput Chem 14:349–352
28. Wang H, Levinthal C (1991) A vectorized algorithm for calculating the accessible surface area of macromolecules. J Comput Chem 12:868–871
29. Futamura N, Alura S, Ranjan D, Hariharan B (2004) Efficient parallel algorithms for solvent accessible surface area of proteins. IEEE Trans Parallel Dist Syst 13:544–555
30. Wodak SJ, Janin J (1980) Analytical approximation to the accessible surface-area of proteins. Proc Natl Acad Sci U S A 77:1736–1740
31. Cavallo L, Kleinjung J, Fraternali F (2003) POPS: a fast algorithm for solvent accessible surface areas at atomic and residue level. Nucleic Acids Res 31:3364–3366
32. Street AG, Mayo SL (1998) Pairwise calculation of protein solvent-accessible surface areas. Fold Des 3:253–258
33. Connolly M (1983) Analytical molecular surface calculation. J Appl Cryst 16:548–558
34. Richmond TJ (1984) Solvent accessible surface-area and excluded volume in proteins. Analytical equations for overlapping spheres and implications for the hydrophobic effect. J Mol Biol 178:63–89
35. Dodd L, Theodorou D (1991) Analytical treatment of the volume and surface area of molecules formed by an arbitrary collection of unequal spheres intersected by planes. Mol Phys 72:1313–1345
36. Irisa M (1996) An elegant algorithm of the analytical calculation for the volume of fused spheres with different radii. Comput Phys Commun 98:317–338
37. Gibson K, Scheraga H (1987) Exact calculation of the volume and surface area of fused hard-sphere molecules with unequal atomic radii. Mol Phys 62:1247–1265
38. Kratky KW (1978) Area of intersection of n equal circular disks. J Phys A Math Gen 11:1017–1024
39. Edelsbrunner H (1995) The union of balls and its dual shape. Discrete Comput Geom 13:415–440
40. Mach P, Koehl P (2011) Geometric measures of large biomolecules: surface, volume, and pockets. J Comput Chem 32:3023–3038
41. Li J, Mach P, Koehl P (2013) Measuring the shapes of macromolecules – and why it matters. Comput Struct Biotechnol J 8: e201309001
42. Bondi A (1964) vdW volumes and radii. J Phys Chem 68:441–451
43. Rowland RS, Taylor R (1996) Intermolecular non-bonded contact distances in organic crystal structures: comparison with distances expected from vdW radii. J Phys Chem 100:7384–7391
44. Tsai J, Taylor R, Chothia C, Gerstein M (1999) The packing density in proteins: standard radii and volumes. J Mol Biol 290:253–266
45. Chothia C (1975) Structural invariants in protein folding. Nature (London) 254:304–308
46. Fauchere J-L, Pliska V (1983) Hydrophobic parameters of amino acid side-chains from the

partitioning of N-acetyl-amino-acid amides. Eur J Med Chem Chim Ther 18:369–375

47. Sharp KA, Nicholls A, Friedman R, Honig B (1991) Extracting hydrophobic free energies from experimental data : relationship to protein folding and theoretical models. Biochemistry 30:9686–9687
48. Holtzer A (1992) The use of Flory-Huggins theory in interpretating partitioning of solutes between organic liquids and water. Biopolymers 32:711–715
49. Koehl P, Delarue M (1994) Polar and non polar atomic environment in the protein core: implications for folding and binding. Proteins 20:264–278
50. Edelsbrunner H, Mucke EP (1994) Three-dimensional alpha shapes. ACM Trans Graph 13:43–72
51. Edelsbrunner H, Fu P (1994) Measuring space filling diagrams and voids (UIUC-BI-MB-94-01)
52. Edelsbrunner H, Shah NR (1996) Incremental topological flipping works for regular triangulations. Algorithmica 15:223–241
53. Mucke EP, Saias I, Zhu B (1999) Fast randomized point location without preprocessing in two- and three-dimensional Delaunay triangulations. Comput Geom Theor Appl 12:63–83
54. Edelsbrunner H (1992) Weighted alpha shapes (UIUC-CS-R-92-1760)
55. Edelsbrunner H, Facello MA, Liang J (1998) On the definition and construction of pockets in macromolecules. Discrete Appl Math 88:83–102
56. Liang J, Edelsbrunner H, Fu P, Sudhakar PV, Subramaniam S (1998) Analytical shape computation of macromolecules. II. Inaccessible cavities in proteins. Proteins 33:18–29
57. Liang J, Edelsbrunner H, Woodward C (1998) Anatomy of protein pockets and cavities: measurement of binding site geometry and implications for ligand design. Protein Sci 7:1884–1897
58. Petitjean M (1994) On the analytical calculation of Van der Waals surfaces and volumes: some numerical aspects. J Comput Chem 15:507–523
59. Fraternali F, Cavallo L (2002) Parameter optimized surfaces (POPS): analysis of key interactions and conformational changes in the ribosome. Nucleic Acids Res 30:2950–2960

Chapter 6

Modeling Binding Affinity of Pathological Mutations for Computational Protein Design

Miguel Romero-Durana*, Chiara Pallara*, Fabian Glaser, and Juan Fernández-Recio

Abstract

An important aspect of protein functionality is the formation of specific complexes with other proteins, which are involved in the majority of biological processes. The functional characterization of such interactions at molecular level is necessary, not only to understand biological and pathological phenomena but also to design improved, or even new interfaces, or to develop new therapeutic approaches. X-ray crystallography and NMR spectroscopy have increased the number of 3D protein complex structures deposited in the Protein Data Bank (PDB). However, one of the more challenging objectives in biological research is to functionally characterize protein interactions and thus identify residues that significantly contribute to the binding. Considering that the experimental characterization of protein interfaces remains expensive, time-consuming, and labor-intensive, computational approaches represent a significant breakthrough in proteomics, assisting or even replacing experimental efforts. Thanks to the technological advances in computing and data processing, these techniques now cover a vast range of protocols, from the estimation of the evolutionary conservation of amino acid positions in a protein, to the energetic contribution of each residue to the binding affinity. In this chapter, we review several existing computational protocols to model the phylogenetic, structural, and energetic properties of residues within protein–protein interfaces.

Key words Protein–protein interactions, Hot-spots identification, Interface prediction, Evolutionary conservation, Protein–protein docking, Biomolecular dynamics simulation, In silico alanine scanning, pyDock, AMBER package, ConSurf

1 Introduction

One of the current goals of proteomics is to predict and characterize protein–protein complex interfaces. Access to such information is highly valuable as it helps to elucidate large protein interaction networks, increases the current knowledge on biochemical pathways, improves comprehensive description of disease pathogenesis,

*Corresponding authors.

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_6,

and finally suggests putative new therapeutic targets [1–3]. Moreover, the use of computational approaches offers faster and more cost-efficient procedures in comparison to experimental methods such as co-immunoprecipitation, affinity chromatography, yeast two-hybrid, or mass spectroscopy.

In this chapter, we review several computational methods that exploit phylogenetic, structural, and energetic properties of interface residues for the computational design of protein complexes or the characterization of pathological mutations involved in protein–protein interfaces. First, we describe two methods that do not need the structure of the protein–protein complex, namely ConSurf [4–7] and Normalized Interface Propensity (NIP) [8]. ConSurf identifies functionally and structurally important residues (e.g., involved in enzymatic activity, in ligand binding or protein-protein interactions) [9] on a protein by estimating the degree of conservation of each amino acid site among their close sequence homologues. NIP computes the tendency of a given residue to be located at the interface, from rigid-body docking poses evaluated by pyDock scoring function [10] (based on accessible surface area-based desolvation, coulombic electrostatics, and van der Waals energy). Then, we describe two other protocols which require previous knowledge of the complex structure: residue contribution to binding energy computed with pyDock, and in silico Alanine (Ala) scanning, based on molecular dynamics simulations with AMBER14 package [11] and binding energy calculations using the MM-GBSA method [12]. The use of these methods is illustrated on one example, the MEK1-BRAF complex (PDB ID 4MNE) [13], in which several pathological mutations are annotated [14].

2 Materials

2.1 ConSurf Server

1. ConSurf Server is a bioinformatics tool that estimates the evolutionary conservation of amino acid positions in protein molecules based on the phylogenetic relations among close homologous sequences. It can be found at http://consurf.tau.ac.il.

2.2 PyDock

1. PyDock is docking package freely available to academic users. Go to pyDock download web page http://life.bsc.es/pid/pydock/get_pydock.html [15] and fill in the form with the requested information. pyDock team will quickly send you a copy of the application and instructions to install it.

2.3 FTDock

1. From the FTDock [16] web page http://www.sbg.bio.ic.ac.uk/docking/download.html, download file *gnu_licensed_3D_Dock.tar.gz* to the folder of your choice.
2. From the FFTW web page http://www.fftw.org/download.html, download file *fftw-2.1.5.tar.gz*.

3. Move to the folder where you have downloaded the file *fftw-2.1.5.tar.gz* and unpack the package with the following commands:

   ```
   cd folder-where-fftw-2.1.5.tar.gz-has-been-downloaded
   gunzip fftw-2.1.5.tar.gz
   tar xvf fftw-2.1.5.tar
   ```

4. Move into directory *fftw-2.1.5* and compile the library:

   ```
   cd fftw-2.1.5
   ./configure
   make
   ```

5. Move to the folder where you have downloaded *gnu_licensed_3D_Dock.tar.gz* and unpack FTDock package.
6. Move to the unpacked folder *3D_Dock/progs*. Edit file *Makefile* and set the correct complete path to the *fftw-2.1.5* directory. This is done by setting the variable FFTW_DIR on line 15. You should also check the value of the CC_FLAGS variable, and make it fit to your system (e.g., for a x86_64 Linux system, CC_FLAGS variable has been modified and set to '-O -m64'.
7. Type the following command:

   ```
   make
   ```

8. You should now have the executable files *ftdock*, *build*, and *randomspin* available. Optional: Edit your *.bashrc* file to include *3D_Dock/progs* folder in your system path (PATH variable).

2.4 UCSF Chimera Molecular Viewer

UCSF Chimera [17] is a highly extensible program for interactive visualization, molecular structure analysis and high-quality images generation. Here are the instructions to install UCSF Chimera Molecular viewer:

1. Go to UCSF Chimera Molecular viewer web page at http://www.cgl.ucsf.edu/chimera.
2. Go to the download session, by clicking on *Download* in the menu on the top-left of the web page, and select the UCSF Chimera Molecular viewer installer appropriate for you platform.
3. Install UCSF Chimera Molecular viewer on your computer following the platform specific installation instructions available on the same page.

2.5 AMBER Package

AMBER is a package of programs for molecular dynamics simulations of proteins and nucleic acids. It is distributed in two parts: AmberTools14 and Amber14. Here are the instructions to install AMBER package:

- Go to the AMBER web page at http://ambermd.org/#Amber14.

- After filling the registration form located on its own section at http://ambermd.org/AmberTools14-get.html, download AmberTools14 clicking on the *Download* button.
- Download the Amber 14 License Agreement, print this form, fill it in, sign and return it to the address given at the bottom of the license agreement. Once the order is processed, download the AMBER program package following the download information you will receive via e-mail.
- Install AMBER on your machine and compile the source code format using Fortran 95, C or C++ compilers following the instructions in the Amber Reference Manual at http://ambermd.org/doc12/Amber14.pdf.

3 Methods

3.1 Analysis of Residue Conservation by ConSurf

1. Go to ConSurf web server page at http://consurf.tau.ac.il. Then, ConSurf web server will ask you several questions regarding the computation you want to run.
2. To the question *Analyze Nucleotides or Amino Acids?* select *Amino-Acids* option.
3. To the question *Is there a known protein structure?* select *Yes* option.
4. Provide the PDB ID (e.g., 4MNE) of the structure you want to analyze or upload your own PDB file, browsing to its location. Press *Next* button.
5. Select the chain identifier of the molecule to be analyzed.
6. Indicate whether there is a multiple sequence alignment (MSA) to upload. If there is not, ConSurf server will generate it. You may set the parameters ConSurf server will use to generate the MSA. For this work, ConSurf server has been run with default parameters.
7. At the bottom of the page, fill the *Job title* field to identify the job.
8. Fill the *User E-Mail* field, check the *Send a link to the results by e-mail* check-box and click the *submit* button. Thus, ConSurf server will send you an e-mail with a link to the results when it has finished.
9. Open the e-mail sent by ConSurf and go to the results page link.
10. Click on the *Download all Consurf outputs in a click!* link, save the ConSurf results file and unzip it.
11. Open *consurf.grades* file. From all the columns of the file, focus on three: 3LATOM, SCORE, and COLOR. The 3LATOM

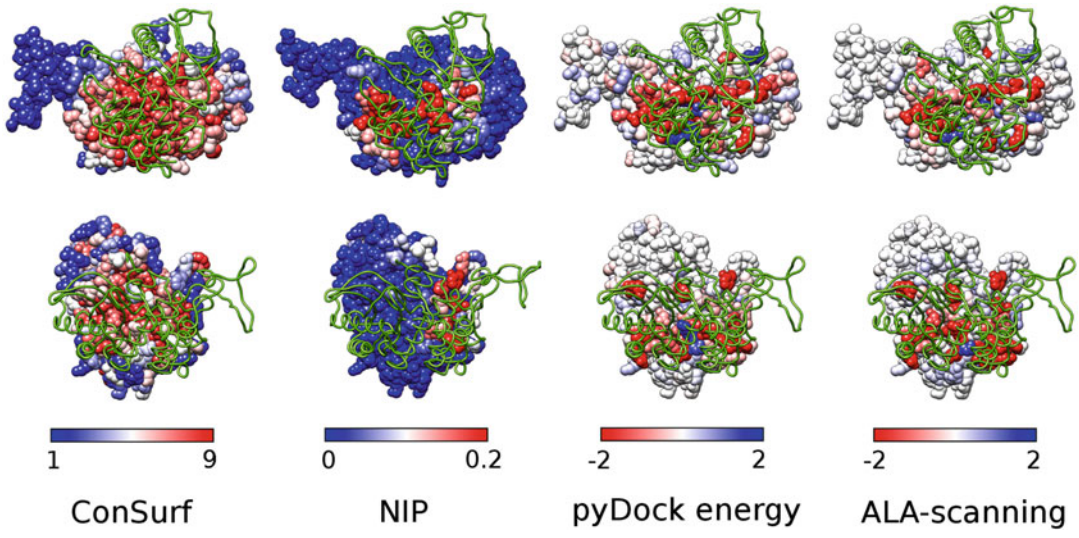


Fig. 1 MEK1–BRAF interface characterization. MEK1 and BRAF interface characterization using different computational techniques (first and second line, respectively): ConSurf evolutionary conservation, pyDock NIP calculation, pyDock binding energy decomposition, binding free energy change ($\Delta\Delta G$) estimated by in silico alanine scanning

column contains an id code of the analyzed residues. The SCORE column contains the computed normalized conservation score. Lower scores (more negative) correspond to more conserved residues, while higher scores (more positive) correspond to less conserved residues. A similar information is shown in column COLOR where, in order to ease visualization of the results, the continuous conservation scores have been partitioned into nine different bins, with bin 9 representing the most conserved positions and bin 1 the most variable positions. It is important to remark that neither the SCORE values nor the COLOR values indicate absolute magnitudes of conservation, but rather the relative degree of conservation of a given residue in the specific protein under study (i.e., neither SCORE nor COLOR values of residues of different proteins are generally comparable).

12. ConSurf provides two PDB files where the SCORE and COLOR values are assigned to the bfactor field. This is quite useful in order to get a picture of which residues are more conserved. With your favorite molecular visualization application open **.pdb_With_Conservation_Scores.pdb* and **.pdb_ATOMS_section_With_Consurf* files for displaying SCORE and COLOR values, respectively (*see* Fig. 1).

3.2 Prediction of Binding Hot-Spots by NIP

NIP computation can be divided in four different steps: (1) initial setup, where the receptor and ligand PDB files of the complex are preprocessed in order to generate the input files that FTDock and pyDock require, (2) sampling phase, where FTDock generates a set

```
[receptor]
pdb      =        3eqi.pdb
mol      =        A
newmol   =        A

[ligand]
pdb      =        4mne.pdb
mol      =        B
newmol   =        B
```

Fig. 2 Example of pyDock input file. The input file is typically divided into two sections, *[receptor]* and *[ligand]*, designed to specify the variables related to the receptor and ligand, respectively. The *pdb* line defines the PDB file name. The *mol* line specifies the original chain name in each PDB file, whereas the *newmol* indicates the new one in the pyDock output files. Please be aware that the *newmol* chain names must be different for the receptor and the ligand

of docking poses, (3) scoring phase, where pyDock dockser module scores and ranks the poses generated by FTDock, and (4) NIP computation, where the first 100 ranked docking poses (those with lower binding energy) are selected from the whole set of generated docking poses, and pyDock patch module is used to compute the NIP values.

Next, we describe each one of these phases in more detail.

1. Initial setup.
   (a) Create a project folder and move to it.
   (b) From the PDB website, download the receptor and ligand structures, e.g., download the PDB files of receptor (3EQI) and ligand (4MNE) into the project_folder (*see* **Note 1**).
   (c) Create pyDock ini file: open your favorite text editor and create the file 4mne.ini as shown in Fig. 2.
   (d) Run pyDock *setup* module:

   pydock3 4mne setup

   (e) pyDock *setup* module should have generated several new files (*see* Table 1).
2. FTDock sampling.
   (a) Run FTDock:

   ftdock -static 4mne_rec.pdb -mobile 4mne_lig.pdb -calculate_grid 0.7 -angle_step 12 -internal -15 -surface 1.3 -keep 3 -out 4mne.ftdock

   (b) When FTDock is finished, you should have a new file named *4mne.ftdock* in the folder.
3. Scoring.

   In this phase, the docking poses generated in the sampling phase are scored and ranked with pyDock *dockser* module.

Table 1
pyDock modules input and output files.

Module name	Input files	Output files
setup	docking_name.ini	docking_name_rec.pdb docking_name_lig.pdb docking_name_rec.pdb.H docking_name_lig.pdb.H docking_name_rec.pdb.amber docking_name_lig.pdb.amber
rotftdock	docking_name_rec.pdb docking_name_lig.pdb	docking_name.rot
rotzdock	docking_name_rec.pdb docking_name_lig.pdb	docking_name.rot
dockser	docking_name_rec.pdb docking_name_lig.pdb docking_name_rec.pdb.H docking_name_lig.pdb.H docking_name_rec.pdb.amber docking_name_lig.pdb.amber docking_name.rot	docking_name.ene
patch	docking_name_rec.pdb docking_name_lig.pdb docking_name.rot docking_name.ene	docking_name.recNIP docking_name.rec.pdb.nip docking_name.ligNIP docking_name.lig.pdb.nip
bindEy	docking_name.ini	docking_name_rec.pdb docking_name_lig.pdb docking_name_rec.pdb.H docking_name_lig.pdb.H docking_name_rec.pdb.amber docking_name_lig.pdb.amber docking_name.rot docking_name.ene
resEnergy	docking_name_rec.pdb docking_name_lig.pdb docking_name_rec.pdb.H docking_name_lig.pdb.H docking_name_rec.pdb.amber docking_name_lig.pdb.amber docking_name.rot	docking_name.receptor.residueEne docking_name.ligand.residueEne docking_name.receptor.atomEne docking_name.ligand.atomEne

(a) Run pyDock *rotftdock* module:

pydock3 4mne rotftdock

(b) There should be now a new file *4mne.rot*. Each line in this file represents a rotation and translation matrix. FTDock *4mne.rot* file should have 10,000 different lines.

(c) Score and rank FTDock poses by running pyDock *dockser* module:

pydock3 4mne dockser

(d) Once *dockser* module has finished, it should have created file *4mne.ene* with 10002 different lines (*see* **Note 2** for a detailed description of this file).

4. NIP computation.
   (a) Run pyDock *patch* module:

   pydock3 4mne patch

   (b) *4mne.recNIP* and *4mne.ligNIP* files should have been created. These files show the computed NIP value for each residue of receptor and ligand, respectively. Those residues with NIP values greater than 0.2 are predicted to be hot-spots.

   (c) For visualization proposes, *patch* module output includes two PDB files, with extension *.pdb.nip, where the NIP values have been assigned to the bfactor field. With your favorite molecular visualization application open *_rec.pdb.nip or *_lig.pdb.nip files for displaying the NIP values of receptor and ligand, respectively (*see* Fig. 1).

3.3 Computation of Binding Energy per Residue with pyDock

1. Create a folder for computing residue binding energy.
2. From the PDB website, download the structure of a protein-protein complex, e.g., BRAF/MEK1 (PDB ID 4MNE).
3. Create pyDock ini file: Open your favorite text editor and create the *4mne.ini* file specifying receptor and ligand subunits.
4. Compute pyDock binding energy by running the following command:

   pydock3 4mne bindEy
5. pyDock should have generated several new files. Please *see* Table 1 to confirm.
6. Run pyDock residue energy module:

   pydock3 4mne resEnergy
7. The module should have created for ligand and receptor *.*atomEne* and *.*residueEne* files with the contribution to the binding energy of each individual atom and residue, respectively.
8. You may get a graphical representation of the residue binding energy (*see* Fig. 1), by assigning the binding energy values given in *.*residueEne* files to the bfactor field of the corresponding PDB file of the target molecules.

3.4 In-Silico Alanine Scanning with AMBER

The Alanine scanning workflow can be divided into three different steps: (1) the preparation of the PDB files for both the wild type and the mutated structures, (2) the molecular dynamics simulation

of the wild type complex and (3) the binding free energy calculation on both the wild type and the mutated structures.

1. Wild type and mutated structures PDB files preparation.
   (a) Start a new session of UCSF Chimera Molecular viewer and open *4MNE* PDB file clicking on *File* → *Fetch by ID* entering *4mne* as *PDB ID* in the new window and then clicking on the *Fetch* button. Delete all chains but A and B, and all existing water molecules from the system.
   (b) Build missing segments starting the Chimera interface to MODELLER. Click on *Tools* → *Structure Editing* → *Model/Refine Loops*. In the new window, select *all missing structure* as model/remodel option and *one* as both number of residues adjacent to missing region allowed to move and number of models to generate. Write the MODELLER license key and start the rebuilding by clicking on *OK*. The MODELLER license key is freely available only for academic use and can be requested at the MODELLER web page https://salilab.org/modeller/registration.html, filling up the license agreement and clicking on *agreed and accepted* button.
   (c) Save the PDB files of the complex and each subunit in the wild type form. Go to *File* → *SavePDB*. In the new window enter *MEK1-BRAF.pdb* as file name of the refined complex structure and finally click on *Save*. Select each subunit of the complex by its chain name from *Select* → *Chain*. Go to *File* → *SavePDB*, specify the subunit new file name (i.e., *MEK1.pdb* for chain A and *BRAF.pdb* for chain B), pick the *save selected atom only* option and finally click on *Save*.
   (d) Save the complex and the subunit PDB files for each mutant. Start a new session of UCSF Chimera Molecular viewer, open *MEK1-BRAF.pdb* file, select only one residue to be mutated then go to *Tools* → *Structure Editing* → *Rotamers*, choose *ALA* as rotamer type and click on *OK*. Save the resulting mutated complex structure going to *File* → *Save PDB* and specifying the mutation in the new file name (e.g., *MEK1-BRAF_F468A.pdb*). Finally, select the mutated subunit structure only and save it in a separate file (e.g., *BRAF_F468A.pdb*). Repeat the same protocol for each BRAF and MEK1 residue to be mutated.
   e) Edit all *MEK1-BRAF.pdb* and MEK1 *.pdb* files (both wild type and mutated). Rename MG residue to MG2 and convert ACP molecule to ATP.
2. Molecular dynamics simulation.

```
source leaprc.ff99SB
source leaprc.gaff

#Load ATP parameters
loadamberprep ATP.prep
loadamberparams ATP.frcmod

#Check ATP parameters
check ATP

#Load pdb file
4mne=loadpdb MEK1-BRAF.pdb

#Check pdb structure
check 4mne

#Compute total charge
charge 4mne

#Put an 12A-buffer of TIP3P water around the system
solvateoct 4mne TIP3PBOX 12.0

#Neutralize the system
addions 4mne Na+ 4

#Save topology and coordinate files
saveamberparm 4mne MEK1-BRAF_solv.prmtop MEK1-BRAF_solv.inpcrd

quit
```

Fig. 3 Example of AMBER LEaP input file to build topology and coordinates files of wild type solvated system. The *source* command tells LEaP AMBER tool to execute the start-up script for ff99SB and GAFF force fields. First, ATP parameters are loaded and checked, then *MEK1-BRAF.pdb* file is loaded into a new unit called *4mne*, the structure is checked (i.e., close contacts and bond distances, bond and angle parameters) and the total charge is computed. Then, the system is solvated by adding a truncated octahedral 12 Å box of TIP3P water molecules around the protein, and neutralized by adding four Na+ ions. Finally, the topology and coordinate files are saved in the *prmtop* and *inpcrd* AMBER format, respectively

(a) Download the ATP molecule parameters from the AMBER parameter database (*see* **Note 3**). Go to the AMBER parameter database web page at http://www.pharmacy.manchester.ac.uk/bryce/amber/. Search the row *ATP (revised phosphate parameters)* in the *Cofactors* table and save the *PREP* and *FRCMOD* files as *ATP.prep* and *ATP.frcmod*, respectively.

(b) Modify the ATP atom names in your PDB file to match the atom names in the *ATP.prep* file so that LEaP AMBER tool will be able to match them up.

(c) Create the input files for the MD simulation (topology and coordinate files) using LEaP AMBER tool. Run the

```
#Solvent minimization

&cntrl
imin=1,
maxcyc=1000,
ncyc=500,
ntb=1,
cut=12,
ntr=1,
restraintmask='!:WAT,Na+,Cl-',
restraint_wt=50,
drms=0.01
/
```

Fig. 4 Example of AMBER pmemd input file for solvent minimization. In the input file, *imin = 1* specifies that minimization instead of molecular dynamics will be performed, the parameter *maxcyc* specifies the total number of minimization cycles to be run while *ncyc* specify the number of steepest descent minimization followed by *maxcyc-ncyc* steps of conjugate gradient minimization, *drms* sets the convergence criterion for the energy gradient (in Å). The parameter *ntb = 1* means that a period boundary will be set around the system to maintain a constant volume while *cut* sets the cutoff value (in Å) applied for non-bonded interactions. The flag *ntr = 1* indicates that the positional restraint method is applied for the energy minimization, *restraintmask* specifies the atoms to be restrained (in this cases all but water and ions molecule) and finally *restraint_wt* defines the restraints strength in terms of force constant in kcal mol^{-1} $Å^{-2}$ applied on each restrained atom

input script *tleap-solv.in* (shown in Fig. 3, *see* **Note 4**) using the following command:

$AMBERHOME/bin/tleap -f tleap-solv.in > tleap-solv.out

Flag -f tells tleap to execute the start-up script after-specified.

(d) Run a short solvent minimization step using AMBER *pmemd* input script *min_solv.in* (shown in Fig. 4) and the following input command:

$AMBERHOME/bin/pmemd -i min_solv.in -o min_-solv.out -c MEK1-BRAF_solv.inpcrd -p MEK1-BRAF_-solv.prmtop -r MEK1-BRAF_min.rst -ref MEK1-BRAF_solv.inpcrd

Flag *-i* specifies the input file, *-o* the output file, *-c* the coordinate file, *-p* the parameter and topology file, *-r* the output restart file with coordinates and velocities, and *-ref* the reference coordinates file for positional restraints, if this option is specified in the input file.

(e) Run a 5-step equilibration by which the system temperature is raised from 0 to 300 K, and a gradual relaxation is performed by progressively releasing the initially set positional restraints. The following protocol should be used:

```
#Equilibration (I)

&cntrl
imin=0,
irest=0,
ntx=1,
ntb=1,
cut=12,
ntc=2,
ntf=2,
tempi=0.0,
temp0=300.0,
ntt=3,
gamma_ln=1.0,
nstlim=20000,
dt=0.002,
ntwx=5000,
ntwr=5000,
ntpr=5000,
ntr=1,
restraintmask='!:WAT,Na+,Cl-',
restraint_wt=25,
ig=-1,
/
```

Fig. 5 Example of AMBER pmemd input file for first step equilibration. In the input file, *imin* = *0* specifies that molecular dynamics instead of minimization will be performed, the parameters *irest* = *0* and *ntx* = *1* indicate that only coordinates but no velocity information will be taken from the previous restart file, the flag *ntc* = *2* indicates that all bonds involving H-bonds are constrained by the SHAKE algorithm to eliminate high frequency oscillations in the system while *ntf* = *2* means that all types of forces in the force filed are being calculated except bond interaction involving H-atoms. The parameters *temp0* and *tempi* define the initial and the temperature at which the system is to be kept, respectively; *ntt* = *3* indicates that the temperature Langevin thermostat will be used while *gamma_ln*=*1.0* sets the collision frequency to 1 fs. The flag *nstlim* defines the number of simulation steps, *dt* defines the length of each frame (set at 2 fs, here) while *ntwx*, *ntwr*, *ntpr* define the frequency of data deposition (coordinates, energy, and restart, respectively). Finally *ig* = *−1* sets the random seed based on the current date and time and hence will be different for every run. The meaning of the rest of the parameters listed in the input file was previously explained

- As a first equilibration step, run a 40-ps simulation in isovolume condition applying harmonic restraints to all the protein atoms and heating the system to 300 K. Run *equil1.in* input script (shown in Fig. 5) using the following command:

  $AMBERHOME/bin/pmemd -i equil1.in -o equil1.out -c MEK1-BRAF_min.rst -p MEK1-BRAF_solv.prmtop -r MEK1-BRAF_eq1.rst -ref MEK1-BRAF_min.rst -x MEK1-BRAF_eq1.mdcrd

```
#Equilibration (II)

&cntrl
imin=0,
irest=1,
ntx=5,
ntb=1,
cut=12,
ntc=2,
ntf=2,
tempi=300.0,
temp0=300.0,
ntt=3,
gamma_ln=1.0,
nstlim=10000,
dt=0.002,
ntwx=5000,
ntwr=5000
ntpr=5000,
ntr=1,
restraintmask='!:WAT,Na+,Cl-',
restraint_wt=10,
ig=-1,
/
```

Fig. 6 Example of AMBER pmemd input file for the second step equilibration. In the input file, the flags *ntx* = *5* and *irest* = *1* mean that velocity and coordinate information will be taken from the previous restart file. The meaning of the rest of the parameters listed in the input file was previously explained

- Perform an additional 20-ps step in isothermal-isovolume condition reducing the harmonic restraints to all the protein atoms from 25 to 10 kcal/(mol Å^2). Run *equil2.in* input script (shown in Fig. 6) using the following command:

  $AMBERHOME/bin/pmemd -i equil2.in -o equil2.out -c MEK1-BRAF_eq1.rst -p MEK1-BRAF_solv.prmtop -r MEK1-BRAF_eq2.rst -ref MEK1-BRAF_eq1.rst -x MEK1-BRAF_eq2.mdcrd

- Run another 20-ps step applying the harmonic restraints only to the backbone atoms. Run *equil3.in* input script (shown in Fig. 7) using the following command:

  $AMBERHOME/bin/pmemd -i equil3.in -o equil3.out -c MEK1-BRAF_eq2.rst -p MEK1-BRAF_solv.prmtop -r MEK1-BRAF_eq3.rst -ref MEK1-BRAF_eq2.rst -x MEK1-BRAF_eq3.mdcrd

- Run further 20-ps step decreasing protein backbone restraints to 5 kcal/(mol Å^2). Run *equil4.in* input script (shown in Fig. 8) using the following command:

  $AMBERHOME/bin/pmemd -i equil4.in -o equil4.out -c MEK1-BRAF_eq3.rst -p MEK1-BRAF_solv.

```
#Equilibration (III)

&cntrl
imin=0,
irest=1,
ntx=5,
ntb=2,
ntp=1,
cut=12,
ntc=2,
ntf=2,
tempi=300.0,
temp0=300.0,
ntt=3,
gamma_ln=1.0,
nstlim=10000,
dt=0.002,
ntwx=5000,
ntwr=5000
ntpr=5000,
ntr=1,
restraintmask='@CA,N,C,O',
restraint_wt=10,
ig=-1,
/
```

Fig. 7 Example of AMBER pmemd input file for the third step equilibration. In the input file the flags *ntb = 2* and *ntp = 1* indicate that constant pressure instead of constant volume is applied. The meaning of the rest of the parameters listed in the input file was previously explained

```
#Equilibration (IV)

&cntrl
imin=0,
irest=1,
ntx=5,
ntb=2,
ntp=1,
cut=12,
ntc=2,
ntf=2,
tempi= 300.0,
temp0= 300.0,
ntt=3,
gamma_ln=1.0,
nstlim=10000,
dt=0.002,
ntwx=5000,
ntwr=5000
ntpr=5000,
ntr=1,
restraintmask='@CA,N,C,O',
restraint_wt=5,
ig=-1,
/
```

Fig. 8 Example of AMBER pmemd input file for the fourth step equilibration. The meaning of all the parameters listed in the input file was previously explained

```
#equilibration (V)

&cntrl
imin=0,
irest=1,
ntx=5,
ntb=2,
ntp=1,
cut=12,
ntc=2,
ntf=2,
tempi=300.0,
temp0=300.0,
ntt=3,
gamma_ln=1.0,
nstlim=50000,
dt=0.002,
ntwx=5000,
ntwr=5000
ntpr=5000,
ntr=0,
ig=-1,
/
```

Fig. 9 Example of AMBER pmemd input file for the fifth step equilibration. In the input file, the flag *ntr = 0* indicates that the positional restraint method is turned off. The meaning of the rest of the parameters listed in the input file was previously explained

prmtop -r MEK1-BRAF_eq4.rst -ref MEK1-BRAF_eq3.rst -x MEK1-BRAF_eq4.mdcrd

- Run the last step of the equilibration consisting on 100-ps unrestrained MD simulation in isothermal-isobaric condition. Run *equil5.in* input script (shown in Fig. 9, *see* **Note 5**) using the following command:

  $AMBERHOME/bin/pmemd -i equil5.in -o equil5.out -c MEK1-BRAF_eq4.rst -p MEK1-BRAF_solv.prmtop -r MEK1-BRAF_eq5.rst -ref MEK1-BRAF_eq4.rst -x MEK1-BRAF_eq5.mdcrd

(f) Finally, perform 5-ns MD unrestrained simulation keeping the same system condition as the last equilibration step. Run *prod.in* input script (shown in Fig. 10, *see* **Note 6**) using the following command:

$AMBERHOME/bin/pmemd -i prod.in -o prod.out -c MEK1-BRAF_eq5.rst -p MEK1-BRAF_solv.prmtop -r MEK1-BRAF_prod.rst -ref MEK1-BRAF_eq5.rst -x MEK1-BRAF_prod.mdcrd

3. Binding free energy calculation.

(a) Build the topology and coordinate files of the unsolvated wild type (WT) structure for both the complex and its single subunits using *tleap-WT.in* input file (shown in

```
#5ns-MD simulation

&cntrl
imin=0,
irest=1,
ntx=5,
ntb=2,
ntp=1,
cut=12,
ntc=2,
ntf=2,
tempi=300.0,
temp0=300.0,
ntt=3,
gamma_ln=1.0,
nstlim=2500000,
dt=0.002,
ntwx=5000,
ntwr=5000
ntpr=5000,
ntr=0,
ig=-1,
/
```

Fig. 10 Example of AMBER pmemd input file for unrestrained MD. The meaning of all the parameters listed in the input file was previously explained

```
source leaprc.ff99SB
source leaprc.gaff

#Load ATP parameters
loadamberprep ATP.prep
loadamberparams ATP.frcmod

#Load pdb files
4mne=loadpdb MEK1-BRAF.pdb
mek1=loadpdb MEK1.pdb
braf=loadpdb BRAF.pdb

#Save topology and coordinate files
saveamberparm 4mne MEK1-BRAF.prmtop MEK1-BRAF.inpcrd
saveamberparm mek1 MEK1.prmtop MEK1.inpcrd
saveamberparm braf BRAF.prmtop BRAF.inpcrd

quit
```

Fig. 11 Example of AMBER LEaP input file to build topology and coordinates files of wild type dry systems

Fig. 11). Run LEaP AMBER tool using the following command:

$AMBERHOME/bin/tleap -f tleap-WT.in > tleap-WT.out

(b) For each mutation studied, build the topology and coordinate files of the mutated structure for both the complex and mutated subunit using *tleap-mut.in* input file (shown

```
source leaprc.ff99S
source leaprc.gaff

#Load ATP parameters
loadamberprep ATP.prep
loadamberparams ATP.frcmod

#Load pdb files
4mne=loadpdb MEK1-BRAF_F468A.pdb
braf=loadpdb BRAF_F468A.pdb

#Save topology and coordinate files
saveamberparm 4mne MEK1-BRAF_F468A.prmtop MEK1-BRAF_F468A.inpcrd
saveamberparm braf BRAF_F468A.prmtop BRAF_F468A.inpcrd

quit
```

Fig. 12 Example of AMBER LEaP input file to build topology and coordinates files of mutated dry systems. Here, F468 BRAF residue is taken as example

```
#Alanine scanning

&general
receptor_mask=":1-346,623,624"
startframe=3000, endframe=5000, interval=10,
verbose=1,
/

&gb
saltcon=0.1
/

&pb
istrng=0.100
/

&alanine_scanning
/
```

Fig. 13 Example of MMPBSA.py input file to perform alanine scanning calculation. The input file is typically divided into four sections (&general, &gb, &pb, &alanine_scanning). The &*general* section is designed to specify generic variables related to the overall calculation. For instance, the flag *startframe* and *endframe* specifies the frame from which to begin and to stop extracting snapshots, respectively, the parameter *interval* indicates the offset from which to choose frames from the trajectory file, *verbose* = *1* means that complex, ligand, and receptor energy terms will be printed in the output file. The &*gb* and &*pb* section markers tells the script to perform MM-GBSA and MM-PBSA calculations with the given values defined within those sections (i.e., the variables *saltcon* and *istrng* that specify the salt concentration and the ionic strength, respectively). Finally the &*alanine_scanning* section marker initializes alanine scanning in the script. Please be aware that given the higher computational costs of MM-PBSA calculation, only MM-GBSA calculation is performed in this work

in Fig. 12). Run LEaP AMBER tool using the following command:

$AMBERHOME/bin/tleap -f tleap-mut.in > tleap-mut.out

(c) Perform alanine scanning calculation on 200 snapshots extracted from the last 2 ns of each MD trajectory. Run *mmpbsa.in* input file for *MMPBSA.py* script in AMBER14 (shown in Fig. 13) using the following command:

$AMBERHOME/bin/MMPBSA.py -i mmpbsa.in -sp MEK1-BRAF_solv.prmtop -cp MEK1-BRAF.prmtop -rp MEK1-BRAF.prmtop -lp MEK1-BRAF.prmtop -y MEK1-BRAF_prod.mdcrd -mc MEK1-BRAF_F468A.prmtop -ml BRAF_F468A.prmtop

Flag -i specifies the input file, *-sp* the solvated WT complex topology file, *-cp* the unsolvated WT complex topology file, *-rp* the unsolvated WT receptor topology file, *-lp* the unsolvated WT ligand topology file, -y the complex trajectory file to analyze, *-mc* the unsolvent mutated complex topology file and *-ml* the unsolvated mutated subunit topology file. Please be aware that as MEK1 is the first molecule in the complex, for alanine scanning calculations the unsolvated mutated subunit topology file will be specified with the flag *-mr*.

(d) Extract the $\Delta\Delta G$ of binding related to the specific mutations estimated as the difference between the binding ΔG of the WT and that of the mutated complex. All these data are easily available in the final output file, *FINAL_RESULTS_MMPBSA.dat,* including all the wild type and mutated system average binding energies (reported as van der Waals, electrostatic, and nonpolar energy contributions), as shown in Fig. 14.

(e) You may get a graphical representation of the $\Delta\Delta G$ of binding (*see* Fig. 1), by assigning the values given in *FINAL_RESULTS_MMPBSA.dat* file to the bfactor field of the corresponding PDB file of the complex structure.

4 Notes

1. As there is no unbound structure for the ligand yet, the ligand structure contained on the complex PDB file (4MNE) is used here instead for illustration purposes. However, in a standard NIP computation, unbound structures should be used.
2. The principal columns of the *4mne.ene* file are:
   - Conf: Conformation number of the docking pose as in the last column of the rot file.
   - Ele: Electrostatic energy of the pose.
   - Desolv: Desolvation energy of the pose.

```
|Calculations performed using 201 complex frames.
|
|All units are reported in kcal/mole.
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------

GENERALIZED BORN:

Differences (Complex - Receptor - Ligand):
Energy Component            Average              Std. Dev.   Std. Err. of Mean
-------------------------------------------------------------------------------
VDWAALS                   -161.0164               8.5993              0.6065
EEL                      -1068.5067              36.1059              2.5467
EGB                       1172.6667              35.5088              2.5046
ESURF                      -23.1830               0.9495              0.0670

DELTA G gas              -1229.5231              36.2700              2.5583
DELTA G solv              1149.4837              35.4458              2.5002

DELTA TOTAL                -80.0394              11.0084              0.7765


-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
F367A MUTANT:
GENERALIZED BORN:

Differences (Complex - Receptor - Ligand):
Energy Component            Average              Std. Dev.   Std. Err. of Mean
-------------------------------------------------------------------------------
VDWAALS                   -158.7570               8.4809              0.5982
EEL                      -1068.8691              36.0357              2.5418
EGB                       1172.3985              35.5099              2.5047
ESURF                      -22.7274               0.9551              0.0674

DELTA G gas              -1227.6261              36.3593              2.5646
DELTA G solv              1149.6712              35.4335              2.4993

DELTA TOTAL                -77.9549              11.0214              0.7774


-------------------------------------------------------------------------------
-------------------------------------------------------------------------------

RESULT OF ALANINE SCANNING: (F468A) DELTA DELTA G binding =     -2.0844+/-0.5545
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
```

Fig. 14 Extract from the MMPBSA.py *FINAL_RESULTS_MMPBSA.dat* output file. The file includes all the average energies, standard deviations, and standard error of the mean for GB followed by PB calculations (if calculated). After each section, the ΔG of binding is given along with the error values. After each method, the $\Delta\Delta G$ of binding is reported, corresponding to the relative effect the mutation has on the ΔG of binding for the complex. The specific mutation is also printed at the end of the file. Here, F468 residue alanine scanning is taken as example

- VDW: van der Waals energy of the pose.
- Total: Total docking energy of the pose, computed as ele + Desolv + 0.1 * VDW (note a 0.1 weight for VDW).
- RANK: Pose rank according to its computed total binding energy.

3. Files from the PDB may contain bound ligands, cofactors or nonstandard residues whose parameters are not available in the AMBER parameters database. In this case you should make use of the Antechamber tools, which ship with AmberTools, to create PREP and FRCMOD files. For more information, see the ANTECHAMBER tutorial (http://ambermd.org/tutorials/basic/tutorial4b/) and the AMBER manual.
4. LEaP AMBER tool renumbers PDB residues starting from 1. Thus, the original numeration of your PDB file may not be always kept.
5. Since your system may not start from an equilibrium state, additional time steps may be required during the minimization and equilibration steps of the MD simulation. One can check for equilibrium by verifying whether properties, such as potential energy, temperature, or pressure, no longer change in any systematic fashion and are just fluctuating around a mean value.
6. To guarantee reliable results in the in silico Alanine scanning calculation, RMSD simulation should be highly equilibrated. Ideally one should probably run a much longer production run than 5 ns (i.e., 20 ns).

References

1. Arkin MR, Wells JA (2004) Small-molecule inhibitors of protein-protein interactions: progressing towards the dream. Nat Rev Drug Discov 3:301–317
2. DeLano WL (2002) Unraveling hot spots in binding interfaces: progress and challenges. Curr Opin Struct Biol 12:14–20
3. Toogood PL (2002) Inhibition of protein-protein association by small molecules: approaches and progress. J Med Chem 45:1543–1558
4. Glaser F, Pupko T, Paz I et al (2003) ConSurf: identification of functional regions in proteins by surface-mapping of phylogenetic information. Bioinformatics 19:163–164
5. Ashkenazy H, Erez E, Martz E et al (2010) ConSurf 2010: calculating evolutionary conservation in sequence and structure of proteins and nucleic acids. Nucleic Acids Res 38:W529–533
6. Landau M, Mayrose I, Rosenberg Y et al (2005) ConSurf 2005: the projection of evolutionary conservation scores of residues on protein structures. Nucleic Acids Res 33:W299–302
7. Celniker G, Nimrod G, Ashkenazy H et al (2013) ConSurf: using evolutionary data to raise testable hypotheses about protein function. Isr J Chem 53:199–206
8. Grosdidier S, Fernandez-Recio J (2008) Identification of hot-spot residues in protein-protein interactions by computational docking. BMC Bioinformatics 9:447
9. Branden CI, Tooze J (1999) Introduction to protein structure, 2nd edn. Garland Pub, New York, NY
10. Cheng TM, Blundell TL, Fernandez-Recio J (2007) pyDock: electrostatics and desolvation for effective scoring of rigid-body protein-protein docking. Proteins 68:503–515
11. Case DA, Cheatham TE, Darden T et al (2005) The Amber biomolecular simulation programs. J Comput Chem 26:1668–1688

12. Miller BRI, McGee DTJ, Swails JM et al (2012) MMPBSA.py: an efficient program for end-state free energy calculations. J Chem Theor Comput 8:3314–3321
13. Haling JR, Sudhamsu J, Yen I et al (2014) Structure of the BRAF-MEK complex reveals a kinase activity independent role for BRAF in MAPK signaling. Cancer Cell 26:402–413
14. Kiel C, Serrano L (2014) Structure-energy-based predictions and network modelling of RASopathy and cancer missense mutations. Mol Syst Biol 10:727
15. Jimenez-Garcia B, Pons C, Fernandez-Recio J (2013) pyDockWEB: a web server for rigid-body protein-protein docking using electrostatics and desolvation scoring. Bioinformatics 29:1698–1699
16. Gabb HA, Jackson RM, Sternberg MJ (1997) Modelling protein docking using shape complementarity, electrostatics and biochemical information. J Mol Biol 272:106–120
17. Pettersen EF, Goddard TD, Huang CC et al (2004) UCSF Chimera—a visualization system for exploratory research and analysis. J Comput Chem 25:1605–1612

Chapter 7

Multistate Computational Protein Design with Backbone Ensembles

James A. Davey and Roberto A. Chica

Abstract

The ability of computational protein design (CPD) to identify protein sequences possessing desired characteristics in vast sequence spaces makes it a highly valuable tool in the protein engineering toolbox. CPD calculations are typically performed using a single-state design (SSD) approach in which amino-acid sequences are optimized on a single protein structure. Although SSD has been successfully applied to the design of numerous protein functions and folds, the approach can lead to the incorrect rejection of desirable sequences because of the combined use of a fixed protein backbone template and a set of rigid rotamers. This fixed backbone approximation can be addressed by using multistate design (MSD) with backbone ensembles. MSD improves the quality of predicted sequences by using ensembles approximating conformational flexibility as input templates instead of a single fixed protein structure. In this chapter, we present a step-by-step guide to the implementation and analysis of MSD calculations with backbone ensembles. Specifically, we describe ensemble generation with the PertMin protocol, execution of MSD calculations for recapitulation of *Streptococcal* protein G domain β1 mutant stability, and analysis of computational predictions by sequence binning. Furthermore, we provide a comparison between MSD and SSD calculation results and discuss the benefits of multistate approaches to CPD.

Key words Single-state design, Multistate analysis, Multistate design, PertMin, Protein stability prediction, Receiver operating characteristic, Protein G

1 Introduction

The continued development of computational protein design (CPD) methodologies has led to an increasing number of designed proteins possessing unique structural [1–3] and functional [4–7] characteristics. CPD is a powerful tool for protein engineering because it enables the identification of sequences displaying desired properties in spaces astronomically larger ($>10^{80}$ sequences) [8] than those that can be tested experimentally. CPD simulations are typically performed using a single-state design (SSD) approach in which amino-acid sequences are optimized on a single protein structure. Most SSD procedures consist of three steps: (1) a side-chain placement step where discrete side-chain rotamers are

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_7,

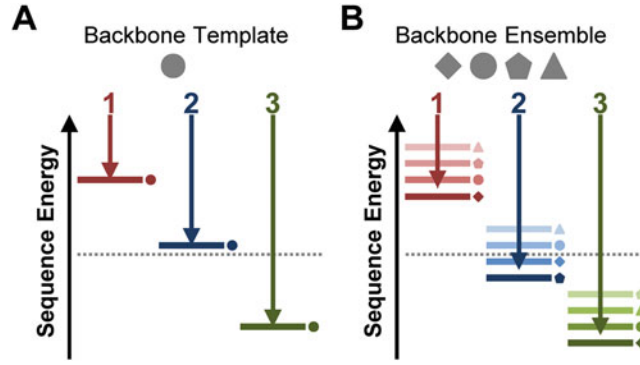


Fig. 1 Single-state and multistate design. In single-state design (**a**), a single backbone template (*circle*) is used to score and rank sequences (1, 2, and 3) according to their predicted stability (*arrow*). Application of an arbitrary energy cutoff (*dotted line*) results in acceptance of sequence 3 as stable and rejection of sequences 1 and 2 as unstable. In multistate design (**b**), an ensemble of four backbone structures (*diamond*, *circle*, *pentagon*, and *triangle*) is used to score and rank sequences. Predicted stability (*arrow*) is computed as the Boltzmann weighted average energy across all members of the ensemble for each sequence. Application of the same energy cutoff as in single-state design (*dotted line*) results in sequences 2 and 3 being accepted and sequence 1 being rejected

threaded onto specified positions on a fixed protein backbone template, (2) an energy calculation where interaction energies between pairs of rotamers and between each rotamer and the backbone template are computed using a potential energy function, and (3) a sequence optimization step where combinations of rotamers are optimized using a search algorithm that explores both rotamer and sequence space to identify optimal sequences. At the conclusion of this process, a list of sequences is generated (Fig. 1a) with each sequence being ranked according to a score value that reflects its stability on the target protein structure.

Although SSD has been successfully applied to the design of numerous protein functions and folds, the approach is susceptible to false negative predictions that result from the combined use of a fixed protein backbone template and a set of rigid rotamers to model mutant protein structures. This fixed backbone approximation leads to the incorrect rejection of desirable sequences that would be accepted if the backbone geometry was allowed to relax or if a slightly different rotamer configuration was allowed [9]. To address the fixed backbone approximation, several strategies have been developed including the use of softer repulsive potential energy terms [10–12], flexible backbone algorithms [13–15], iterative energy minimization [16, 17], and continuous rotamer optimization [18]. Recently, multistate design (MSD) with backbone ensembles approximating protein conformational flexibility

has emerged has a useful alternative to these methods [19, 20]. In MSD, sequence optimization is guided by energy contributions of multiple protein structures simultaneously, enabling the evaluation of sequences in the context of an ensemble of backbone templates. MSD simulations consist of multiple independent single-state calculations in which rotamers for a specific amino-acid sequence are optimized in the context of each backbone template included in the ensemble. Individual SSD scores obtained on each template are then combined into a single fitness value for each amino-acid sequence that represents its predicted stability across the ensemble. MSD optimization algorithms [21–23] attempt to improve this fitness value as a function of amino-acid sequence to identify optimal sequences. Thus, MSD differs from SSD by its use of an energy combination function to compute sequence fitness and a modified search algorithm to find optimal sequences in the context of multiple backbone templates. Because multiple backbones are used to inform sequence selection in MSD, combinations of rotamers that would be rejected in SSD because they cause steric clashes in a single fixed backbone template can be accepted if they have a stabilizing effect in at least one of the backbone templates included in the ensemble (Fig. 1b). In this way, MSD with backbone ensembles leads to fewer false negatives and improved overall prediction accuracy [20].

An alternate approach to MSD that can be used to evaluate fitness of amino-acid sequences across multiple backbone templates is multistate analysis (MSA). MSA involves the combination of scores obtained from parallel SSD simulations into a single fitness value for each sequence that is computed post-CPD. The resulting fitness values are then used to re-rank sequences (Fig. 2). By employing alternate backbones as input templates to these parallel SSD calculations, MSA can be used to identify the most favorable template to score each sequence [24] or to evaluate how well each sequence stabilizes an ensemble of backbone templates. MSA differs from MSD by its sequence optimization procedure, which is not informed by the energetic contributions of multiple backbones. Instead, sequence optimization in MSA is performed as in SSD and only the use of an energy combination function to compute sequence fitness distinguishes it from SSD. Because of this, MSA has the benefit of being less computationally demanding than MSD but has the drawback of potentially constraining explored sequence space since sequence optimization is not guided by multiple states.

In order to implement MSD or MSA, multiple backbone templates are required. These templates can be obtained from available x-ray or NMR structures or can be generated *in silico* from the atomic coordinates of a single protein. Several computational methods have been developed to generate backbone ensembles for use in MSD [19, 25]. In this chapter, we will focus on the coordinate perturbation followed by energy minimization

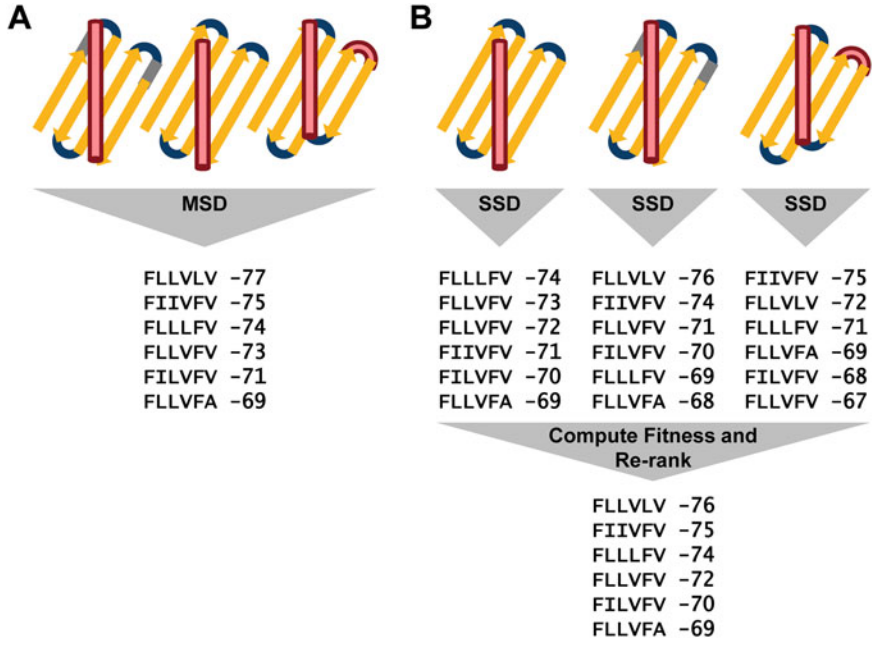


Fig. 2 Multistate approaches to computational protein design. In MSD (**a**), sequence optimization is guided by the energetic contributions of multiple protein structures simultaneously. Thus, sequence optimization and scoring are performed concertedly, resulting in a list of sequences that are ranked according to their predicted stability across the ensemble of three structures. In MSA (**b**), multiple independent SSD calculations are performed in parallel using alternate backbones as input templates. The sequence scores obtained from each SSD calculation are combined post-CPD into a fitness value for each sequence across the ensemble of three structures. Sequences are then re-ranked based on their fitness values, generating a new ranked list of scored sequences

(PertMin) protocol that we recently developed [20]. In this procedure, small coordinate perturbations are introduced into a starting protein structure to generate a set of randomly perturbed structures. An energy minimization procedure is then applied to the perturbed structures, which minimize to different local minima that become accessible because of diverging trajectories (Fig. 3a). PertMin thus exploits the initial condition sensitivity of energy minimization [26]. A benefit of the PertMin protocol is that structural deviation from the input structure and ensemble diversity (i.e., structural deviation between ensemble members) can be controlled by the number of minimization steps (Fig. 3b). While PertMin does not allow for a large area of protein conformational space to be explored, it enables the rapid and tunable generation of ensemble backbones having high coordinate similarity to their progenitor structure and low potential energy. Thus, application of PertMin ensembles in MSD results in improved prediction accuracy compared to SSD by reducing the number of false negatives and increasing the number of true positives [20].

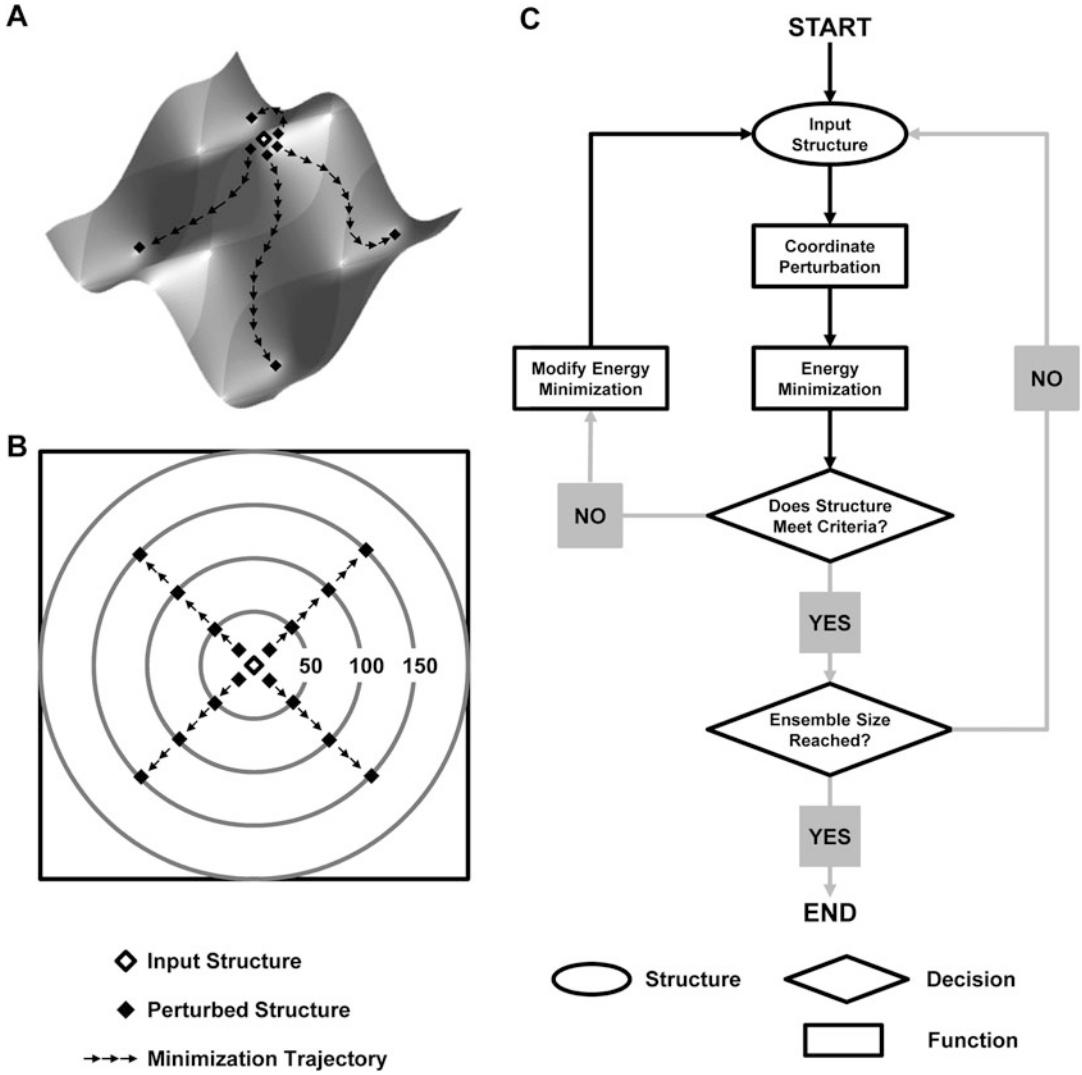


Fig. 3 The PertMin protocol. (**a**) PertMin functions by introducing small random coordinate perturbations into an input protein structure (*white diamond*) to yield a set of perturbed structures (*black diamonds*). This perturbation step is followed by energy minimization (*arrows*) of each perturbed structure into different local minima surrounding the input structure. (**b**) In PertMin, a higher number of energy minimization steps (represented by *circles*) leads to increased root-mean-square coordinate deviation from the input structure and greater ensemble diversity (represented by the *circle arc*). (**c**) The PertMin algorithm

In this chapter, we provide a step-by-step guide to the implementation and analysis of MSD calculations with backbone ensembles. To facilitate this explanation, we present an example involving the recapitulation of a training set of 84 *Streptococcal* protein G domain β1 (Gβ1) mutant sequences of known stability (Table 1) [19]. A specific focus is placed on the generation of backbone ensembles using the PertMin protocol and on their application in both MSD and MSA. An analysis of CPD predictions and a comparison between SSD, MSA, and MSD calculation results are presented.

Table 1
Gβ1 training set sequences

Stabilizing[a]	Destabilizing[b]	Unfolded[c]	Nonnative[d]
FLIAAFAIWFV[e]	FIIAAFAIWFV	FIVAAFAVWFI	FAFAAFFIWFA
FLIAAFALWFI	FIIAAFAIWFI	FLIAAFAVWLV	FALAAFFIWFA
FLFAAFALWFI	FIVAAFAIWFV	FLVAAFAVWIV	FAFAAIFIWFA
FLVAAFAIWFV	FILAAFAIWFV	FLLAAFAVWLV	FAFAALFIWFA
FLIAAFAVWFV	FIVAAFAIWFI	FIIAAFAVWFV	FALAAIFIWFA
FLFAAFAIWFV	FILAAFAVWFV	FLIAAFAIWIV	FALAALFIWFA
FLFAAFAIWFI	FIIAAFAVWFI	FLIAAFAIWLV	FAFAAFFLWFA
FLIAAFAIWFI	FIVAAFAVWFV	FLIAAFAIWVV	FALAAFFLWFA
FLIAAFALWFV	FLIAAFAVWIV	FLIAAFALWIV	FAFAAIFLWFA
FLVAAFAIWFI	FILAAFAIWFI	FLIAAFALWLV	FAFAALFLWFA
FLVAAFALWFI	FILAAFAVWFI	FLIAAFALWVV	FALAAIFLWFA
FLLAAFAIWFV	FLLAAFAVWVV	FLIAAFAVWVV	FALAALFLWFA
FLFAAFALWFV		FLLAAFAIWIV	FAFAAFFIWFV
FLIAAFAVWFI		FLLAAFAIWLV	FALAAFFIWFV
FLLAAFAVWFV		FLLAAFAIWVV	FAFAAIFIWFV
FLVAAFAVWFV		FLLAAFALWIV	FAFAALFIWFV
FLVAAFALWFV		FLLAAFALWLV	FALAAIFIWFV
FLLAAFALWFI		FLLAAFALWVV	FALAALFIWFV
FLVAAFAVWFI		FLLAAFAVWIV	FAFAAFFLWFV
FLLAAFAIWFI		FLVAAFAIWIV	FALAAFFLWFV
FLLAAFALWFV		FLVAAFAIWLV	FAFAAIFLWFV
FLFAAFAVWFV		FLVAAFAIWVV	FAFAALFLWFV
FLFAAFAVWFI		FLVAAFAVWLV	FALAAIFLWFV
FLLAAFAVWFI		FLVAAFAVWVV	FALAALFLWFV

All sequences are from [19]
[a]Stabilizing sequences consist of 24 Gβ1 mutants whose stability is approximately equal to or greater than that of the wild type (WT)
[b]Destabilizing sequences consist of 12 Gβ1 mutants whose stability is lower than that of the WT
[c]Unfolded sequences consist of 24 Gβ1 mutants that do not fold
[d]Nonnative sequences consist of 24 Gβ1 mutants postulated to adopt an alternate protein fold
[e]Amino-acid sequences of Gβ1 mutants show residue identity at core positions in this order: positions 3, 5, 7, 20, 26, 30, 34, 39, 43, 52, and 54

2 Materials

Ensemble generation and data analysis were performed by single-threaded calculations on an AMD Athlon II personal computer. All CPD simulations were conducted on a Linux cluster consisting of Intel Xeon and AMD Opteron x86-64 CPUs. The PertMin protocol was implemented using the Molecular Operating Environment (MOE) software package [27] and CPD calculations were run using PHOENIX [4, 19, 28]. Parsing and analysis of CPD results were done using Python 2.7 and Microsoft Excel 2007. Input coordinates for the Gβ1 fold were retrieved from the Protein Data Bank (PDB ID: 1PGA) [29]. Gβ1 training set sequences and their stabilities were retrieved from [19].

3 Methods

3.1 Structure Preparation for Single-State Design

1. Retrieve the Gβ1 crystal structure from the Protein Data Bank (PDB ID: 1PGA) [29].
2. Remove water molecules included in the crystal structure.
3. Prepare the Gβ1 structure for calculation by adding hydrogens, counter-ions, and solvent using MOE [27] (*see* **Note 1**).
4. Energy minimize the prepared Gβ1 structure with 50 steps of conjugate gradient energy minimization [30].

The resulting energy minimized structure will be used as the input backbone template for SSD.

3.2 Ensemble Preparation

Preparation of a backbone ensemble to be used as input to MSA and MSD calculations is carried out with the PertMin protocol [20] (Fig. 3c).

1. Perturb all atoms of the unminimized Gβ1 structure prepared with added hydrogens, counter-ions, and solvent by introducing random Cartesian coordinate perturbations of $\pm$ 0.001 Å along each axis.
2. Energy minimize the perturbed Gβ1 structure with 50 iterations of truncated Newton energy minimization [31].
3. Evaluate the resulting minimized structure to ensure that it meets user-specified criteria. In this case, a protein backbone atom coordinate root-mean-square (RMS) deviation from the 1PGA crystal structure of 0.3 Å or more is required. If the structure meets this requirement, add it to the PertMin ensemble. If not, discard it and modify the energy minimization procedure accordingly (*see* **Note 2**).
4. Repeat **steps 1** through **3** until the PertMin ensemble contains 64 structures.

The 64-member PertMin ensemble prepared as described above has an average backbone atom RMS coordinate deviation from the 1PGA crystal structure of 0.46 ± 0.02 Å and an ensemble diversity (backbone atoms) of 0.25 ± 0.04 Å.

3.3 Computational Protein Design Calculations

SSD, MSA, and MSD calculations to recapitulate the known stability of the Gβ1 training set sequences (Table 1) were conducted using the PHOENIX protein design software [4, 19, 28].

1. Design Gβ1 core residues (positions 3, 5, 7, 30, 34, 39, 52, and 54) with amino-acid types found at these positions in the training set of 84 mutant Gβ1 sequences (Fig. 4). For Gβ1 core residues A20, A26, and W43, allow conformation to vary but not amino-acid identity.
2. Thread amino-acid side-chain rotamers onto backbone template(s) at these positions using the Dunbrack backbone-dependent rotamer library with expansions of ± 1 standard deviation around χ_1 and χ_2 [32]. The crystallographic conformer found at these positions is also included.
3. Evaluate interaction energies between pairs of rotamers and between each rotamer with the backbone template using a physics-based four-term potential energy function that includes (a) a van der Waals term from the Dreiding II force field with atomic radii scaled by 0.9 [33], (b) a direction-specific hydrogen-bond term having a well depth of 8.0 kcal/mol [11], (c) an electrostatic energy term modeled using Coulomb's law with a distance-dependent dielectric of 40, and (d) a surface area-based solvation penalty term [34, 35].
4. Apply a 1000 kcal/mol potential energy penalty against the crystallographic conformer found at each designed position. Application of this penalty ensures adequate sampling away from the wild-type sequence.

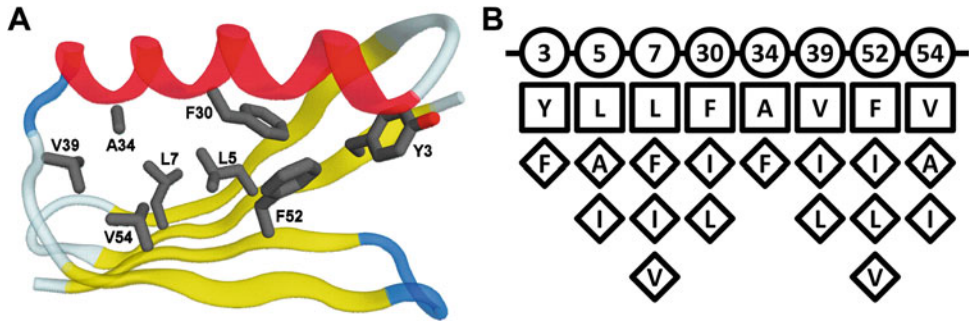


Fig. 4 Computational design of Gβ1 core residues. (**a**) Designed Gβ1 core residues. (**b**) Wild-type (*square*) and mutant (*diamond*) residues included in calculations are shown for each designed position (*circle*). The total searched sequence space during calculation consists of 5184 possible sequences

5. For SSD and MSA, optimize sequences on a single backbone template at a time using the FASTER (Fast and accurate side-chain topology and energy refinement) algorithm [36, 37]. For MSD, optimize sequences in the context of the PertMin ensemble using a modified version of the FASTER algorithm, MSD-FASTER [21].
6. After SSD calculations are completed, compute fitness values for MSA as the Boltzmann weighted average of individual sequence energies obtained on all backbones included in the PertMin ensemble (*see* **Note 3**). For MSD, the Boltzmann weighted average fitness is computed during sequence optimization.

Following CPD, a rank ordered list of scored sequences is obtained (Fig. 5). In SSD, sequences are ranked according to their score, which is the potential energy on a single backbone template. In MSA and MSD, sequences are ranked according to their fitness value, which is the Boltzmann weighted average energy across all backbone templates included in the PertMin ensemble. Because fitness is evaluated concertedly to sequence optimization in MSD but not in MSA, where fitness is instead computed post-CPD (Fig. 2), sequence ranking and fitness values obtained by these multistate approaches are not identical (Fig. 5).

SSD

	WT	YLLAAFAVWFV	
0	F-I--------	-83.306
1	F-I----I---	-82.785
2	F-V--------	-79.413
3	FII--------	-78.823
4	F-V----I---	-78.696
5	FII----I---	-78.343
6	F-I-------I	-76.970
7	F----------	-76.719
8	--I--------	-76.485
9	F-I----I--I	-76.284
10	F------I---	-76.009
11	FI---------	-75.773
12	--I----I---	-75.375
13	FIV--------	-75.353
14	F-I----L---	-73.977
15	FI-----I---	-73.776
16	F-I-------A	-73.697
17	FIV----I---	-73.403
18	F-I------L-	-73.167
19	F-V----L---	-72.747
20	F-I----I-L-	-72.522
21	F------L---	-72.397
22	FII-------I	-72.259
23	-II--------	-71.972
24	FAI-------I	-71.927

MSA

	WT	YLLAAFAVWFV	
0	F-I----I---	-86.137
1	F-I----L---	-84.304
2	F-I--------	-83.405
3	F------I---	-83.353
4	F-V----I---	-82.544
5	F-I----L--I	-81.060
6	F-V----L---	-80.746
7	F----------	-80.691
8	F-I----I--I	-80.656
9	F------L---	-80.273
10	F-I-------I	-80.243
11	FAI---FI---	-79.915
12	F-V--------	-79.879
13	--I----I---	-79.297
14	F-V----L--I	-78.138
15	FAI---F----	-77.940
16	--I----L---	-77.877
17	F---------I	-77.268
18	F-I--L-I---	-77.095
19	FAI---FI--I	-76.988
20	--I--------	-76.648
21	F-V-------I	-76.491
22	F-I----I-L-	-76.333
23	FAI--------	-76.232
24	F-I----I--A	-76.188

MSD

	WT	YLLAAFAVWFV	
0	F-I----I---	-86.137
1	F-I----L---	-84.325
2	F-I--------	-83.439
3	F------I---	-83.316
4	F-V----I---	-82.630
5	F-I----L--I	-81.434
6	F----------	-81.359
7	F-I----I--I	-81.033
8	F-V----L---	-80.995
9	F------L---	-80.195
10	F-I-------I	-80.089
11	F-V--------	-80.053
12	FAI---FI---	-79.911
13	--I----I---	-79.746
14	F---------I	-79.148
15	FAI---F----	-78.293
16	F-V----L--I	-78.026
17	--I----L---	-77.925
18	F------I--I	-77.813
19	FAI---F---I	-77.629
20	FAV---FI---	-77.506
21	F-I--L-I---	-77.494
22	F-V----I--I	-77.455
23	FAI--------	-77.351
24	FAI---FI--I	-77.006

Fig. 5 Ranked lists of scored sequences obtained by various computational protein design methods. The wild-type (WT) and top 25 mutant sequences predicted by SSD, MSA, and MSD are shown. *Numbers* represent single-state score or multistate fitness values for each sequence

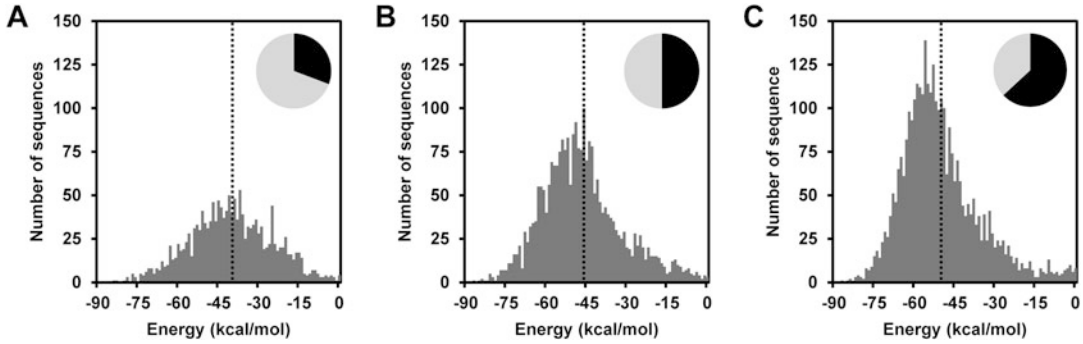


Fig. 6 Sequence energy distributions. Distributions depict the number of sequences predicted by SSD (**a**), MSA (**b**), and MSD (**c**) with energy values grouped in incremental bins of 1 kcal/mol. The average sequence energy is indicated by a *dotted black line*. The fraction of the pie charts in black (31, 50, and 63 % for SSD, MSA, and MSD, respectively) corresponds to the percentage of the 5184 total possible sequences with predicted energy below 0 kcal/mol

Sequence energy distributions (Fig. 6) show that more favorable energies are obtained for a larger number of sequences by the multistate approaches than by SSD, with average sequence energies of −40, −46, and −50 kcal/mol obtained by SSD, MSA, and MSD, respectively. The lower energies obtained by multistate approaches result from their ability to identify better backbone templates to score each sequence than the single template used in SSD [20, 24]. This is exemplified by the greater number of sequences that are scored with an energy lower than 0 kcal/mol by the multistate approaches (50 and 63 % for MSA and MSD, respectively) than by SSD (31 %), highlighting how multistate approaches help to address the fixed backbone approximation.

3.4 Energy Analysis of Predicted Sequences

In this section, the top 100 sequences predicted by the various CPD methods will be analyzed with the sequence binning procedure [20]. In this procedure, sequences from each rank-ordered list are binned as either stable or unstable by comparing their energy value relative to that of the wild-type (WT) sequence. The energy of the WT is used as the cutoff because the WT is known to be stable in the context of the Gβ1 fold and because all CPD methods used here are expected to rank the WT sequence favorably. Stabilizing sequences are thus expected to be ranked ahead of the WT while destabilizing, unfolded, and nonnative sequences (Table 1) are expected to be ranked below the WT.

1. Compute the energy difference (ΔE) between each sequence included in the top 100 and the WT. WT energy values obtained by SSD, MSA, and MSD are −69.8, −73.6, and −74.5 kcal/mol, respectively (Table 2).
2. Bin sequences as potential positives or negatives if their ΔE value is lower or greater than 0 kcal/mol, respectively.

Table 2
Sequence binning results

Binning statistics	SSD	MSA	MSD
Success rate (%)	70	85	88
Cutoff (kcal/mol)	−69.8	−73.6	−74.5
True positives	12	17	18
False negatives	12	7	6
False positives	13	6	4
True negatives	47	54	56

3. From the set of potential positive sequences, identify true and false positive sequences. True positives are the 24 stabilizing sequences of the training set while false positives include the 12 destabilizing, 24 unfolded, and 24 nonnative sequences (Table 1). Because the experimental stability of the remaining 5099 designed sequences is unknown, they are not considered in our binning procedure.
4. From the set of potential negative sequences, identify true and false negative sequences. True negatives include the destabilizing, unfolded, and nonnative sequences of the training set while false positives are the stabilizing sequences (Table 1).
5. Compute the success rate of the binning procedure, which is the percentage of correctly binned sequences (true positives and true negatives) out of the complete training set. For this sequence binning analysis, the WT sequence is not included in the binning statistic.
6. Perform **steps 1** through **5** using a series of cutoff values ranging from −90 to +90 kcal/mol in 1 kcal/mol increments. Build receiver operating characteristic (ROC) curves by plotting the true positive ratio (fraction of true positives out of the positives) as a function of the false positive ratio (fraction of false positives out of the negatives) for every possible cutoff value.

Sequence binning results shown in Fig. 7 demonstrate that MSD is the only method that can score all 84 training set sequences below 0 kcal/mol. In contrast, SSD and MSA score fewer of the training set sequences below 0 kcal/mol (68 and 82 %, respectively). The multistate methods correctly reject a higher number of the 60 true negatives (Table 1, destabilizing, unfolded, and nonnative sequences) and correctly accept a higher number of the true positives (Table 1, stabilizing sequences), compared to SSD (Table 2). As a result, fewer false negatives are predicted by MSA

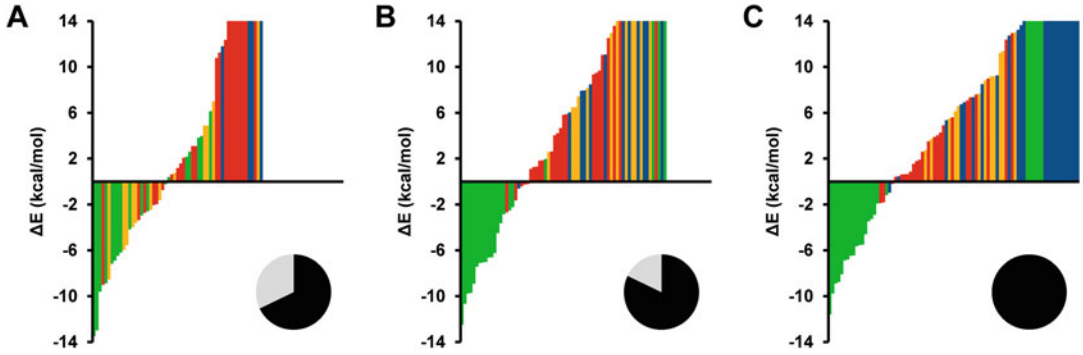


Fig. 7 Sequence binning analysis. Gβ1 training set sequences predicted by SSD (**a**), MSA (**b**), and MSD (**c**) calculations are binned according to their energy difference from the wild-type (WT) sequence. Sequences with lower energy than the WT ($\Delta E < 0$ kcal/mol) are potential positive sequences while sequences with higher energy than the WT ($\Delta E > 0$ kcal/mol) are potential negative sequences. Sequences are colored according to their experimental stability, with sequences having stability greater than or approximately equal to the WT in *green* (stabilizing), sequences having lower stability than the WT in *yellow* (destabilizing), sequences that do not fold in red (unfolded), and sequences postulated to adopt a nonnative fold in blue (nonnative). Positive ΔE values are capped at +14 kcal/mol, even if the predicted energy difference is greater than this value. The fraction of the pie charts in black (68, 82, and 100 % for SSD, MSA, and MSD, respectively) corresponds to the percentage of the 84 training set sequences with predicted energy below 0 kcal/mol

and MSD than by SSD, an expected result given that they help to address the fixed backbone approximation. Additionally, the success rates of multistate methods are greater than that of SSD, with the MSD success rate being the highest (88 %). The number of false negative predictions made by MSD increases if the ensemble does not cover a sufficient structure space to score training set sequences, resulting in decreased average binning success rates (*see* **Note 4**).

Although the binning profiles described above demonstrate the utility of using the WT sequence energy as the cutoff, ROC curves were generated to help determine if there is an ideal cutoff to be used. An ideal ROC has a true positive ratio approaching 1 and a false positive ratio approaching 0 across a broad range of cutoffs, resulting in a large area under the curve. Our data demonstrates that this desirable binning behavior is obtained for multistate methods but not for SSD (Fig. 8), further demonstrating the improved accuracy of MSA and MSD calculations relative to SSD.

3.5 Predicted Sequence Space Analysis

In this section, diversity of the top 100 sequences predicted by the various CPD methods will be analyzed. To do so, the frequency of amino-acid residues found at each designed position as well as the number of identical sequences in the top 100 sequences predicted by each method will be compared.

1. Extract the list of top 100 sequences predicted by SSD, MSA, and MSD.

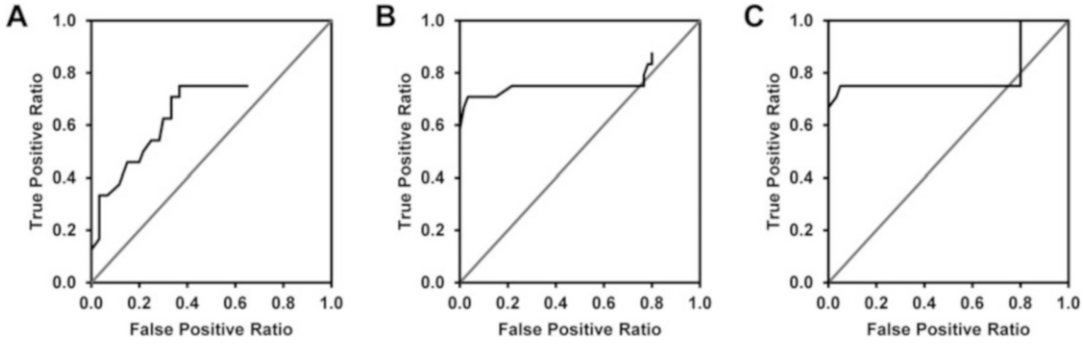


Fig. 8 Receiver operating characteristic (ROC) curves. ROC curves for SSD (**a**), MSA (**b**), and MSD (**c**) calculations were produced by binning Gβ1 training set sequences with respect to an energy cutoff between −90 and +90 kcal/mol that was increased in 1 kcal/mol increments. ROC curves for SSD and MSA do not reach true and false positive ratios of 1.0 because the energy of some training set sequences was predicted to be greater than +90 kcal/mol. The *diagonal gray line* indicates random sequence binning

2. Use the Weblogo server (http://weblogo.berkeley.edu/) [38] to compute the frequency of each amino-acid type found at each designed position in the top 100 sequences.
3. Compare sequence logos obtained from the top 100 sequences predicted by each CPD method.

As shown in Fig. 9, amino-acid diversity at each designed position of Gβ1 is nearly identical in the top 100 sequences predicted by MSA and MSD. However, sequence diversity obtained by SSD is significantly different, in particular at positions 5, 30, and 52. For example, many sequences predicted by SSD contain an Ile at position 5 or 52, or do not include Leu or Ile substitutions at position 30, in contrast with sequences predicted by the multistate methods. The highly similar amino-acid diversity at each designed position obtained by the multistate methods suggests that their top 100 sequences are nearly identical. To verify whether this is true, we compared the overlap in identical sequences contained in the top 100 sequences predicted by the various CPD methods. We found that MSA and MSD share 89 of their top 100 ranked sequences, confirming that these methods predict nearly identical top 100 sequences. In contrast, MSA and MSD share a much lower number of their top 100 sequences with SSD (54 or 51, respectively).

4 Conclusions

We have described three CPD methods that can be used for the prediction of mutant sequence stabilities. Of these, the multistate approaches result in improved prediction accuracy by addressing the fixed backbone approximation via the incorporation of backbone ensembles that simulate protein conformational flexibility.

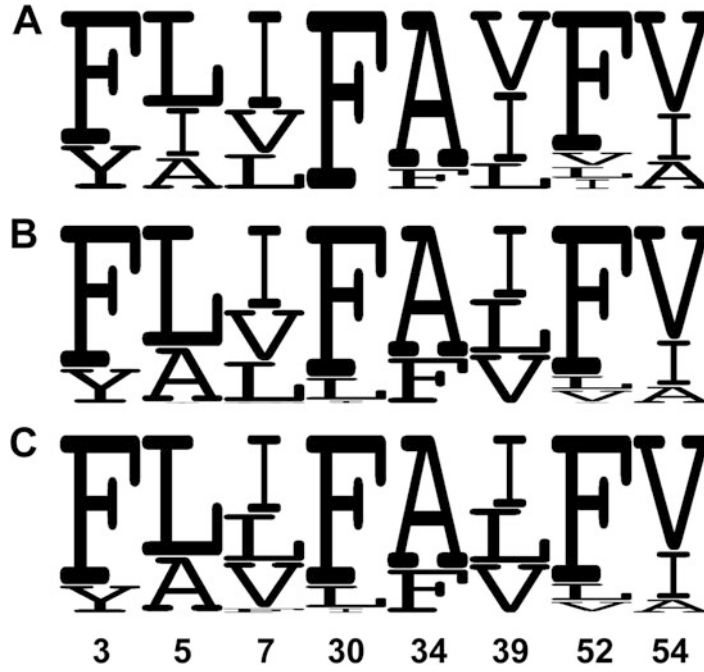


Fig. 9 Amino-acid diversity found at Gβ1 designed positions in the top 100 ranked sequences predicted by various computational protein design methods. Sequence logos for SSD (**a**), MSA (**b**), and MSD (**c**), are shown with amino-acid substitution frequency proportional to letter height. Designed positions are indicated by *numbers*. Sequence logos were prepared using WebLogo 2.8.2 [38]

Superior prediction accuracy is afforded by improved sequence scoring that results in fewer false negative predictions. We also described the PertMin ensemble generation method, which is easy to implement, computationally inexpensive, and generally applicable for the creation of backbone ensembles to be used in multistate CPD methods. Because of the benefits highlighted above, multistate CPD with PertMin backbone ensembles represents a valuable addition to the protein engineering toolbox.

5 Notes

1. Gβ1 structure preparation requires the addition of hydrogen atoms to the 1PGA crystal structure, as well as the inclusion of counter-ions and solvent water molecules. Hydrogen atoms were added at pH 7 using the Protonate3D utility [39] included in MOE [27], which facilitates the optimal placement of hydrogens by considering multiple configurations and protonation states. Hydrogen configurations were adjusted by Unary Quadratic Optimization using a 12-6 Lennard-Jones potential and a distance-dependent dielectric of 1. Alternatively, hydrogens can

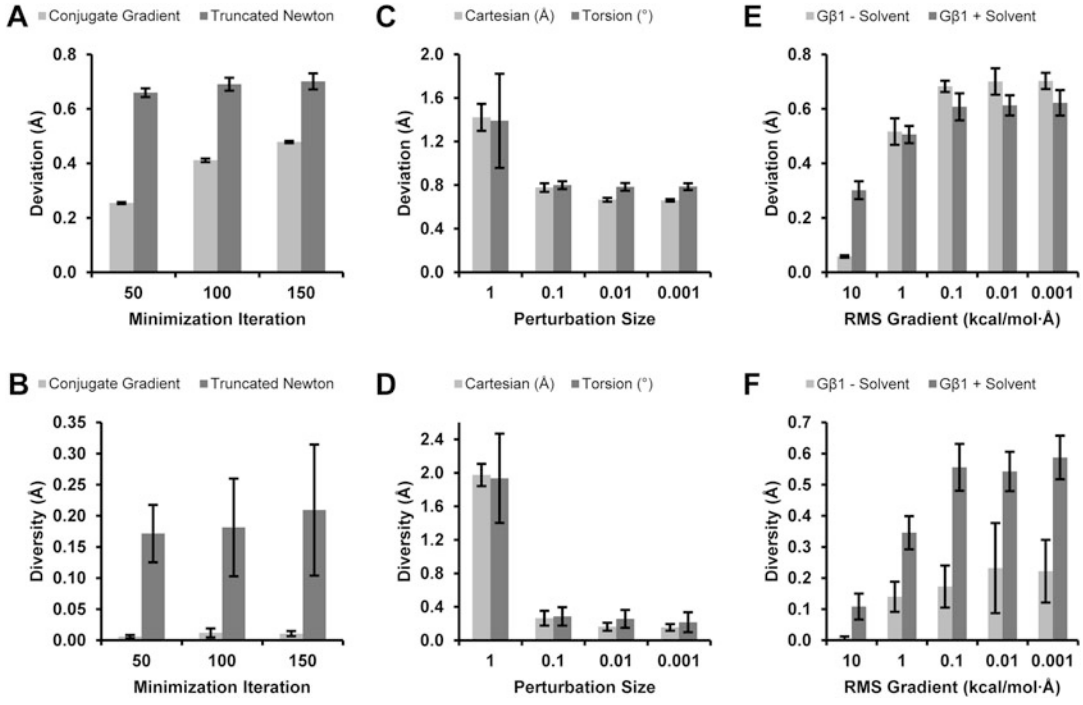


Fig. 10 Tuning of PertMin ensemble properties. Various 30-member Gβ1 ensembles were prepared with variations on the following PertMin protocol: Cartesian coordinate perturbations of ± 0.001 Å along each axis followed by 50 iterations of truncated Newton energy minimization in the absence of water solvent molecules. Effect of energy minimization algorithm and number of iterations on ensemble root-mean-square (RMS) coordinate deviation (**a**) and diversity (**b**). Effect of type and size of random perturbations on ensemble deviation (**c**) and diversity (**d**). Effect of system size and energy minimization RMS gradient on ensemble deviation (**e**) and diversity (**f**). To increase system size, protein structures were solvated in a box of water molecules with a depth of 3 Å. In order to compare systems of different sizes, they should occupy regions on the potential energy surface located at similar distances to the nearest minimum. Therefore, energy minimizations were terminated at specific RMS gradients (**e** and **f**) instead of at specific numbers of minimization iterations

be added using other tools such as Reduce [40] or MolProbity [41]. Following hydrogen addition, MOE was used to add counter-ions (Na^+ and Cl^-) to neutralize surface charges and water molecules to give a box with a depth of a 6 Å around the protein surface in a periodic boundary.

2. Ensemble properties, such as its RMS backbone coordinate deviation from the input structure (deviation) or backbone coordinate similarity between ensemble members (diversity), can be tuned by altering the PertMin protocol. For example, when generating a 30-member Gβ1 ensemble, the choice of energy minimization algorithm can influence ensemble properties. Energy minimization with the truncated Newton algorithm produces an ensemble with higher deviation (Fig. 10a)

and diversity (Fig. 10b) than the one produced with conjugate gradient minimization. This is because truncated Newton is a second-order energy minimization algorithm that is more sensitive to initial conditions [42] than conjugate gradient, which is first order. In addition, average deviation and diversity increase with the number of minimization iterations, regardless of minimization algorithm, to a maximum value that is dependent on the location of energy minima on the protein potential energy surface.

Different types of perturbations (torsion or Cartesian) applied to the input structure coordinates will yield ensembles with similar average deviation (Fig. 10c) and diversity (Fig. 10d). Perturbation magnitude does not significantly affect deviation or diversity when kept at values below or equal to 0.1 Å or degree. This is because small perturbations result in similar perturbed structures occupying the same region of the potential energy surface, making accessible the same local minima. When the perturbation is sufficiently large (1 Å or degree), another region of the potential energy surface and a different set of local minima become accessible, resulting in ensembles with larger deviation and diversity.

System size, i.e., the number of atoms subjected to energy minimization, will also affect the rate at which ensemble deviation (Fig. 10e) and diversity (Fig. 10f) increase. The more atoms are included in the energy minimization calculation, for example by addition of solvent molecules, the fewer number of iterations are required to produce the same amount of deviation and diversity. In all cases, whether altering the energy minimization protocol, perturbation method, or system size, a maximum deviation and diversity is reached. This is because PertMin generates structures at local minima, which are fixed on the potential energy surface specific to the system.

3. The Boltzmann weighted average is calculated for an ensemble containing n members with the following equation: $E = \frac{\sum_{i=1}^{n} E_i \cdot e^{(-E_i/kT)}}{\sum_{i=1}^{n} e^{(-E_i/kT)}}$, where k is the Boltzmann constant, T is the temperature (300 K in this case), and E is the energy. Evaluation of sequence fitness as the Boltzmann weighted average ensures that sequences that stabilize a majority of ensemble members are not penalized if they destabilize a few. Alternatively, the energy of a sequence on its most favorable scoring state can also be used as its fitness value [24].
4. The number of backbone templates included in an ensemble (i.e., ensemble size) can affect predictions made by MSD. For example, MSD performed using a 128-member PertMin ensemble results in the most favorable WT sequence fitness (Fig. 11a), the highest success rate (Fig. 11b), and the lowest

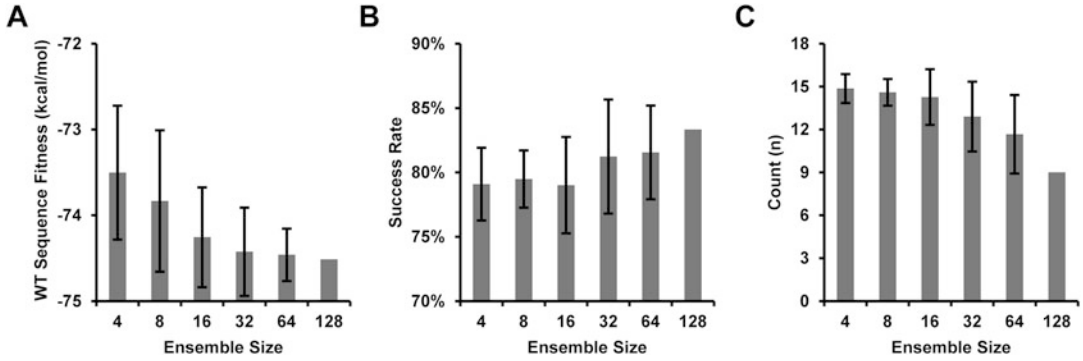


Fig. 11 MSD with ensembles of various sizes. MSD was performed with a single 128-member Gβ1 ensemble or with 30 ensembles containing random combinations of 4, 8, 16, 32, or 64 backbones extracted from the 128-member ensemble. The 128-member ensemble was prepared as described in Subheading 3.2, with the exception that solvent water molecules were added to a depth of 3 Å. (**a**) The average wild-type (WT) sequence energy becomes more favorable as ensemble size increases. (**b**) The average success rate increases with ensemble size. (**c**) The average number of false negatives decreases with ensemble size. Error bars show the standard deviation of 30 independent MSD calculations

number of false negatives (Fig. 11c). As the ensemble size decreases, fitness of the WT sequence increases in energy, average success rate decreases, and the number of false negative sequences increases. Nevertheless, MSD using 64-member ensembles gives results similar to those obtained with the 128-member ensemble. While we recommend using an ensemble containing at least 64 templates, MSD with a small 4-member ensemble is still preferable to SSD with a single backbone template because it results in a higher success rate and fewer false negatives.

References

1. Koga N, Tatsumi-Koga R, Liu G, Xiao R, Acton TB, Montelione GT, Baker D (2012) Principles for designing ideal protein structures. Nature 491(7423):222–227. doi:10.1038/nature11600
2. Murphy GS, Sathyamoorthy B, Der BS, Machius MC, Pulavarti SV, Szyperski T, Kuhlman B (2015) Computational de novo design of a four-helix bundle protein-DND_4HB. Protein Sci 24(4):434–445. doi:10.1002/pro.2577
3. Kuhlman B, Dantas G, Ireton GC, Varani G, Stoddard BL, Baker D (2003) Design of a novel globular protein fold with atomic-level accuracy. Science 302(5649):1364–1368. doi:10.1126/science.1089427
4. Privett HK, Kiss G, Lee TM, Blomberg R, Chica RA, Thomas LM, Hilvert D, Houk KN, Mayo SL (2012) Iterative approach to computational enzyme design. Proc Natl Acad Sci U S A 109(10):3790–3795, doi:1118082108 [pii] 10.1073/pnas.1118082108
5. Kapp GT, Liu S, Stein A, Wong DT, Remenyi A, Yeh BJ, Fraser JS, Taunton J, Lim WA, Kortemme T (2012) Control of protein signaling using a computationally designed GTPase/GEF orthogonal pair. Proc Natl Acad Sci U S A 109(14):5277–5282. doi:10.1073/pnas.1114487109
6. Siegel JB, Zanghellini A, Lovick HM, Kiss G, Lambert AR, St Clair JL, Gallaher JL, Hilvert D, Gelb MH, Stoddard BL, Houk KN, Michael FE, Baker D (2010) Computational design of an enzyme catalyst for a stereoselective bimolecular Diels-Alder reaction. Science

329(5989):309–313. doi:10.1126/science.1190239
7. Frey KM, Georgiev I, Donald BR, Anderson AC (2010) Predicting resistance mutations using protein design algorithms. Proc Natl Acad Sci U S A 107(31):13707–13712. doi:10.1073/pnas.1002162107
8. Dahiyat BI (1999) In silico design for protein stabilization. Curr Opin Biotechnol 10 (4):387–390. doi:10.1016/S0958-1669(99)80070-6
9. Kuhlman B, Choi EJ, Guntas G (2009) Future challenges of computational protein design. In: Park SJ, Cochran JR (eds) Protein engineering and design. CRC Press, Boca Raton, FL. doi:10.1201/9781420076592.ch18
10. Kellogg EH, Leaver-Fay A, Baker D (2011) Role of conformational sampling in computing mutation-induced changes in protein structure and stability. Proteins 79(3):830–838. doi:10.1002/prot.22921
11. Dahiyat BI, Mayo SL (1997) Probing the role of packing specificity in protein design. Proc Natl Acad Sci U S A 94(19):10172–10177
12. Grigoryan G, Ochoa A, Keating AE (2007) Computing van der Waals energies in the context of the rotamer approximation. Proteins 68 (4):863–878. doi:10.1002/prot.21470
13. Murphy GS, Mills JL, Miley MJ, Machius M, Szyperski T, Kuhlman B (2012) Increasing sequence diversity with flexible backbone protein design: the complete redesign of a protein hydrophobic core. Structure 20 (6):1086–1096. doi:10.1016/j.str.2012.03.026
14. Ollikainen N, Smith CA, Fraser JS, Kortemme T (2013) Flexible backbone sampling methods to model and design protein alternative conformations. Methods Enzymol 523:61–85. doi:10.1016/B978-0-12-394292-0.00004-7
15. Smith CA, Kortemme T (2011) Predicting the tolerated sequences for proteins and protein interfaces using RosettaBackrub flexible backbone design. PLoS One 6(7), e20451. doi:10.1371/journal.pone.0020451
16. Wang C, Schueler-Furman O, Baker D (2005) Improved side-chain modeling for protein-protein docking. Protein Sci 14 (5):1328–1339. doi:10.1110/ps.041222905
17. Borgo B, Havranek JJ (2012) Automated selection of stabilizing mutations in designed and natural proteins. Proc Natl Acad Sci U S A 109(5):1494–1499. doi:10.1073/pnas.1115172109
18. Gainza P, Roberts KE, Donald BR (2012) Protein design using continuous rotamers. PLoS Comput Biol 8(1), e1002335. doi:10.1371/journal.pcbi.1002335
19. Allen BD, Nisthal A, Mayo SL (2010) Experimental library screening demonstrates the successful application of computational protein design to large structural ensembles. Proc Natl Acad Sci U S A 107(46):19838–19843. doi:10.1073/pnas.1012985107
20. Davey JA, Chica RA (2014) Improving the accuracy of protein stability predictions with multistate design using a variety of backbone ensembles. Proteins 82(5):771–784. doi:10.1002/prot.24457
21. Allen BD, Mayo SL (2010) An efficient algorithm for multistate protein design based on FASTER. J Comput Chem 31(5):904–916. doi:10.1002/jcc.21375
22. Leaver-Fay A, Jacak R, Stranges PB, Kuhlman B (2011) A generic program for multistate protein design. PLoS One 6(7), e20937. doi:10.1371/journal.pone.0020937
23. Yanover C, Fromer M, Shifman JM (2007) Dead-end elimination for multistate protein design. J Comput Chem 28(13):2122–2129. doi:10.1002/jcc.20661
24. Howell SC, Inampudi KK, Bean DP, Wilson CJ (2014) Understanding thermal adaptation of enzymes through the multistate rational design and stability prediction of 100 adenylate kinases. Structure 22(2):218–229. doi:10.1016/j.str.2013.10.019
25. Babor M, Mandell DJ, Kortemme T (2011) Assessment of flexible backbone protein design methods for sequence library prediction in the therapeutic antibody Herceptin-HER2 interface. Protein Sci 20(6):1082–1089. doi:10.1002/pro.632
26. Williams CI, Feher M (2008) The effect of numerical error on the reproducibility of molecular geometry optimizations. J Comput Aided Mol Des 22(1):39–51. doi:10.1007/s10822-007-9154-7
27. Chemical Computing Group Inc (2012) Molecular operating environment (MOE) 2012, 14th edn. Chemical Computing Group Inc, Montreal, QC
28. Chica RA, Moore MM, Allen BD, Mayo SL (2010) Generation of longer emission wavelength red fluorescent proteins using computationally designed libraries. Proc Natl Acad Sci U S A 107(47):20257–20262. doi:10.1073/pnas.1013910107
29. Gallagher T, Alexander P, Bryan P, Gilliland GL (1994) Two crystal structures of the B1 immunoglobulin-binding domain of streptococcal protein G and comparison with NMR. Biochemistry 33(15):4721–4729
30. Leach AR (1998) Molecular modelling: principles and applications. Longman, Harlow

31. Nash SG (2000) A survey of truncated-Newton methods. J Comput Appl Math 124 (1–2):45–59. doi:10.1016/S0377-0427(00)00426-X
32. Dunbrack RL, Cohen FE (1997) Bayesian statistical analysis of protein side-chain rotamer preferences. Protein Sci 6(8):1661–1681
33. Mayo SL, Olafson BD, Goddard WA (1990) Dreiding – a generic force-field for molecular simulations. J Phys Chem 94(26):8897–8909. doi:10.1021/J100389a010
34. Lazaridis T, Karplus M (1999) Effective energy function for proteins in solution. Proteins 35 (2):133–152. doi:10.1002/(Sici)1097-0134(19990501)35:2<133::Aid-Prot1>3.0.Co;2-N
35. Street AG, Mayo SL (1998) Pairwise calculation of protein solvent-accessible surface areas. Fold Des 3(4):253–258. doi:10.1016/S1359-0278(98)00036-4
36. Desmet J, Spriet J, Lasters I (2002) Fast and accurate side-chain topology and energy refinement (FASTER) as a new method for protein structure optimization. Proteins 48(1):31–43. doi:10.1002/Prot.10131
37. Allen BD, Mayo SL (2006) Dramatic performance enhancements for the FASTER optimization algorithm. J Comput Chem 27 (10):1071–1075. doi:10.1002/jcc.20420
38. Crooks GE, Hon G, Chandonia JM, Brenner SE (2004) WebLogo: a sequence logo generator. Genome Res 14(6):1188–1190. doi:10.1101/Gr.849004
39. Labute P (2009) Protonate3D: assignment of ionization states and hydrogen coordinates to macromolecular structures. Proteins 75 (1):187–205. doi:10.1002/Prot.22234
40. Word JM, Lovell SC, Richardson JS, Richardson DC (1999) Asparagine and glutamine: using hydrogen atom contacts in the choice of side-chain amide orientation. J Mol Biol 285 (4):1735–1747. doi:10.1006/jmbi.1998.2401
41. Davis IW, Leaver-Fay A, Chen VB, Block JN, Kapral GJ, Wang X, Murray LW, Arendall WB III, Snoeyink J, Richardson JS, Richardson DC (2007) MolProbity: all-atom contacts and structure validation for proteins and nucleic acids. Nucleic Acids Res 35((Web Server issue)):375–383. doi:10.1093/nar/gkm216
42. Davey JA (2011) On the energy minimization of large molecules, M.Sc. thesis. Carleton University, Canada, Ottawa, ON

Chapter 8

Integration of Molecular Dynamics Based Predictions into the Optimization of De Novo Protein Designs: Limitations and Benefits

Henrique F. Carvalho, Arménio J.M. Barbosa, Ana C.A. Roque, Olga Iranzo*, and Ricardo J.F. Branco*

Abstract

Recent advances in de novo protein design have gained considerable insight from the intrinsic dynamics of proteins, based on the integration of molecular dynamics simulations protocols on the state-of-the-art de novo protein design protocols used nowadays. With this protocol we illustrate how to set up and run a molecular dynamics simulation followed by a functional protein dynamics analysis. New users will be introduced to some useful open-source computational tools, including the GROMACS molecular dynamics simulation software package and ProDy for protein structural dynamics analysis.

Key words Protein essential dynamics, Principal component analysis, Normal mode analysis, Elastic network models, Internal molecular dynamics

1 Introduction

The design of innovative and versatile biocatalysts that are more robust and catalytically proficient than the native ones found in Nature for specific bioconversion in a given reactional media has long been pursued [3]. The discovery of enzymatic activity dates back to the end of nineteenth century, with the isolation and characterization of amylase and urease enzymes. Since then, it has been realized that enzymes are highly efficient nanoscale machines, which are able to outperform chemical reactions specifically as no other catalyst-based system developed so far, with rate enhancements (kcat/knon) up to 10^{19}-fold relative to the uncatalyzed reaction [2].

Naively, one can think that it would be easy and straightforward to recapitulate the mode of action, as well as catalytic features of

* Corresponding authors

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_8,

natural enzymes into a given protein scaffold of interest by simply applying the mechanistic rules and structural constraints theoretically predicted for efficient biocatalysis. However, reality has proved to be much more complex and even when all catalytic determinants are gathered in a perfect theoretical active site model—theozyme, term coined by the seminal work of Houk's lab in 1998 [3], the few successful cases of de novo protein design showed a considerable gap between their catalytic efficiency (k_{cat}/K_M of 10^4–10^5 M^{-1} s^{-1}) and those from the natural occurring enzymes (k_{cat}/K_M of 10^6–10^8 M^{-1} s^{-1}) [17]. This gap corresponds to more than four orders of magnitude away from the diffusion rate limit.

Several strategies have been implemented to circumvent this apparent theoretical design paradox of low activity, namely the integration of molecular dynamics (MD) based predictions into the state-of-the-art protein design protocols [18]. The flexibility shown by protein structures is essential, allowing conformational changes during catalysis which are required for the substrate binding, product release, or for many other functional related motions, as in the case of *Candida antarctica* Lipase B loop movement that is responsible for the solvent accessibility to the active site [4]. However, there is a hierarchy of motions from low-frequency interdomain hinge motions to high-frequency bond vibrational motions that occur at considerable different range, in the femtosecond time scale, that needs to be considered. The preorganized enzyme active site drives the Michaelis-Menten complex formation toward the most reactive set of conformations around the transition state to maximize catalytic efficiency. This means that the protein scaffold cannot be treated as a rigid body and has an intrinsic dynamics that has to be taken into account during the computationally driven protein design [1].

1.1 Turning a Protein Design into an Active Enzyme

The de novo protein design strategy, has been applied successfully to only few biotransformations, like the one applied to the quantum mechanics-based (QM) active site design of a Kemp eliminase, a biocatalyst that performs a reaction not catalyzed by any other naturally occurring enzyme [5]. This strategy starts with the definition of the most suitable catalytic mechanism and plausible transition state (TS) geometry for a given reaction—theozyme. Then, the corresponding transition state geometry will be quantum-mechanically determined either using small gas phase models or more accurately, using hybrid QM/MM approachs that take also into account the impact of the protein environment on the active site's electronic structure, calculated at lower molecular mechanical (MM) level. This precisely depicted TS model at atomic level might then be crafted in the protein scaffold using a MM modeling software such as RosettaMatch [6]. This step searches for putative protein scaffold candidates that are able to host the theozyme model, ensuring the TS conformation to be placed in the correct geometry and protein neighborhood that maximize its stabilization, without substantial steric clashes or electrostatic conflicts.

Finally, the new competent catalytic pocket needs to be redesigned to maximize the stability of the entire active-site conformation, the integrity of the TS geometry, and the affinity for the substrate to bind efficiently through RosettaEnzDes software module [6].

However, experimental characterization of computationally designed enzymes has shown some limitations caused essentially by nonoptimal polar interactions with the substrate, inactive conformation of the substrate in the bound state, or simply inadequate solvent-mediated contacts to promote the stabilization of the protein-substrate complex [1]. At this stage, knowledge from MD reveals to be essential to iteratively improve protein designs efficiency. A systematic population analysis of the most stable substrate binding modes, essential dynamics, and preferential molecular interactions might reveal structural limitations of a putative scaffold and shed light on the MD-assisted design refinement, leading to more active enzymes.

2 Materials

Useful links for accessing open-source software and webservers undermentioned:

1. GROMACS (http://www.gromacs.org/).
2. Propka (http://propka.ki.ku.dk/).
3. PDB (http://www.rcsb.org/pdb/home/home.do).
4. ProDy (http://prody.csb.pitt.edu/).
5. Bio3D (http://thegrantlab.org/bio3d/index.php).
6. VMD (http://www.ks.uiuc.edu/Research/vmd/).
7. Pymol (https://www.pymol.org/).

3 Methods

Molecular dynamics simulation of a protein in explicit water solvent.

3.1 Preparation of Protein Structure for a MD Simulation

3.1.1 Check and Clean Up the Protein Structure of Interest

In order to obtain a reliable result and avoid computational bias, the protein structure and simulation conditions have to be carefully inspected and set up. This protocol was developed to work on a Linux environment with the open-source simulation package GROMACS 4.6.1 version installed. Due to the continuous development of simulation software, some command line might have to be rephrased in future releases, whenever necessary. The standard MD protocol used here for a case study example is described as follows:

1. For the MD protocol description, the crystallographic structure of a xylanase from *B. circulans* (PDB code 3LB9) [7] was selected and downloaded from the RCSB Protein Data Bank [8] in the corresponding *.pdb file format, which will be referred thereafter as *PROTEIN* for convenience.
2. The *RCSB_PROTEIN*.pdb structure has to be first inspected with a visualization software such as the PyMOL Molecular Graphics System Version 1.2 Schrödinger, LLC. The remark of "*MISSING*" residues or side chain atoms in the PDB file, as well as any other "*HETATM*" cocrystalized with the protein structure, has to be taken into account. For the MD simulation of an unbounded protein structure, all the crystallographic water molecules and ligands were discarded and only the three-dimensional coordinates of the bare protein structure were saved separately in a new *PROTEIN*.pdb file.
3. Before setting up the simulation box, the corresponding *pKa* of chargeable amino acids and terminal groups must be checked out, as well as the number of structural cysteine sulfur bridges must be determined. The protonation state of titratable side chains and N-/C-terminus, including Arg, Lys, His, Glu, and Asp, might be determined based on the pK_a prediction using, for example, the Propka software, available at the webserver (http://propka.ki.ku.dk/) [9].

3.1.2 Setting Up the Simulation System and Input Files

1. To convert the *PROTEIN.pdb* structure into the compatible file format of GROMACS, the *pdb2gmx* command was used to generate the corresponding coordinate file *PROTEIN.gro*, the topology file *PROTEIN.top*, and the position constraints file *PROTEIN.itp* for the equilibration phase. Additionally, the solvent water model SPCE and the protonation state of titrable amino acid and terminals of the protein chain might be called by using the *–water* and *–inter* flags, respectively. The unified-atom GROMOS force field 53A6 implemented in GROMACS [10] was used in this protocol workflow.

```
$ pdb2gmx –f [PROTEIN.pdb] –o [PROTEIN.gro] –p [PROTEIN.top] –i [PROTEIN.itp] –water [spce] –inter
 $4 #Gromos96 ff53A6 force field selection
```

Note: "#" symbol stands for a programming line annotation, which should not be parsed into the command line at the prompt.

2. In order to resize and center the protein structure in an explicit solvent box, the coordinates file *PROTEIN.gro* will be rewritten by the *editconf* tool of GROMACS software package. The indication of the box symmetry was added with the *–bt* flag, as

well as the minimum offset distance (in nm) between the protein surface and the box edges was defined by the *–d* flag. This set up prevents unrealistic interactions of the protein with surrounding images, due to the periodic boundary conditions. To efficiently reduce the computational costs of simulation, the rhombic octahedron box fits better the globular shape of *PROTEIN* in use, optimizing the number of water molecules to be simulated together with the protein solute.

```
$ editconf –f [PROTEIN.gro] –o [PROTEIN.gro] –bt [octahedron] –d [1.2]
```

The *PROTEIN.gro* input file will then be overwritten and new box dimensions updated accordingly at the end of input file. To avoid file name conflicts a sequential numbering of input files might be given. Otherwise, the GROMACS code automatically renames the old files with the prefix *"#"*.

3. Then, the new *PROTEIN.gro* coordinate file centered in the new solvent box size will be filled with a pre-equilibrated box, containing 216 water molecules, by calling the *genbox* tool of GROMACS simulation software package.

```
$ genbox –cp [PROTEIN.gro] –cs [spc216.gro] –p [PROTEIN.top] –o [PROTEIN.gro]
```

4. For neutralizing the total formal charge of the system, the addition of counterions is needed. In this case, the GROMACS preprocessor script *grompp* is required to generate first the GROMACS portable binary run input *PROTEIN.tpr* file. A set of exemplary input *template_*.mdp* files specific for each phase of the simulation run can be accessed from Subheading 4. These files define explicitly the appropriate set of parameters to be used, in a sequential order, along the entire simulation process, including: (a) energy minimization; (b) equilibration; and (c) MD production phases.

```
$ grompp –f [template_*.mdp] –c [PROTEIN.gro] –p [PROTEIN.top] –o [PROTEIN.tpr]
```

5. The *genion* tool will then replace an equal number of water molecules in the coordinate file by the corresponding number of chosen counterions defined interactively by the selection of the group of atoms for replacement. The *–pname (-np)* and *–nname (-nn)* flags stand for the nature (and number) of positive or negative counterions to be added, respectively. The topology and coordinates files will be automatically updated accordingly.

```
$ genion –s [PROTEIN.tpr] –o [PROTEIN.gro] –p [PROTEIN.top] –g [PROTEIN.log]
–pname/-nname [NA]/[CL] –np/-nn ["x"] # "x" is the total charge of the system.
    $ 15 # group of SOL for replacement
```

3.1.3 Running the MD Simulation for Trajectory Production

1. With the aim to mitigate bad contacts or any atomic clashes due to inappropriate side chains configuration, a preliminary energy minimization divided into two sequential cycles and using different minimization algorithms was required.

```
; 1st Step energy minimization with the Steepest Descend algorithm
$ grompp –f [template_em1.mdp] –c [PROTEIN.gro] –p [PROTEIN.top] –o
[PROTEIN_em1.tpr]
$ mdrun –v –s [PROTEIN_em1.tpr] –c [PROTEIN_em1.gro] –e [PROTEIN_em1.edr] –g
[PROTEIN_em1.log]
; 2nd Step energy minimization with the Conjugated Gradient algorithm
$ grompp –f [template_em2.mdp] –c [PROTEIN_em1.gro] –p [PROTEIN.top] –o [PRO-
TEIN_em2.tpr]
$ mdrun –v –s [PROTEIN_em2.tpr] –c [PROTEIN_em2.gro] –e [PROTEIN_em2.edr] –g
[PROTEIN_em2.log]
```

2. After the energy convergence of two preliminary minimization cycles, the system was then coupled to a thermostat (e.g., the V-rescale modified Berendsen) and to a barostat, according to the specifications defined on the input *template_*mdp* file. The equilibration phase was carried out in three sequential simulation steps, of 100 ps each, in an isothermal-isobaric NPT ensemble. A positional restrain was imposed to the protein backbone heavy atoms, defined by the *–DPOSRES* flag of the corresponding *template_eq1-3.mdp* files. The imposed harmonic force constants for the positional restrain are written in the *PROTEIN.itp* file and changed stepwise from 1000/100/10 kJ/mol.

```
; 1st Equilibration Step
$ grompp –f [template_eq1.mdp] –c [PROTEIN_em2.gro] –p [PROTEIN.top] –o
[PROTEIN_eq1.tpr]
$ mdrun –v –s [PROTEIN_eq1.tpr] –c [PROTEIN_eq1.gro] -e [PROTEIN_eq1.edr] -g
[PROTEIN_eq1.log]
$ perl -pi -e 's/ 1000/ 100/g' PROTEIN.itp
; 2nd Equilibration Step
$ grompp –f [template_eq2.mdp] –c [PROTEIN_eq1.gro] –p [PROTEIN.top] –o
[PROTEIN_eq2.tpr]
$ mdrun –v –s [PROTEIN_eq2.tpr] –c [PROTEIN_eq2.gro] -e [PROTEIN_eq2.edr] -g
[PROTEIN_eq2.log]
```

```
$ perl -pi -e 's/ 100/ 10/g' PROTEIN.itp
; 3rd Equilibration Step
$ grompp –f [template_eq3.mdp] –c [PROTEIN_eq2.gro] –p [PROTEIN.top] –o
[PROTEIN_eq3.tpr]
$ mdrun –v –s [PROTEIN_eq3.tpr] –c [PROTEIN_eq3.gro] -e [PROTEIN_eq3.edr] -g
[PROTEIN_eq3.log]
```

Equilibration phase is intended to gradually adjust the temperature and pressure of the system preventing any unphysical and irreversible structural deformation or unfolding event that could compromise the canonical ensemble of protein conformations sampled during the simulation trajectory.

3. Finally, the system is equilibrated and production phase takes place freely, without any positional constraint.

```
$ grompp –f [template_md1.mdp] –c [PROTEIN_eq3.gro] –p [PROTEIN.top] –o
[PROTEIN_md1.tpr]
$ mdrun –v –s [PROTEIN_md1.tpr] –c [PROTEIN_md1.gro] -e [PROTEIN_md1.edr]
-g [PROTEIN_md1.log] –cpo state.cpt
```

3.2 Essential Dynamics Analysis of Designed Proteins

This type of analysis can be performed for any protein of interest from which a Molecular Dynamics (MD) simulation trajectory is available, using software packages such as ProDy or Bio3D [11, 12]. The advantage of using such software packages stands for the ease of integration, manipulation, and comparison of data obtained from different models. These tools are usually open-source and require basic skills and familiarity with programming languages as Python or R.

3.2.1 Preparation of Files for Essential Dynamics Analysis: Concatenation of Trajectory Files

The results presented were obtained from a 20 ns MD simulation of *B. circulans* xylanase (PDB ID: 3LB9). For other proteins, simply replace the term *PROTEIN* for the PDB Identifier code or name attributed to the protein of interest. If multiple trajectory files (**.trr or *.xtc format*) are used, it is necessary to first concatenate them using, for example, the GROMACS suit of tools for analysis, namely the *trjcat* script. More information on *trjcat* can be found here: ftp://ftp.gromacs.org/pub/manual/manual-4.6.7.pdf, Section D. 93. In this example, three consecutive equilibration steps with 100 ps each (*PROTEIN_eq1-3.trr*) are concatenated with a production phase trajectory file (*PROTEIN_md1.trr),* being specified as input files with the *-f* flag. The output file name (*PROTEIN_concatenated.xtc*) is specified with the *-o* flag. To concatenate the corresponding **.trr* or **.xtc* files in a sorted order, a new start time for each file is required. This can be performed interactively by using the *-settime* flag

and then specifying the correspondent starting time of each trajectory fragment in a row.

Commands can be executed either interactively on the prompt or in a bash or shell script launched by the user as described in the following boxes:

```
$trjcat -f PROTEIN_eq1.trr PROTEIN_eq2.trr PROTEIN_eq3.trr PROTEIN_md1.trr -o
PROTEIN_concatenated.xtc -settime
$0     #start time for eq1 trajectory fragment
$100 #start time for eq2 trajectory fragment
$200 #start time for eq3 trajectory fragment
$300 #start time for md1 trajectory fragment
```

The resulting concatenated **.xtc* file now contains the entire simulation, i.e., all the written frames of the MD simulation trajectory, which can then be used for further analysis.

3.2.2 Processing of Trajectory Files

The *trjconv* tool is another useful postprocessing tool implemented in GROMACS that may be used for several purposes, including extracting specific frames from trajectory or simply correcting computational artifacts caused by the use of periodic boundary conditions. More information on it should be found here: ftp://ftp.gromacs.org/pub/manual/manual-4.6.7.pdf, Section D. 94. In this example, a script was used to obtain a trajectory file of the protein that will be suitable for further analysis:

1. Obtain a reference frame of the system (0) from the concatenated input trajectory file (*PROTEIN_concatenated.xtc)*, by defining the same starting and final time with *-b* and *-e* flags, respectively. The option *-pbc whole* is used to correct periodicity artifacts like broken molecules at the edges of solvent box replicas:

```
$trjconv -f PROTEIN_concatenated.xtc -o PROTEIN_concatenated_0ps.gro -b 0.0 -e 0.0 -
s PROTEIN_md1.tpr -pbc whole
$0 #group of atoms for output corresponding to the entire system, including solvent and solutes
```

2. Removal of jumps caused by the periodic boundary conditions from the trajectory, by using the option *-pbc nojump* and setting the correct time step between frames (ps) with the *-timestep* flag, corresponding to the continuous integration time in fentoseconds.

```
$trjconv -f PROTEIN_concatenated.xtc -s PROTEIN_concatenated_0ps.gro -o
PROTEIN_nojump.xtc -pbc nojump -timestep 2
$0 #group of atoms for output
```

3. Solvent molecules can be discarded, since in this example the analysis is only focused on the protein structure (*1*):

```
$trjconv -f PROTEIN_nojump.xtc -o PROTEIN_nojump.gro -b 0.0 -e 0.0 -s PROTEIN_con-
catenated_0ps.gro
$1 #group of atoms for output corresponding to the protein
```

4. Centering the protein atoms in the box (*1*) with the *-center* flag:

```
$trjconv -f PROTEIN_nojump.xtc -s PROTEIN_nojump.gro -o PROTEIN_center.xtc -
center -timestep 2
$1 #group of atoms for centering
$1 #group of atoms for output
```

A new centered reference frame is also acquired, corresponding to the first frame of centered trajectory:

```
$trjconv -f PROTEIN_center.xtc -o PROTEIN_center.gro -b 0.0 -e 0.0 -s PROTEIN_no-
jump.gro
$1 #group of atoms for output
```

The resulting reference frame (*PROTEIN_center.gro*) and trajectory file (*PROTEIN_center.xtc*) may be used for further analysis.

3.2.3 Inspection of Trajectories

Trajectory files can be inspected with molecular visualization software such as VMD [13] (http://www.ks.uiuc.edu/Research/vmd/). A full description of the type of analysis typically implemented is beyond the scope of this protocol. In this case, only the portion of the trajectory where RMSD converged to a structural stabilization plateau is considered for further analysis (Fig. 1). This can be evaluated by the least-squares fitting of the protein backbone atoms of each frame to a given reference structure, usually the initial one, to discount the diffusional protein movements in solution, namely the typical global rotational and translational movements. VMD is also helpful to generate a trajectory file in *****.*dcd* format with the corresponding segment of the trajectory. The

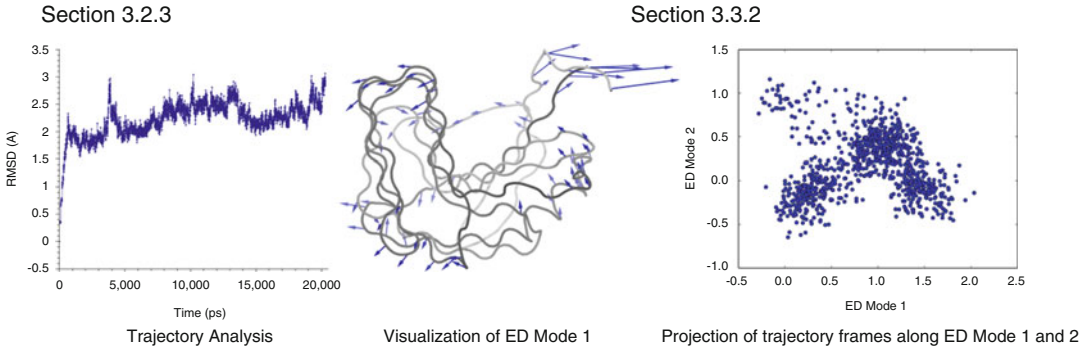


Fig. 1 Essential modes analysis for the example of the xylanase structure from *B. circulans* (PDB code 3LB9)

PROTEIN_trajectory.dcd file is required for further analysis since ProDy exclusively works with this type of trajectory file (see below). The first frame (*PROTEIN*_1st_frame.pdb) can be also extracted from the trajectory and used as a reference structure.

3.3 Essential Dynamics Analysis of MD Trajectories Using ProDy

Starting from a reference structure *PROTEIN_1st_frame.pdb* and a trajectory file *PROTEIN_trajectory.dcd*, Essential Dynamics Analysis can be carried out with the available tools in ProDy. [11] ProDy requires installation of other software; instructions for download and documentation can be found here: http://prody.csb.pitt.edu/. The following example is executed using the IPython interactive command shell [14].

3.3.1 Calculation of Essential Modes

1. Import of all related content from ProDy:

```
$from prody import *
$from pylab import *
```

2. Defining the reference structure:

```
$PROTEIN_EDA = parsePDB('PROTEIN_1st_frame.pdb')
```

3. Defining the trajectory file:

```
$trajectory_EDA= parseDCD('PROTEIN_trajectory.dcd')
```

4. Restrict the analysis only to the subset of C_α atoms of the reference structure:

```
$trajectory_EDA.setAtoms(PROTEIN_EDA.calpha)
```

5. Defining the atom reference coordinates:

```
$trajectory_EDA.setCoords(PROTEIN_EDA)
```

6. Superposition of all trajectory frames onto the reference structure:

```
$trajectory_EDA.superpose()
```

7. Defining the class for Essential Dynamics Analysis:

```
$eda=EDA('PROTEIN_EDA')
```

8. Construction of the $3N \times 3N$ covariance matrix of atomic coordinates over f trajectory frames, where N in this example is the number of C_α atoms. Each frame corresponds to snapshot conformations contained in the *.*dcd* file, being superposed to the reference coordinate set, as in **step 7**):

```
$eda.buildCovariance(trajectory_EDA)
```

9. Calculation of the n (e.g., $n = 3$) essential modes by diagonalization of the covariance matrix to obtain eigenvectors with nonzero eigenvalues:

```
$eda.calcModes(n)
```

10. Saving the model and an *.*nmd* file containing n essential modes for visualization in VMD with the normal mode wizard plugin (NMWiz):

```
$saveModel(eda)
$writeNMD('PROTEIN_EDA.nmd', eda[:n], PROTEIN_EDA.calpha)
```

3.3.2 Essential Modes Analysis

The obtained essential modes can be additionally analyzed with available ProDy built-in functions (Fig. 1). The following examples can be used to quantitatively describe them.

1. Projection of the trajectory frames onto the first $n<3$ essential modes. The projection of the trajectory frames onto the first

essential modes provides a description of the conformational space explored during simulation time:

```
$traj_frames=Trajectory('PROTEIN_trajectory.dcd')
$traj_frames.link(PROTEIN_EDA)
$traj_frames.setCoords(PROTEIN_EDA)
$traj_frames.setAtoms(PROTEIN_EDA .calpha)
$showProjection(traj_frames, eda[:n])
```

2. Fractional variance of the first n modes. Fractional variance corresponds to the ratio between the variance obtained along an essential mode to the trace of the covariance matrix:

```
$calcFractVariance(eda[:n])
```

3. Collectivity degree k of essential mode n [15]. The collectivity degree is used as a measure of the number of atoms affected by a given essential mode. It ranges from $k = 1$ for global translations of the protein to $k = N^{-1}$ if only one C_α atom is affected:

```
$calcCollectivity(eda[n])
```

3.4 Comparative Analysis of Essential Modes

The previous section concerns to the single description of essential modes from a designed protein. However, of particular importance is to compare them with other sets of modes, as the ones obtained from a MD trajectory of the native protein or from different trajectory files of the same MD simulation varying in length or start time (Subheading 3.4.1). It can also be relevant to check for the correspondence between the conformational space described by the first essential modes and the modes describing the structural fluctuations observed experimentally from an ensemble of native crystallographic or NMR structures (Subheading 3.4.2). This provides insights on the ability of the designed protein to effectively reproduce the dynamical properties observed experimentally. One can also check if coarse-grained Elastic Network Models, such as the Anisotropic Network Model (ANM) implemented in ProDy, are able to capture the conformational space explored by either the native or designed protein during the simulation trajectory (Subheading 3.4.3, Fig. 2).

3.4.1 Comparison Between Essential Modes from Two Distinct Trajectories

This step requires prior calculation of essential modes from a second reference structure *PROTEIN2_1st_frame.pdb* and second trajectory file *PROTEIN2_trajectory.dcd*, as described in Subheading 3.3.1. A requirement for the second reference structure *PROTEIN2* is to contain the same number of N atoms as the reference

structure of *PROTEIN*. Therefore, the second reference structure can be the same as *PROTEIN_1st_frame.pdb*, if the trajectory to be analyzed is from the same simulation run and starting frame but with different lengths. Therefore, **steps 1–10** of Subheading 3.3.1 might also be considered here for the calculation of the Essential Modes of the second trajectory. At the end, two distinct sets of modes are obtained, *eda* and *eda2*, that can be used in comparison functions built-in on ProDy. The following is not an exhaustive list of comparison functions:

1. Calculation of the overlap, or correlation cosine, between *eda n* mode and *eda2 m* mode, as given by the dot product of the respective eigenvectors after normalization. This value is equal to 1 if modes *n* and *m* are identical:

```
$calcOverlap(eda[:n], eda2[:m])
```

A normalized table with overlap between *eda* modes $<n$ and *eda2* modes $<m$ can also be obtained:

```
$showOverlapTable(eda[:n], eda2[:m])
```

2. Calculation of the subspace overlap between *eda* modes $<n$ and *eda2* modes $<m$, as given by the Root Mean Square Inner Product value [16]:

```
$calcSubspaceOverlap(eda[:n], eda2[:m])
```

3. Projection of the *trajectory* (or *trajectory2*) onto the subspace defined by *eda* mode *n* and *eda2* mode *m*. Dispersion of the frames along the diagonal indicates close correspondence between mode *n* and *m*:

```
$showCrossProjection(traj_frames, eda[n], eda2[m])
```

The corresponding correlation coefficient can also be calculated:

```
$eda_eda2_corr=calcCrossProjection(traj_frames, eda[n], eda2[m])
$print(np.corrcoef(eda_eda2_corr))
```

4. Comparison of normalized square fluctuations of *eda* mode *n* and *eda2* mode *m*. C_α atoms are sorted by index value:

```
$showNormedSqFlucts(eda[n], eda2[m])
$legend()
```

The corresponding plot of scaled square fluctuations can also be obtained. Legend contains the respective scaling factor:

```
$showScaledSqFlucts(eda[n], eda2[m])
$legend()
```

3.4.2 Comparison Between Essential Modes and Principal Components from an Ensemble of Structures

This step requires an ensemble of protein structures, corresponding to either a set of native *i* crystallographic or NMR-derived structures. Principal Component Analysis (PCA) is employed to extract the modes of structural variation occurring within the structural set, which can then be used to compare directly with the essential modes using the built-in functions implemented in ProDy. It should be noted that reliable results can only be obtained for a sufficiently large number of *i* and with significant similarity with the chosen reference structure.

1. Defining the set of structures to be analyzed. In this example, each structure is identified by its corresponding PDB Identifier code (PDB_ID). ProDy can download directly the respective ***.pdb* files from the PDB database, or read them from a given working directory, as follows:

```
$structures=['PDB_ID1', 'PDB_ID2', ..., 'PDB_IDi']
$pdb_structures=fetchPDB(*structures, compressed=False)
```

2. Defining the class of conformational ensemble:

```
$ensemble_PCA=PDBEnsemble('PROTEIN')
```

3. Defining the reference structure and chain:

```
$reference=parsePDB('reference.pdb', subset='calpha')
$reference_chain=reference.getHierView().getChain('X')
$ensemble_PCA.setAtoms(reference)
$ensemble_PCA.setCoords(reference)
```

Note: 'X' is the chain identifier of the reference structure

4. Iterative superpositioning of the ensemble. All structures are first superposed to the *reference* structure and then iteratively superposed to the mean coordinates until convergence to eliminate rigid-body rotational and translational differences:

```
$for pdb_structure in pdb_structures:
$structure_pca=parsePDB(pdb_structure, subset='calpha')
$mappings=mapOntoChain(structure_pca,reference_chain)
$atommap=mappings[0][0]
$ensemble_PCA.addCoordset(atommap,weights=atommap.getFlags('mapped'))
$ensemble_PCA.iterpose()
```

5. Defining the class for Principal Component Analysis:

```
$pca=PCA('PROTEIN')
```

6. Construction of the $3N \times 3N$ covariance matrix of atomic coordinates over i structures, where N is the number of C_α atoms:

```
$pca.buildCovariance(ensemble_PCA)
```

7. Calculation of the n principal components by diagonalization of the covariance matrix to obtain eigenvectors with nonzero eigenvalues:

```
$pca.calcModes(n)
```

8. Saving the model and a **.nmd* file containing n principal components for visualization in VMD with NMWiz:

```
$saveModel(pca)
$writeNMD('PROTEIN_PCA.nmd', pca[:n], ensemble_PCA)
```

9. The set of *pca* modes is ready to be further analyzed as in Subheading 3.3.2 and compared with *eda* modes as in Subheading 3.4.1. Both sets of modes can also be visualized in VMD by loading the respective *PROTEIN_EDA.nmd* and *PROTEIN_PCA.nmd* files with the NMWiz.

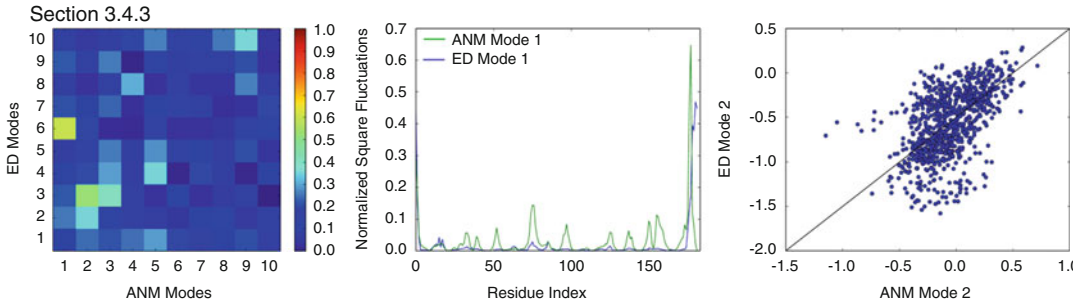


Fig. 2 ANM normal mode analysis for the example of the xylanase structure from *B. circulans* (PDB code 3LB9)

3.4.3 Comparison Between Essential Modes and Modes Derived from the Anisotropic Network Model

In this step, the calculation of ANM of the designed protein is performed for comparison with the essential modes derived from a trajectory of the same protein (Fig. 2), as follows:

1. Defining the structure for ANM calculation. The model considers only the C_α atoms:

```
$PROTEIN_anm= parsePDB('PROTEIN.pdb', subset='calpha')
```

2. Defining the class for ANM analysis:

```
$anm=ANM('PROTEIN')
```

3. Construction of the Hessian matrix of atomic coordinates.

```
$anm.buildHessian(PROTEIN_anm)
```

4. Calculation of *n* normal modes by diagonalization of the Hessian matrix. Only modes with nonzero eigenvalues are obtained:

```
$anm.calcModes()
```

5. Saving the model and a ***.nmd* file containing *n* normal modes for visualization in VMD with NMWiz:

```
$saveModel(anm)
$writeNMD('PROTEIN_ANM.nmd', anm[:n], PROTEIN_anm)
```

6. In the same routine for *pca* modes, *anm* normal modes can be further analyzed as in Subheading 3.3.2 and compared with *eda* modes as in Subheading 3.4.1. Both sets of modes can also be visualized in VMD by loading the respective *PROTEIN_EDA.nmd* and *PROTEIN_ANM.nmd* files with the NMWiz.

4 Notes

Input parameter files used for the MD simulations (command lines starting with ";" are comments):

1st Minimization Step: input template_em1.mdp file

```
title = Energy Minimization; Title of run
; Parameters describing what to do, when to stop and what to save
integrator = steep; Algorithm (steep = steepest descent minimization)
emtol = 1000.0; Stop minimization when the maximum force < 10.0 kJ*mol-1*nm-1
emstep = 0.01; Energy step size in nm
nsteps = 2000; Maximum number of (minimization) steps to perform
energygrps = system; Which energy group(s) to write to disk
; Parameters describing how to find the neighbors of each atom and how to calculate the interactions
nstlist = 10; Frequency to update the neighbor list and long range forces
ns_type = grid; Method to determine neighbor list (simple, grid)
rlist = 1.0; Cut-off for making neighbor list (short range forces)
coulombtype = PME; Treatment of long range electrostatic interactions
rcoulomb = 1.0; long range electrostatic cut-off
vdwtype = cut-off; Treatment of van der Walls interactions
rvdw = 1.4; long range Van der Waals cut-off
pbc = xyz; Periodic Boundary Conditions (yes/no)
fourierspacing = 0.12
fourier_nx = 0
fourier_ny = 0
fourier_nz = 0
pme_order = 4
ewald_rtol = 1e-5
optimize_fft = yes
tcoupl = no
pcoupl = no
gen_vel = no
```

2nd Minimization Step: input template_em2.mdp file

```
title = Energy Minimization; Title of run
; Parameters describing what to do, when to stop, and what to save
```

```
integrator = cg; Algorithm (steep = steepest descent minimization)
emtol = 400.0; Stop minimization when the maximum force < 10.0 kJ*mol-1*nm-1
emstep = 0.01; Energy step size in nm
nsteps = 1000; Maximum number of (minimization) steps to perform
energygrps = system; Which energy group(s) to write to disk
; Parameters describing how to find the neighbors of each atom and how to calculate the
 interactions
nstlist = 10; Frequency to update the neighbor list and long range forces
ns_type = grid; Method to determine neighbor list (simple, grid)
rlist = 1.0; Cut-off for making neighbor list (short range forces)
coulombtype = PME; Treatment of long range electrostatic interactions
rcoulomb = 1.0; long range electrostatic cut-off
vdwtype = cut-off; Treatment of van der Walls interactions
rvdw = 1.4; long range Van der Waals cut-off
pbc = xyz; Periodic Boundary Conditions (yes/no)
fourierspacing = 0.12
fourier_nx = 0
fourier_ny = 0
fourier_nz = 0
pme_order = 4
ewald_rtol = 1e-5
optimize_fft = yes
tcoupl = no
pcoupl = no
gen_vel = no
```

1st – 3rd Equilibration Step: input template_eq*.mdp file

```
 title = Protein-ligand complex NPT equilibration phase
define = -DPOSRES; position restrain the protein and ligand
; Run parameters
integrator = md; leap-frog integrator
nsteps = 50000; 0.002 * 50000 = 100 ps
dt = 0.002; 2 fs
; Output control
nstxout = 1000; save coordinates every 2 ps
nstvout = 1000; save velocities every 2 ps
nstenergy = 2000; save energies every 4 ps
nstlog = 1000; update log file every 2 ps
energygrps = Protein Non-protein
; Bond parameters
continuation = yes; first dynamics run
constraint_algorithm = lincs; holonomic constraints
constraints = all-bonds; all bonds (even heavy atom-H bonds) constrained
lincs_iter = 1; accuracy of LINCS
lincs_order = 4; also related to accuracy
; Neighborsearching
```

```
ns_type = grid; search neighboring grid cells
nstlist = 10; 20 fs
rlist = 1.0; short-range neighborlist cutoff (in nm)
rcoulomb = 1.0; short-range electrostatic cutoff (in nm)
vdwtype = cut-off
rvdw = 1.4; short-range van der Waals cutoff (in nm)
; Electrostatics
coulombtype = PME; Particle Mesh Ewald for long-range electrostatics
pme_order = 4; cubic interpolation
fourierspacing = 0.16; grid spacing for FFT
ewald_rtol = 1e-5
optimize_fft = yes
; Berendsen temperature coupling is on
tcoupl = V-rescale; modified Berendsen thermostat
tc-grps = Protein non-protein; two coupling groups - more accurate
tau_t = 0.1 0.1; time constant, in ps
ref_t = 300 300; reference temperature, one for each group, in K
; Pressure coupling is on
pcoupl = Berendsen; pressure coupling is on for NPT
pcoupltype = isotropic; uniform scaling of box vectors
tau_p = 0.6; time constant, in ps
ref_p = 1.0; reference pressure, in bar
compressibility = 4.5e-5; isothermal compressibility of water, bar^-1
; Periodic boundary conditions
pbc = xyz; 3-D PBC
; Dispersion correction
DispCorr = EnerPres; account for cut-off vdW scheme
; Velocity generation
gen_vel = yes; assign velocities from Maxwell distribution
gen_temp = 300; temperature for Maxwell distribution
gen_seed = -1; generate a random seed
```

Production Step: input template_md1.mdp file

```
title = Protein-ligand complex NPT nonconstraint explicit solvent md simulation
; Run parameters
integrator = md; leap-frog integrator
nsteps = 10000000; 0.002 * 10000000 = 20000 ps (20 ns)
dt = 0.002; 2 fs
; Output control
nstcomm = 1
nstxout = 1000; save coordinates in .trr output every 2 ps
nstvout = 1000; save velocities in .trr output every 2 ps
nstenergy = 2000; save energies every 4 ps
nstlog = 1000; update log file every 2 ps
nstfout = 0; do not collect forces
energygrps = Protein Non-protein
```

```
; Bond parameters
continuation = yes; first dynamics run
constraint_algorithm = lincs; holonomic constraints
constraints = hbonds
lincs_iter = 1; accuracy of LINCS
lincs_order = 4; also related to accuracy
; Neighborsearching
ns_type = grid; search neighboring grid cells
nstlist = 1.0; 2 fs
rlist = 1.0; short-range neighborlist cutoff (in nm)
rcoulomb = 1.0; short-range electrostatic cutoff (in nm)
vdwtype = cut-off
rvdw = 1.4; short-range van der Waals cutoff (in nm)
; Electrostatics
coulombtype = PME; Particle Mesh Ewald for long-range electrostatics
pme_order = 4; cubic interpolation
fourierspacing = 0.12; grid spacing for FFT
fourier_nx = 0
fourier_ny = 0
fourier_nz = 0
ewald_rtol = 1e-5
optimize_fft = yes
; Berendsen temperature coupling is on
tcoupl = V-rescale; modified Berendsen thermostat
tc-grps = Protein non-protein; two coupling groups - more accurate
tau_t = 0.1 0.1; time constant, in ps
ref_t = 300 300; reference temperature, one for each group, in K
; Pressure coupling is on
pcoupl = Berendsen; pressure coupling is on for NPT
pcoupltype = isotropic; uniform scaling of box vectors
tau_p = 0.6; time constant, in ps
ref_p = 1.0; reference pressure, in bar
compressibility = 4.5e-5; isothermal compressibility of water, bar^-1
; Periodic boundary conditions
pbc = xyz; 3-D PBC
; Dispersion correction
DispCorr = EnerPres; account for cut-off vdW scheme
; Velocity generation
gen_vel = no; assign velocities from Maxwell distribution
gen_temp = 300; temperature for Maxwell distribution
```

Acknowledgments

The authors thank the financial support from Fundação para a Ciência e a Tecnologia, Portugal, through contracts SFRH/BD/90644/2012 (H.F.C.), SFRH/BPD/69163/2010 (R.J.F.B.),

and ERA-IB-2/0001/2013. This work was supported by the Unidade de Ciências Biomoleculares Aplicadas-UCIBIO, which is financed by national funds from FCT/MEC (UID/Multi/04378/2013) and co-financed by the ERDF under the PT2020 Partnership Agreement (POCI-01-0145-FEDER-007728). The authors would like to thank also the support from Centre National de la Recherche Scientifique (CNRS), France and FCT, Portugal, through the International Program of Scientific Cooperation (PROJECT PICS-147340).

References

1. Privett HK et al (2012) Iterative approach to computational enzyme design. Proc Natl Acad Sci U S A 109:3790–3795
2. Wolfenden R, Snider MJ (2001) The depth of chemical time and the power of enzymes as catalysts. Acc Chem Res 34:938–945
3. Tantillo DJ, Chen J, Houk KN (1998) Theozymes and compuzymes : biological catalysis theoretical models for biological catalysis. Curr Opin Chem Biol 2:743–750
4. Branco RJF, Graber M, Denis V, Pleiss J (2009) Molecular mechanism of the hydration of Candida antarctica lipase B in the gas phase: water adsorption isotherms and molecular dynamics simulations. Chembiochem 10:2913–2919
5. Röthlisberger D et al (2008) Kemp elimination catalysts by computational enzyme design. Nature 453:190–195
6. Richter F, Leaver-Fay A, Khare SSD, Bjelic S, Baker D (2011) De novo enzyme design using Rosetta3. PLoS One 6:e19230
7. Reitinger S et al (2010) Circular permutation of Bacillus circulans xylanase: a kinetic and structural study. Biochemistry 49:2464–2474
8. Bernstein FC et al (1977) The Protein Data Bank. A computer-based archival file for macromolecular structures. Eur J Biochem 80:319–324
9. Olsson MHM, SØndergaard CR, Rostkowski M, Jensen JH (2011) PROPKA3: consistent treatment of internal and surface residues in empirical p K a predictions. J Chem Theory Comput 7:525–537
10. Oostenbrink C, Villa A, Mark AE, van Gunsteren WF (2004) A biomolecular force field based on the free enthalpy of hydration and solvation: the GROMOS force-field parameter sets 53A5 and 53A6. J Comput Chem 25:1656–76
11. Bakan A, Meireles LM, Bahar I (2011) ProDy: protein dynamics inferred from theory and experiments. Bioinformatics 27:1575–7
12. Skjærven L, Yao XQ, Scarabelli G, Grant JB (2014) Integrating protein structural dynamics and evolutionary analysis with Bio3D. BMC Bioinformatics 15:339
13. Humphrey W, Dalke A, Schulten K (1996) VMD: visual molecular dynamics. J Mol Graph 14(33–8):27–8
14. Pérez F, Granger BE (2007) IPython: a system for interactive scientific computing. Comput Sci Eng 9:21–29
15. Brüschweiler R (1995) Collective protein dynamics and nuclear spin relaxation. J Chem Phys 102:3396
16. Amadei A, Ceruso MA, Di Nola A (1999) On the convergence of the conformational coordinates basis set obtained by the essential dynamics analysis of proteins' molecular dynamics simulations. Proteins 36:419–424
17. Hilvert, D. (2013). Design of protein catalysts. Annual Review of Biochemistry. 10.1146/annurev-biochem-072611-101825, 82, 447–70. doi:10.1146/annurev-biochem-072611-101825
18. Samish, I., Gu, J., & Klein, M. L. (2009). Protein Motion: Simulation. In P. E. Bourne & J. Gu (Eds.), *Structural Bioinformatics* (2nd ed., pp. 909–938). Wiley-Blackwell

Chapter 9

Applications of Normal Mode Analysis Methods in Computational Protein Design

Vincent Frappier, Matthieu Chartier, and Rafael Najmanovich

Abstract

Recent advances in coarse-grained normal mode analysis methods make possible the large-scale prediction of the effect of mutations on protein stability and dynamics as well as the generation of biologically relevant conformational ensembles. Given the interplay between flexibility and enzymatic activity, the combined analysis of stability and dynamics using the Elastic Network Contact Model (ENCoM) method has ample applications in protein engineering in industrial and medical applications such as in computational antibody design. Here, we present a detailed tutorial on how to perform such calculations using ENCoM.

Key words Normal mode analysis, Protein stability, Protein dynamics, Mutations, Vibrational entropy, Protein engineering

1 Introduction

Protein engineering aims at modulating the physico-chemical and biological properties of proteins through chemical modifications for industrial and medical applications. Such modifications include derivatizing surface residues and the introduction of mutations. Industrial applications often require mutations that confer increased efficiency in conditions drastically different than physiological as well as improved resistance to denaturation [1]. In a visionary article in 1983, Kevin Ulmer proposed that the integration of experimental approaches in protein chemistry, X-ray crystallography, and computer modeling held the key to understand and engineer protein structure and function [2]. Over 30 years later, much progress has been made but we are far from truly understanding protein function and structure to the point where we can engineer de novo functions. Traditionally, protein engineering involved structure-guided design through site-directed mutagenesis. While this approach is still used [3, 4], new methodologies such as directed evolution are commonly used today. Directed evolution is an experimental approach mimicking biological evolution where

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_9,

a large number of random mutants are produced and evolutionary pressure is applied in which successive rounds of selection are used to favor the emergence of desired phenotypes [5]. In that respect and depending on the goal, promiscuity in terms of binding or catalysis often simplifies the engineering task [6]. Otherwise, directed evolution can be sensitive to local minima of the fitness landscape [7, 8]. The late physicist Richard Feynman stated "what I cannot create, I do not understand." Directed evolution shows that it is possible to create new proteins without full understanding. However, in the spirit of Ulmer, the true potential of protein engineering will be achieved once we understand enough of the principles underlying protein structure and function to perform ab initio protein design.

Computational approaches have been used to identify mutations that change protein affinity [9], function [10], and stability [11]. However, most computational methods that focus on the impact of mutations on protein stability are biased toward predicting destabilizing mutations. This bias comes at times as an artifact of machine learning, but it can also be caused by the inherent difficulty of modeling stabilizing mutations. Therefore, most computational methods currently available fail to correctly predict stabilizing mutations [12, 13]. Another important point to consider is that changes in thermodynamic stability may have a detrimental effect on enzymatic activity [14–19]. A striking example comes from the comparison of mesophilic enzymes with their more stable thermophilic counterparts that exhibit lower enzyme efficiency at room temperatures [20]. This loss of efficiency is often associated with a rigidification of the structure [21, 22]. More generally, dynamics affects molecular recognition [9, 23–26] and catalytic rates [27, 28]. It is especially true for antibodies [29] where a rigidification of the complementarity determining region (CDR) is observed during the maturation process [30] and crucial to obtain high affinity specific molecules [31]. Allosteric mutations that improve binding affinity [32] in therapeutic antibodies highlight the importance of assessing the impact of mutations on protein dynamics. Finally, describing a protein as the conformational ensemble rather than a single structure has been shown to improve the prediction of the effect of mutations [33, 34] and improved the outcome of protein design protocols [35].

The evaluation of dynamic properties of proteins in a high-throughput context is not a trivial task. Experimental procedures (NMR or crystallographic b-factors) can be time-consuming and despite tremendous advances in molecular dynamic simulations, the ability to assess the effect of a mutation on dynamic properties of proteins is still computationally demanding, particularly for the long timescales associated with protein function [36]. Thus, evaluating several hundred mutants would seem unrealistic without specialized hardware. Normal mode analysis (NMA) provides an alternative.

It is a computational approach that predicts vibrational frequencies and movements of a system around an equilibrium state using a harmonic potential. The fundamentals of NMA have been extensively reviewed [37, 38] and classically is applied on all atoms of the structure with a molecular dynamics force field after initial minimization. Pioneering work by Tirion [39] showed that it is possible to reproduce the slow dynamics of proteins with a single-parameter potential by considering the structure as already in its equilibrium conformation and building a mass-spring system, removing the requirement for minimization. Tama et al. [40] showed that it is possible to replace all atoms of a residue by a single mass generally centered at the position of the alpha carbon, drastically reducing computational time. The speed of such coarse-grained NMA methods made possible their use in many applications to explore conformational space in small molecule docking [41, 42], to predict conformational changes [43] and in structural refinement [44, 45]. However, most coarse-grained methods do not account for the nature of amino acids by using spring constants that are independent of residue type. We recently introduced a coarse-grained NMA method called ENCoM [46], which uses a potential based on STeM [47] considering bond stretching, angle bending, dihedral rotation, and long-range interactions. Crucially, ENCoM adds an additional factor to the long-range interactions using the surface area in contact and the type of heavy atoms in contact. Thus, unlike other coarse-grained NMA methods, ENCoM calculations are affected by the specific amino acid nature of the protein in addition to its structure. Compared to the Anisotropic Network Model (ANM), one of the most used coarse-grained NMA methods [48], ENCoM shows an increased predictive power for conformational change between crystal structures of bound and unbound enzymes with an average increase in squared overlap of 28 % for 117 coupled movements and 60 % for 236 cases of coupled loop movements.

With ENCoM, we also introduced a novel application for coarse-grained NMA methods in the prediction of the effect of mutations on protein stability and dynamic properties. Predicted vibrational entropy differences (ΔS_{vib}) upon mutation were analyzed for 303 manually curated mutations [49] and compared to several existing methods, notably FoldX3.0 (beta 3.0) [50], Rosetta [51], DMutant [52], and PoPMusic [49]. Although not the overall best predictive method, ENCoM proved to be the most self-consistent and least biased. ENCoM and DMutant gave the best predictive power on the subset of 45 stabilizing mutations versus other methods that predicted as good or worse than a random model. Classic coarse-grained NMA models predicted every mutation as neutral and did not have any predictive power. The combination of ENCoM with enthalpy-based methods such as Rosetta and FoldX was synergistically beneficial [53]. As a proof of concept for the prediction of the effect of mutations on function,

ENCoM predicted the effect of the G121V mutation on *E. coli* DHFR consistent with S^2 differences NMR results [54]. Despite having a modest effect on protein stability (0.77 kcal/mol [55]) and being 15 Å away from the binding site, this mutation disrupts enzyme efficiency by 200-fold through allosteric effects. More recently, ENCoM was used to show that thermophile proteins are on average more rigid than their mesophile counterpart and used ΔS_{vib} to guide the selection of mutations observed between such proteins with potential uses in protein engineering [22].

In the following sections, we demonstrate how to use ENCoM to predict the effect of mutations on thermal stability and dynamics as well as to generate conformational ensembles (Fig. 1). The ability to perform large-scale combined predictions of the effect of mutations on stability and dynamics offers great possibilities in protein engineering. Likewise, the generation of biologically realistic conformational ensembles has ample applications in protein engineering and beyond.

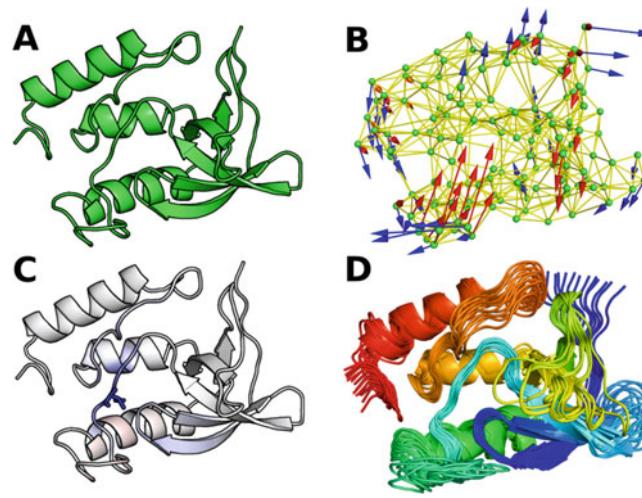


Fig. 1 Uses of ENCoM in protein engineering. The wild-type nuclease from Staphylococcus aureus (1EY0) used in the text is shown in (**a**). The protein structure is represented as an elastic network model using ENCoM algorithm (**b**), where amino acids are represented by masses (*green spheres*) and interactions by springs (*yellow sticks*). The Eigenvectors representing the seventh and tenth modes are shown in *red* and *blue* respectively. The mutation T41I (shown as stick in **c**) increases the thermal stability and rigidifies the protein in the regions identified in *blue* (**c**). A conformational ensemble of 11 conformations of the wild-type nuclease generated using the seventh and tenth modes are shown in (**d)**

2 Materials

For this tutorial it will be necessary to have some basic knowledge of command line environments and to install software (*see* **Note 1**). At the moment ENCoM does not work under the Windows operating system. Thus, for the tutorial below it is necessary to use a Unix-based operating system (Linux or Mac OS). Please make sure your system has up-to-date versions of Python and Perl.

The ENCoM Source code can be found at http://bcb.med.usherbrooke.ca/encom or through GitHub at https://github.com/NRGlab/ENCoM. Code can be compiled by the following instructions in the Readme file (*see* **Note 2**). ENCoM is used for the prediction of the effect of mutations and to generate conformational ensembles. Precompiled executables of FoldX3 can be found at: http://foldx.crg.es (*see* **Note 3**). FoldX3 is used exclusively for the prediction of the effect of mutations. Instructions to download and install Modeller can be found at https://salilab.org/modeller/download_installation.html. PyMOL is used for molecular visualizations. Instructions for installation on different operating systems can be found at http://www.pymolwiki.org/index.php/Category:Installation. Alternatively, the PyMOL source code can be found at: http://sourceforge.net/projects/pymol (*see* **Note 4**). All scripts required for the protocols used below can be found at http://bcb.med.usherbrooke.ca/encom.

3 Methods

The evaluation of the effect of mutations on protein thermodynamic stability is achieved by a linear combination of the predictions of ENCoM and FoldX. The prediction of the effect of mutations on protein dynamic on the other hand uses ENCoM exclusively. ENCoM is also used to generate ensembles of realistic protein conformations. The following protocols can be carried out in standard computers and do not require any specialized hardware. Execution times can vary from a few minutes to a few hours depending on the type of hardware used, the size of the protein, and the number of mutations to evaluate or conformations to generate. The entire protocol can also be automatically executed through the ENCoM Server [53] at http://bcb.med.usherbrooke.ca/encom. The advantage of running oneself the protocols is to overcome restrictions that are in place in the ENCoM Server such as the possibility to model and predict the effect of double (or more) mutants, the manner in which conformations are modeled using Modeller, and to explore combinations of modes that generate larger conformational ensembles than allowed in the webserver. Results obtained through the ENCoM Server interface can

serve to validate results obtained using the protocols below as the user learns how to use ENCoM.

We will be using the structure of the *Staphylococcus aureus* Thermonuclease (PDB ID 1EY0) as an example. However, any protein structure or model can be used (*see* **Note 5**).

During the protocol, we will be using software that can be installed in different directories depending on the computer. The FoldX3 installation folder will be referred to as *FoldX/*, ENCoM installation folder will be referred to as *ENCoM/*, and the perl and python scripts will be referred to as *script/*. The user should make sure to recognize what are the appropriate directories in their installation and replace the names accordingly. Text in italic following the > symbols represent command lines that are to be entered in a terminal.

3.1 Preparing Working Environment

In order to run ENCoM, is it better to create a work directory within which we will place the PDB formatted file containing the coordinates of the protein and prepare it:

1. Create a folder named *work* in which you will be working and change the working directory:

   ```
   > mkdir work
   > cd work
   ```

2. Download the 1EY0 structure from the PDB website using this address http://www.rcsb.org/pdb/files/1EY0.pdb and name it *1ey0_nc.pdb*; alternatively, use the command line below:

   ```
   > curl http://www.rcsb.org/pdb/files/1EY0.pdb > 1ey0_nc.pdb
   ```

3. Clean the PDB file by removing heteroatoms, water molecules, alternative conformations, and hydrogen atoms, changing negative residue numbers or residues with non-numeric characters, removing multiple models and adding a chain identifier Z if none is present using this command (*see* **Note 6**). The cleaned structure is now called *1ey0.pdb*.

   ```
   > perl script/clean_pdb.pl 1ey0_nc.pdb 1ey0.pdb
   ```

3.2 FoldX3 Thermal Stability Predictions

Thermal stability predictions involve a linear combination of FoldX3 predictions and ENCoM. ENCoM. As noted above, users must download FoldX3 and install it first. Once this is done follow the steps below:

1. In order to preprocess the protein structure we start with the following command

   ```
   > echo 1ey0.pdb > list.txt
   ```

2. Copy the *rotabase.txt* file found within the FoldX3 software into the working directory:

```
> cp FoldX/rotabase.txt ./
```

3. Launch FoldX3 repair function. This will generate a file named *RepairPDB_1ey0.pdb*.

```
> FoldX/foldx3b6 -runfile ./script/repair.txt
```

4. Write this filename in a list using

```
> echo RepairPDB_1ey0.pdb > list.txt
```

5. Open the file named *individual_list.txt* using any plain text editor (in the following command line we use nano) and write mutations that are to be evaluated using the following nomenclature: One letter code wild-type residue, chain, position in the structure sequence, and one letter code mutated residues, followed by a semicolon. For example, to mutate threonine 41 to an isoleucine in the 1EY0 structure, write *TA41I*;. For this protocol, please write in the *individual_list.txt* file on different lines the two following mutants: *TA41I;* and *DA21K;* (*see* **Note 7**).

```
> nano individual_list.txt
```

6. Launch the FoldX3 mutation function. The file *Dif_BuildModel_RepairPDB_1ey0.fxout* created in the working directory will have the difference in folding energy between WT and mutated forms (*see* **Note 8**).

```
> FoldX/foldx3b6 -runfile script/run.txt.
```

3.3 Effect of Mutations on Protein Stability and Dynamics

The ENCoM predictions can then be calculated as follows:

1. Generate the structure of the T41I and D21K mutants in chain A with the following command lines, where 1ey0 represents the filename, 41 or 21 the positions to mutate, ILE or LYS the new residues at these positions in chain A. The resulting modeled mutant structures will be in files *1ey0ILE41A.pdb* and *1ey0LYS21A.pdb*. In the command line below, the last two arguments represent the input PDB file containing the wild-type coordinates and the filename for the mutant coordinates respectively.

```
> python script/mutate_model.py 1ey0 41 ILE A 1ey0.pdb
1ey0ILE41A.pdb
> python script/mutate_model.py 1ey0 21 LYS A 1ey0.pdb
1ey0LYS21A.pdb
```

2. Calculate the normal modes and mode amplitudes for the wild-type and mutant structures generated in the previous step using the following command. The *.cov* files represent the entropy for each residue and the *.eigen* files contain the eigenvalues (mode frequencies) and eigenvectors (normal modes) of the different vibrational modes. These files will be used to compare dynamics between structures (*see* **Note 9**).

```
> ./ENCoM/bin/build_encom -i 1ey0.pdb -cov wt.cov -o
wt.eigen
> ./ENCoM/bin/build_encom -i 1ey0ILE41A.pdb -cov
TA41I.cov -o TA41I.eigen
> ./ENCoM/bin/build_encom -i 1ey0LYS21A.pdb -cov
DA21K.cov -o DA21K.eigen
```

3. The following command will use the files produced above to calculate the differences in dynamics between each mutant and the wild type, as well as the predicted $\Delta\Delta G$ for each mutation. The predicted $\Delta\Delta G$ is a linear combination of ENCoM and FoldX calculated earlier (*see* **Note 10**). The order of *.cov* files for the *-mutl* argument must be the same that the one in *individual_list.txt*.

```
> perl script/compare_cov.pl -FoldX Dif_BuildModel_-
RepairPDB_1ey0.fxout -wt wt.cov -mutl TA41I.cov DA21K.
cov.
```

4. The command script will generate a PyMOL session script called *Diff.pml* that colors every amino acid in function of ΔS for residue in each mutant, where blue represents a rigidification of the structure and red a gain in flexibility (*see* **Note 11**). It can be viewed using:

```
> pymol Diff.pml
```

3.4 Generation of Conformational Ensembles

In addition to the prediction of the effect of mutations on stability and dynamics, ENCoM can be used to generate conformational ensembles:

1. The following script generates multiple conformations using ENCoM. In the case below, we are using the wild type and use the eigenvectors previously calculated in Subheading 3.3, **step 2** (file *wt.eigen*). The same could be done for a mutant, using the appropriate mutant structure and calculated eigenvectors. The file *all_conformations.pdb* contains all the exhaustively generated models using the 10th and the 12th slowest vibrational modes (parameter *–ml*) with a maximum RMSD distortion of 2 Å (parameter *–md)* and a minimum RMSD distortion of 1 Å (parameter *–step*) per mode. Remember that the first six modes represent rotations and translations; thus, the smallest value for any argument passed via *–ml* should be 7, representing the slowest, most global mode of movement.

```
> ENCoM/bin/build_grid_rmsd -i 1ey0.pdb -ieig wt.eigen
-md 2 -step 1 -p all_conformations.pdb -ml 10 12
```

2. Each individual mode can be viewed using the motion function. For example, the mode 10 can be given by

```
> ENCoM/bin/motion -i 1ey0.pdb -m 10 -ieig wt.eigen -
p motion_10.pdb
```

3. Cartesian space NMA methods such as ENCoM generated conformations that are linear combinations of movements (translations of atomic coordinates) along different modes. Thus, the structures generated do not respect bond angles and distances. Conformations represent distorted physically unrealistic structures. Modeller is used to rebuild physically realistic structures using each distorted NMA structure as a template. The rebuilt model will be found in the folder called *models*. This is done with the command below.

```
> perl script/rebuild.pl -i all_conformations.pdb -
script script/rebuild.py
```

4 Notes

1. All software employed in the protocols are free at least for nonprofit users. ENCoM is free for everyone and distributed under the GNU General Public License.
2. Users need to have the GNU GSL library installed, more information can be found at http://www.gnu.org/software/gsl/.
3. FoldX is developed and maintained by the research group of Dr. Luis Serrano at the GRG. Users need to make an account and accept a yearly-renewable Licence. FoldX needs to be downloaded anew every year to work with the newly renewed license.
4. Homebrew installation is recommended for Mac OS, particularly for Mac OS 10.10 Yosemite. Binary distributions are recommended for Linux.
5. Experimentally determined protein structures can be found on the PDB depository (http://www.rcsb.org/). If the desired structure is not available, servers such as I-Tasser or Robetta can be used to generate homology models. It is important to note that PDB X-ray structures represent the asymmetric unit that may or may not correspond to the biological unit (quaternary structure). Users can download experimentally verified or predicted biological units from any of the PDB depositories.
6. Alternatively, you can manually curate your PDB file by analyzing the structure in PyMOL, making modifications and saving the modified structure or by editing the file directly in a text editor.
7. Multiple mutations can be specified by separating them with a comma in the same line, i.e., *TA41I,DA21K*; will evaluate a double mutant whereas if these two mutations appear in individual lines, two single mutants will be predicted.

8. This is a relative score representing the ΔΔG of folding; negative values are associated with stabilizing mutations.
9. The first six modes are rotation and translation modes. They should not be considered.
10. Energy is calculated as previously done [22, 46, 53] with higher values corresponding to more rigid structures.
11. The colors are scaled by the maximum absolute difference or three times the standard deviation, whichever is smaller.

Acknowledgments

R.J.N. is part of PROTEO (the Québec network for research on protein function, structure and engineering), and GRASP (Groupe de Recherche Axé sur la Structure des Protéines). The authors would like to thank Dr. Luis Serrano for giving his permission to use FoldX within the ENCoM server.

Funding: V.F. is the recipient of a Ph.D. fellowship from the Fonds de Recherche du Québec—Nature et Technologies (FRQ-NT); M.C. is the recipient of a Ph.D. fellowship from the Natural Sciences and Engineering Research Council of Canada (NSERC). NSERC Discovery Grant RGPIN-2014-05766.

References

1. Bommarius AS, Blum JK, Abrahamson MJ (2011) Status of protein engineering for biocatalysts: how to design an industrially useful biocatalyst. Curr Opin Chem Biol 15:194–200
2. Ulmer KM (1983) Protein engineering. Science 219:666–671
3. Ott K-H, Kwagh J-G, Stockton GW, Sidorov V, Kakefuda G (1996) Rational molecular design and genetic engineering of herbicide resistant crops by structure modeling and site-directed mutagenesis of acetohydroxyacid synthase. J Mol Biol 263:359–368
4. Diskin R, Scheid JF, Marcovecchio PM, West AP, Klein F, Gao H, Gnanapragasam PNP, Abadir A, Seaman MS, Nussenzweig MC, Bjorkman PJ (2011) Increasing the potency and breadth of an HIV antibody by using structure-based rational design. Science 334:1289–1293
5. Socha RD, Tokuriki N (2013) Modulating protein stability – directed evolution strategies for improved protein function. FEBS J 280:5582–5595
6. Khersonsky O, Roodveldt C, Tawfik D (2006) Enzyme promiscuity: evolutionary and mechanistic aspects. Curr Opin Chem Biol 10:498–508
7. Carlson JC, Badran AH, Guggiana-Nilo DA, Liu DR (2014) Negative selection and stringency modulation in phage-assisted continuous evolution. Nat Chem Biol 10:216–222
8. Dickinson BC, Leconte AM, Allen B, Esvelt KM, Liu DR (2013) Experimental interrogation of the path dependence and stochasticity of protein evolution using phage-assisted continuous evolution. Proc Natl Acad Sci U S A 110:9007–9012
9. Tinberg CE, Khare SD, Dou J, Doyle L, Nelson JW, Schena A, Jankowski W, Kalodimos CG, Johnsson K, Stoddard BL, Baker D (2013) Computational design of ligand-binding proteins with high affinity and selectivity. Nature 501:212–216
10. Jiang L, Althoff EA, Clemente FR, Doyle L, Röthlisberger D, Zanghellini A, Gallaher JL, Betker JL, Tanaka F, Barbas CF, Hilvert D, Houk KN, Stoddard BL, Baker D (2008) De novo computational design of retro-aldol enzymes. Science 319:1387–1391

11. Zhang S-B, Wu Z-L (2011) Identification of amino acid residues responsible for increased thermostability of feruloyl esterase A from Aspergillus niger using the PoPMuSiC algorithm. Bioresour Technol 102:2093–2096
12. Thiltgen G, Goldstein RA (2012) Assessing predictors of changes in protein stability upon mutation using self-consistency. PLoS One 7, e46084
13. Kepp KP (2014) Computing stability effects of mutations in human superoxide dismutase 1. J Phys Chem B 118:1799–1812
14. Teilum K, Olsen JG, Kragelund BB (2011) Protein stability, flexibility and function. Biochim Biophys Acta 1818:969–976
15. Poelwijk FJ, Kiviet DJ, Weinreich DM, Tans SJ (2007) Empirical fitness landscapes reveal accessible evolutionary paths. Nature 445:383–386
16. van den Burg B, Eijsink VGH (2002) Selection of mutations for increased protein stability. Curr Opin Biotechnol 13:333–337
17. Bloom JD, Meyer MM, Meinhold P, Otey CR, MacMillan D, Arnold FH (2005) Evolving strategies for enzyme engineering. Curr Opin Struct Biol 15:447–452
18. Shoichet BK, Baase WA, Kuroki R, Matthews BW (1995) A relationship between protein stability and protein function. Proc Natl Acad Sci U S A 92:452–456
19. Giver L, Gershenson A, Freskgard PO, Arnold FH (1998) Directed evolution of a thermostable esterase. Proc Natl Acad Sci U S A 95:12809–12813
20. Ruller R, Deliberto L, Ferreira TL, Ward RJ (2007) Thermostable variants of the recombinant xylanase a from Bacillus subtilis produced by directed evolution show reduced heat capacity changes. Proteins 70:1280–1293
21. Wolf-Watz M, Thai V, Henzler-Wildman K, Hadjipavlou G, Eisenmesser EZ, Kern D (2004) Linkage between dynamics and catalysis in a thermophilic-mesophilic enzyme pair. Nat Struct Mol Biol 11:945–949
22. Frappier V, Najmanovich RJ (2015) Vibrational entropy differences between mesophile and thermophile proteins and their use in protein engineering. Protein Sci 24:474–483
23. Jiménez-Osés G, Osuna S, Gao X, Sawaya MR, Gilson L, Collier SJ, Huisman GW, Yeates TO, Tang Y, Houk KN (2014) The role of distant mutations and allosteric regulation on LovD active site dynamics. Nat Chem Biol 10:431–436
24. Motlagh HN, Wrabl JO, Li J, Hilser VJ (2014) The ensemble nature of allostery. Nature 508:331–339
25. Gaudreault F, Chartier M, Najmanovich RJ (2012) Side-chain rotamer changes upon ligand binding: common, crucial, correlate with entropy and rearrange hydrogen bonding. Bioinformatics 28:i423–i430
26. Gaudreault F, Najmanovich RJ (2015) FlexAID: revisiting docking on non-native-complex structures. J Chem Inf Model
27. van den Bedem H, Bhabha G, Yang K, Wright PE, Fraser JS (2013) Automated identification of functional dynamic contact networks from X-ray crystallography. Nat Methods 10:896–902
28. Doucet N (2011) Can enzyme engineering benefit from the modulation of protein motions? Lessons learned from NMR relaxation dispersion experiments. Protein Pept Lett 18:336–343
29. Elvin JG, Couston RG, van der Walle CF (2013) Therapeutic antibodies: market considerations, disease targets and bioprocessing. Int J Pharm 440:83–98
30. Zimmermann J, Zimmermann J, Oakman EL, Oakman EL, Thorpe IF, Thorpe IF, Shi X, Shi X, Abbyad P, Abbyad P, Brooks CL, Brooks CL, Boxer SG, Boxer SG, Romesberg FE, Romesberg FE (2006) Antibody evolution constrains conformational heterogeneity by tailoring protein dynamics. Proc Natl Acad Sci U S A 103:13722–13727
31. Thielges MC, Zimmermann J, Yu W, Oda M, Romesberg FE (2008) Exploring the energy landscape of antibody–antigen complexes: protein dynamics, flexibility, and molecular recognition. Biochemistry 47:7237–7247
32. Boder ET, Midelfort KS, Wittrup KD (2000) Directed evolution of antibody fragments with monovalent femtomolar antigen-binding affinity. Proc Natl Acad Sci U S A 97:10701–10705
33. Smith CA, Kortemme T (2008) Backrub-like backbone simulation recapitulates natural protein conformational variability and improves mutant side-chain prediction. J Mol Biol 380:742–756
34. Kellogg EH, Leaver-Fay A, Baker D (2010) Role of conformational sampling in computing mutation-induced changes in protein structure and stability. Proteins 79:830–838
35. Davey JA, Chica RA (2013) Improving the accuracy of protein stability predictions with multistate design using a variety of backbone ensembles. Proteins 82:771–784
36. LeVine MV, Weinstein H (2014) NbIT – a new information theory-based analysis of allosteric mechanisms reveals residues that underlie function in the leucine transporter LeuT. PLoS Comput Biol 10, e1003603

37. Mahajan S, Sanejouand Y-H (2015) On the relationship between low-frequency normal modes and the large-scale conformational changes of proteins. Arch Biochem Biophys 567:59–65
38. Fuglebakk E, Tiwari SP, Reuter N (2015) Comparing the intrinsic dynamics of multiple protein structures using elastic network models. Biochim Biophys Acta 1850:911–922
39. Tirion M (1996) Large amplitude elastic motions in proteins from a single-parameter. Atom Anal Phys Rev Lett 77:1905–1908
40. Tama F, Gadea FX, Marques O, Sanejouand YH (2000) Building-block approach for determining low-frequency normal modes of macromolecules. Proteins 41:1–7
41. Abagyan R, Rueda M, Bottegoni G (2009) Consistent improvement of cross-docking results using binding site ensembles generated with elastic network normal modes. J Chem Inf Model 49:716–725
42. Park S-J, Kufareva I, Abagyan R (2010) Improved docking, screening and selectivity prediction for small molecule nuclear receptor modulators using conformational ensembles. J Comput Aided Mol Des 24:459–471
43. Alexandrov V, Lehnert U, Echols N, Milburn D, Engelman D, Gerstein M (2005) Normal modes for predicting protein motions: a comprehensive database assessment and associated Web tool. Protein Sci 14:633–643
44. Schröder GF, Brunger AT, Levitt M (2007) Combining efficient conformational sampling with a deformable elastic network model facilitates structure refinement at low resolution. Structure 15:1630–1641
45. Tama F, Valle M, Frank J, Brooks C (2003) Dynamic reorganization of the functionally active ribosome explored by normal mode analysis and cryo-electron microscopy. Proc Natl Acad Sci U S A 100:9319–9323
46. Frappier V, Najmanovich RJ (2014) A coarse-grained elastic network atom contact model and its use in the simulation of protein dynamics and the prediction of the effect of mutations. PLoS Comput Biol 10:e1003569
47. Lin T-L, Song G (2010) Generalized spring tensor models for protein fluctuation dynamics and conformation changes. BMC Struct Biol 10(Suppl 1):3
48. Atilgan AR, Durell SR, Jernigan RL, Demirel MC, Keskin O, Bahar I (2001) Anisotropy of fluctuation dynamics of proteins with an elastic network model. Biophys J 80:505–515
49. Dehouck Y, Grosfils A, Folch B, Gilis D, Bogaerts P, Rooman M (2009) Fast and accurate predictions of protein stability changes upon mutations using statistical potentials and neural networks: PoPMuSiC-2.0. Bioinformatics 25:2537–2543
50. Schymkowitz J, Borg J, Stricher F, Nys R, Rousseau F, Serrano L (2005) The FoldX web server: an online force field. Nucleic Acids Res 33:W382–W388
51. Leaver-Fay A, Tyka M, Lewis SM, Lange OF, Thompson J, Jacak R, Kaufman K, Renfrew PD, Smith CA, Sheffler W, Davis IW, Cooper S, Treuille A, Mandell DJ, Richter F, Ban Y-EA, Fleishman SJ, Corn JE, Kim DE, Lyskov S, Berrondo M, Mentzer S, Popović Z, Havranek JJ, Karanicolas J, Das R, Meiler J, Kortemme T, Gray JJ, Kuhlman B, Baker D, Bradley P (2011) ROSETTA3: an object-oriented software suite for the simulation and design of macromolecules. Methods Enzymol 487:545–574
52. Zhou H, Zhou Y (2002) Distance-scaled, finite ideal-gas reference state improves structure-derived potentials of mean force for structure selection and stability prediction. Protein Sci 11:2714–2726
53. Frappier V, Chartier M, Najmanovich RJ (2015) ENCoM server: exploring protein conformational space and the effect of mutations on protein function and stability. Nucleic Acids Res
54. Boehr DD, Schnell JR, McElheny D, Bae S-H, Duggan BM, Benkovic SJ, Dyson HJ, Wright PE (2013) A distal mutation perturbs dynamic amino acid networks in dihydrofolate reductase. Biochemistry 52:4605–4619
55. Gekko K, Yamagami K, Kunori Y, Ichihara S, Kodama M, Iwakura M (1993) Effects of point mutation in a flexible loop on the stability and enzymatic function of Escherichia coli dihydrofolate reductase. J Biochem 113:74–80

Part II

Software of Computational Protein Design Applications

Chapter 10

Computational Protein Design Under a Given Backbone Structure with the ABACUS Statistical Energy Function

Peng Xiong, Quan Chen*, and Haiyan Liu*

Abstract

An important objective of computational protein design is to identify amino acid sequences that stably fold into a given backbone structure. A general approach to this problem is to minimize an energy function in the sequence space. We have previously reported a method to derive statistical energies for fixed-backbone protein design and showed that it led to de novo proteins that fold as expected. Here, we present the usage of the program that implements this method, which we now name as ABACUS (A Backbone-based Amino aCid Usage Survey).

Key words Protein design, Statistical energy function, Backbone structure, Mutation analysis

1 Introduction

In protein design, one tries to determine the amino acid sequence of a protein so that it can fold into a certain structure and/or carry out a particular function. A number of successful examples of designed proteins have been reported with rule-based approaches [1–3]. Another type of approach, which is more general, is to computationally optimize the amino acid sequence to minimize an effective energy function [4–6]. Thus, this process can be referred to as automatic design. The quality or accuracy of the energy function is vital in automatic protein design. The energy function should measure or score the compliance of any arbitrary amino acid sequence with respective design objectives. For the objective of designing proteins to fold stably into a given backbone structure, the energy function should measure the free energy of the target conformation relative to alternative conformations, folded or not, as a function of the amino acid sequence. Such a relative free energy cannot be determined exactly. As a result, all

*Corresponding authors.

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_10,

current models are highly approximate. For example, in principle, both the target conformation and alternative conformations are involved in defining the relative free energy. Yet, in practice, alternative conformations are seldom considered explicitly. Despite the fact that the first example of an automatically designed protein to fold expectedly was reported more than a decade ago [4], the accuracy of energy functions for protein design remains to be the major bottleneck of protein design [7]. Recently, we have reported a new strategy to construct statistical energy function (SEF) for protein design [8]. To evaluate the SEF theoretically, ab initio structure predictions were applied to sequences designed for 40 native backbones of different fold classes. It was shown that a significantly higher ratio of target-like structures was predicted for sequences designed by the SEF compared with previous methods [8]. Experimentally, one designed protein was verified to fold as expected. In addition, to improve foldability, three other well-folded proteins were obtained as mutants of the original designed sequences as selected with a directed evolution approach [8, 9]. The structure of such a mutant was solved and turned out to agree with the design target [8]. We now call this approach ABACUS, acronym for *a b*ackbone-based *a*mino a*c*id *u*sage *s*urvey. In this protocol, we will describe in detail the usage of the program that implements the ABACUS SEF combined with van der Waals energy terms [10]. This program may be used to redesign complete or partial amino acid sequences under a given fixed backbone structure. It may also be used to analyze the amino acid preferences at a particular position of a protein as determined by its local structural environment and its interactions with other residues of the protein.

2 Materials

The program is written in Java and it should be able to run on any platform that has a proper Java environment setup (version 1.6.0 or above). In this protocol, we assume that the program is executed on a computer running the Linux operating system, using bash for command and shell script interpretation, and having java version 1.6.0 or above installed (*see* **Note 1**). It should be quite straightforward to adapt the process to other computer platforms. Commands and filenames are in italic and case sensitive.

1. Download the program and its associated preprocessed training data from the web page http://biocomp.ustc.edu.cn/Download.html. The programs were coded in Java. Compiled java classes are provided for anonymous download. For interested readers, source code can be provided on request.
2. Extract the package from the downloaded file *ABACUS.tar.gz* with the command

```
tar -xvzf ABACUS.tar.gz
```

3. Extract the preprocessed training data from the downloaded file *ABACUS.db.tar.gz* with the command

```
tar -xvzf ABACUS.db.tar.gz
```

4. Change to directory ABACUS and run the shell script to set up environmental variables like command search paths and so on:

```
source setup.sh
```

Add ABACUS/bin to your command search path with command

```
export PATH=${PATH}:${ABACUSPATH}/bin
```

The script *setup.sh* will also add the necessary setup commands to *.bashrc* under your home directory. If that is not what you want, you may edit the script to remove corresponding lines.

3 Method

3.1 Definition of the Energy Function

Since the work we reported in ref. 8, we have introduced several small modifications of the ABACUS. These modifications are described here and they reflect what have been implemented in the current version of the program.

1. The total energy comprises the SEF part and the van der Waals part. The SEF part is defined as a function of the rotamer sequence, and is a summation of single residue terms and pairwise terms,

$$E_{SEF}(r_1, r_2, \ldots, r_L) = \sum_{i=1}^{L} E_i(r_i) + \sum_{i=1}^{L} \sum_{\substack{j > i, \\ j \text{ in contact with } i}} w(i,j) E_{ij}(r_i, r_j) \tag{1}$$

in which L is the length of the protein, r_i is the rotamer choice at site i. The energy terms $E_i(r_i)$ and $E_{ij}(r_i, r_j)$ are the same as in ref. 8. Compared with ref. 8, the main change in Eq. (1) is the additional weight $w(i,j)$ that controls the contribution of different types of pairwise terms. The reasoning behind this weight is that strong coupling between multiple sites may cause the sum of pairwise interactions to significantly overestimate the overall interaction. For example, when the pairwise interactions (i,j) and (j,k) are strong and there is lack of average over j, the statistical term (i,k) may show strong interactions even when the actual interaction is weak. Because this problem originated from the lack of average, it mainly concerns positions that are close to each other in the primary sequence. We assumed that the weight $w(i,j)$ depends solely on the separation

between i and j in primary sequence, and determined its respective values as below based on single site redesign,

$$wt(i,j) = \begin{cases} 0.5, \ if \ |i-j| \leq 2 \\ 0.6, \ if \ 2 < |i-j| \leq 4 \\ 1.0, \ if \ |i-j| > 4 \end{cases} \quad (2)$$

2. Another change from ref. 8 is that a revised pseudo residue model is used to estimate the exposure to solvent of a position in a backbone structure. As in 3, all residue positions are substituted with the same fictitious residue with a pseudo side-chain. In contrast to the pseudo side chain proposed in ref 3 and used in ref. 8, the revised pseudo residue comprises three linearly connected atoms (noted as CB, CG, and CD). The bond lengths are 1.7 Å for CA-CB, 1.5 Å for CB-CG, and 1.7 Å for CG-CD. The CA-CB-CG and CB-CG-CD bond angles are 109.5°. The dihedral angle N-CA-CB-CG is 90°. The van der Waals radii of these pseudo atoms are 3.2 Å. Then, the solvent accessible surface area (SASA) of each atom in this model is calculated. For each residue position, $SASA_{res}$ is calculated as a weighted sum of the SASA of its pseudo side-chain atoms.

$$\mathrm{SASA_{res}} = 0.4 \times \mathrm{SASA(CB)} + 0.56 \times \mathrm{SASA(CG)} + 1.0 \times \mathrm{SASA(CD)}. \quad (3)$$

As in ref. 8, the $SASA_{res}$ is transformed into a solvent accessibility index (SAI) based on the relative rank among all positions in the training data set. Geometry of the pseudo sidechain and the weights in Eq. (3) have been optimized so that the resulting SAI for a position contains maximally residue-type information in the training data set.

3. As in ref. 8, van der Waals energies may be combined, with weights, with the SEF. The weights of the van der Waals terms depend solely on the solvent accessibility of the positions involved. Instead of categorizing the positions into three discrete categories and use one van der Waals weight for each category, in the current program the following continuous function is employed to determine the van der Waals weight based on SAI:

$$\mathrm{weight}(i) = 0.05 + \frac{0.48}{1 + e^{\frac{\mathrm{SAI}(i) - 0.4}{0.07}}}. \quad (4)$$

The energy table of Lennard-Jones potential of all pairs is calculated and saved before sequence design.

3.2 Preparation of the Input Backbone Structure

1. Prepare the target backbone structure as a normal PDB file. For the current implementation, all residues must have the same chain ID even if the chain is actually discontinuous or there are multiple chains. In our example, the target backbone will be specified in the input PDB file *1ubq.pdb*. Put this file in an empty directory and change to that directory.
2. Preprocess the input PDB file with the shell script *ABACUS_prepare*. For example, run the command *ABACUS_prepare 1ubq.pdb*.

   This script first runs a program to check the input PDB file to make sure that it contains a single chain ID (*see* **Note 2**), and that there is no missing backbone atom for every position. In addition, any nonstandard residue or cysteine residue will be replaced by alanine (cysteine is not yet supported by the current implementation of the van der Waals energy term). A new file *1ubq.noCys.pdb* will be generated. This file should be kept in the same directory as the input backbone file, as subsequent commands may use it as intermediate input.
3. Calculate the residue-wise properties for the input backbone, including secondary structure type, *SAI*, backbone torsion angles, and so on. Operations in this step should have been automatically carried out by *ABACUS_prepare*, so no additional command needs to be run. The shell script *ABACUS_prepare* will execute a program to generate another structure file (for example, *1ubq.psd*) in which all side chains have been substituted by the pseudo residue. Again in script *ABACUS_prepare*, this structure file is used as input to the program STRIDE [11] (modified here to accommodate the weighted sum of atomic SASA and named as *psdSTRIDE* under *${ABACUSPATH}/bin*) to calculate the residue-wise properties and the result is in file *1ubq.str*.

3.3 Calculation of Energy Tables

The various energy terms are calculated and stored as energy tables in files before sequence design. These are the computationally most expensive steps. Depending on the size of the problem and the computer hardware, the calculations will take some time to finish.

1. Generate SEF energy tables with the shell script *ABACUS_S1S2*. For example, the command

   ```
   ABACUS_S1S2 1ubq.pdb
   ```

   will generate two files: *1ubq.pa* contains energy tables for the single residue terms; *1ubq.sm* contains energy tables for the pairwise terms.
2. Generate van der Waals energy tables with the command *ABACUS_vdwEtable*. For example, run the command

   ```
   ABACUS_vdwEtable 1ubq.pdb
   ```

Or you may need to specify a resFile before this step (*see* **step 1** of Subheading 3.4 below), and run

```
ABACUS_vdwEtable 1ubq.pdb 1ubq.resFile
```

The file *1ubq.resFile* specifies which positions will be redesigned and what would be the allowed residue types at each position (*see* below). If this file is not provided, all positions will be considered for redesign and all residue types will be allowed. This command generates a file *1ubq.etable* that stores the van der Waals energy table and another intermediate file that will also be used in subsequent sequence design.

3.4 Sequence Optimization

In addition to the target backbone structure, the input PDB file may also contain residue types and side chain configurations for some residue positions. This information may either be maintained or discarded during sequence optimization. This can be specified in an input resFile (*see* **Note 3**). Sequence optimization is carried out using Monte Carlo simulated annealing. This is a random optimizer so each optimization run starts from a randomized starting sequence. Consequently, different runs generate similar (sequence identity above 80 %) but not exactly the same set of results.

1. Edit the resFile (for example, *1ubq.resFile*). In this file, each line refers to a backbone position. The first field of the line indicates the peptide chain ID of the position as in the input PDB file (if the input PDB file does not contain chain ID, use "_" as chain ID here). The second field indicates the residue ID of the position in the input PDB. The third field is either a lower case keyword, or an uppercase string comprising the one letter code of allowed amino acid residue types at this position. For this field, the keyword "all" means that all residue types except cysteine are allowed. The keyword "native" means the residue type from the input PDB file will be kept (the rotamer type may change in design). The resFile should be provided to the command *ABACUS_vdwEtable* that constructs the van der Waals energy table (*see* **step 2** of Subheading 3.3 above). If a resFile has indeed been provided in that step, positions not mentioned in this resFile will not be redesigned (i.e., has the same effect as the "native" keyword). For example, a resFile may contain the following lines

```
A 11 all
A 12 AILVF
A 13 DEKRQN
A 14 DEKRQN
A 15 APGS
A 16 native
```

With this file provided to *ABACUS_vdwEtable*, positions 11–15 of chain A will be redesigned with respective sets of allowed residue types by *ABACUS_design* (*see* below). For the remaining positions the input residue types will be maintained.

2. Generate optimized sequences with the command *ABACUS_design*. For example,

   ```
   ABACUS_design 1ubq.pdb 20
   ```

   will use the input backbone in 1ubq.pdb and design 20 sequences. Other files required by this command will be automatically searched in the directory that contains *1ubq.pdb*. The results will be stored as 20 structure files in PDB format (filenames *1ubq.design_001.pdb* to *1ubq.design_020.pdb*) and as 20 sequence files in fasta format (filenames *1ubq.design_001.fasta* to *1ubq.design_001.fasta*)
   The command

   ```
   ABACUS_design 1ubq.pdb 20 tag
   ```

   will do the same (replace "tag" with arbitrary tag string to tag your results), except that the result files will contain the string "tag" in their filenames, e.g., *1ubq.design_tag_001.pdb* and *1ubq.design_tag_001.fasta*. Use a different tag string in later runs of *ABACUS_design* if you do not want earlier results to be overwritten.

3. Visualize the design results with appropriate third-party tools. For example, the designed sequences may be visualized with the sequence logo generator [12] (http://weblogo.berkeley.edu/logo.cgi). From the sequence logo, one may check in the designed sequences which positions are highly conserved, and which are variable and to what extent. From our own experiences, we suggest that users of our program pay close attention to positions that are variable and contain both hydrophobic and hydrophilic amino acid types in different designed sequences. Such design results indicate that the residue types at these positions are underdetermined by the *ABACUS*. It could mean that the residue types at these positions are indeed underdetermined by their structural environment. In such a case, it should be just fine to choose any residue type for these positions. Alternatively, it could mean that the different *ABACUS* terms strongly "disagree" with each other on the amino acid preferences of these positions. Then inaccuracies in the different *ABACUS* terms, namely, the exaggeration or underestimation of certain interaction term relative to other interaction terms, would be exemplified at such positions. Based on biophysics intuitions, if such a position is on the surface, those sequences having a hydrophobic residue designed at this position and having a long fragment of successive hydrophobic

residues around the position should be considered as disfavored, as such sequences may lead to low solubility. Occasionally, a position in the interior of the protein would be designed as a polar residue, probably because the random sequence optimization has been trapped in a local minimum. Such sequences should usually also be discarded. An alternative choice is to specify the allowed residue types for such positions and rerun the design program.

A utility script is provided to single out surface positions designed as hydrophobic residues, interior positions designed as hydrophilic residues, or positions designed as residues of especially unfavorable single residue SEF energies. For example, run the command

```
ABACUS_suspiciousSites 1ubq.pdb 1ubq.design_001.pdb
```

or

```
ABACUS_suspiciousSites 1ubq.pdb 1ubq.design_001.fasta
```

The doubtful sites, if any, can be found in a file named *1ubq.singleMutationScan*. If suspicious sites are found in a designed sequence, it is up to the user to decide, based on the context, whether the sequence is worth further analysis, or another sequence should be selected, or, alternatively, the sequences should be redesigned with the respective positions constrained to a certain subset of residue types.

3.5 Analysis of SEF Energy Changes Associated with Single Mutations

The script *ABACUS_singleMutationScan* is used to calculate the change of the various SEF energy terms associated with all possible single mutations. For example, the command *ABACUS_singleMutationScan 1ubq.pdb output.txt* will calculate the changes in the single residue and the pairwise SEF terms associated with all possible single mutations. If you would like to restrict the single mutation to be at a particular position, add the position ID to the command line. For example,

```
ABACUS_singleMutationScan 1ubq.pdb output.txt 34
```

In *output.txt*, the field "delta S1" represents single residue terms, the field "delta S2" represents pairwise terms, and the field "delta SEF" represents the total SEF.

This command does not analyze van der Waals energy terms.

Please also *see* **Note 4** for suggested sequences of commands to execute for some typical applications of ABACUS.

4 Summary

Energy function plays the most critical role in computational protein design. As other statistical energy functions, ABACUS aims to

quantify how amino acid choices are constrained by target structures using information extracted from the large amount of known sequence-structure data of natural proteins. It requires no sequence or structural homology between design targets and training proteins. Thus, it can be applied to general designable backbone structures. In ref. 8, we have reported several well-folded *de novo* proteins designed with the method, some as mutants generated through directed evolution of designs that initially did not fold well. Since then, we have further refined the computational model in light of the directed evolution results. With the updated model, we have been able to design sequences that are well folded from the beginning for more target backbone structures, including those considered in ref. 8. We hope that the current protocol will be able to assist other researchers to use ABACUS as a helpful tool in their protein engineering efforts.

5 Notes

1. The current implementation has not yet been optimized for memory consumption. If you run into errors with messages like "java.lang.OutOfMemoryError: java heap space," you may need to enlarge the maximum internal memory of the Java virtual machine in relevant bash scripts.
2. In the current implementation, the input backbone must contain a single peptide chain. However, continuity of the peptide backbone is not assumed or required. Thus, if your design target comprises two or more chains, you may give them the same chain ID but make sure that every residue has a unique residue ID and has complete backbone atom coordinates. Side chain coordinates are ignored, so incomplete side chain coordinates are tolerated.
3. Several temporary files and result files will be generated in the directory containing the target PDB file. Consequently, it is suggested that the target PDB file is placed in a fresh new directory when starting a project. If different resFiles are used to constrain designs on the same target PDB file, these designs should be carried out with different working directories (that is the directory containing the target PDB file).
4. The sequence of commands for typical tasks can be like these. If you want to design new sequences, use these commands sequentially:

   ```
   ABACUS_prepare -> ABACUS_S1S2 ->ABACUS_vdwEtable ->
   ABACUS_design
   ```

If you want to find out suspicious sites in a (designed) PDB structure, use the following order of commands:

```
ABACUS_prepare -> ABACUS_S1S2 -> ABACUS_ suspiciousSites
```

If you want to carry out single mutation analysis, use the following order of commands:

```
ABACUS_prepare -> ABACUS_S1S2 -> ABACUS_singleMutationScan
```

Acknowledgments

This work has been supported by Chinese Ministry of Science and Technology (2011CBA00803 to Q.C. and 2012AA02A704 to H.L.) and National Natural Science Foundation of China (31200546 to Q.C. and 31370755 to H.L.).

References

1. Kamtekar S, Schiffer JM, Xiong H, Babik JM, Hecht MH (1993) Protein design by binary patterning of polar and nonpolar amino acids. Science 262(5140):1680–1685
2. Fedorov AN, Dolgikh DA, Chemeris VV, Chernov BK, Finkelstein AV, Schulga AA, Alakhov Yu B, Kirpichnikov MP, Ptitsyn OB (1992) De novo design, synthesis and study of albebetin, a polypeptide with a predetermined three-dimensional structure. Probing the structure at the nanogram level. J Mol Biol 225(4):927–931
3. Marshall SA, Mayo SL (2001) Achieving stability and conformational specificity in designed proteins via binary patterning. J Mol Biol 305 (3):619–631. doi:10.1006/jmbi.2000.4319
4. Dahiyat BI, Mayo SL (1997) De novo protein design: fully automated sequence selection. Science 278(5335):82–87
5. Kuhlman B, Dantas G, Ireton GC, Varani G, Stoddard BL, Baker D (2003) Design of a novel globular protein fold with atomic-level accuracy. Science 302(5649):1364–1368. doi:10.1126/science.1089427
6. Gainza P, Roberts KE, Georgiev I, Lilien RH, Keedy DA, Chen CY, Reza F, Anderson AC, Richardson DC, Richardson JS, Donald BR (2013) OSPREY: protein design with ensembles, flexibility, and provable algorithms. Methods Enzymol 523:87–107. doi:10.1016/B978-0-12-394292-0.00005-9
7. Li Z, Yang Y, Zhan J, Dai L, Zhou Y (2013) Energy functions in de novo protein design: current challenges and future prospects. Annu Rev Biophys 42:315–335. doi:10.1146/annurev-biophys-083012-130315
8. Xiong P, Wang M, Zhou X, Zhang T, Zhang J, Chen Q, Liu H (2014) Protein design with a comprehensive statistical energy function and boosted by experimental selection for foldability. Nat Commun 5:5330. doi:10.1038/ncomms6330
9. Foit L, Morgan GJ, Kern MJ, Steimer LR, von Hacht AA, Titchmarsh J, Warriner SL, Radford SE, Bardwell JC (2009) Optimizing protein stability in vivo. Mol Cell 36(5):861–871. doi:10.1016/j.molcel.2009.11.022
10. Pokala N, Handel TM (2005) Energy functions for protein design: adjustment with protein-protein complex affinities, models for the unfolded state, and negative design of solubility and specificity. J Mol Biol 347 (1):203–227. doi:10.1016/j.jmb.2004.12.019
11. Frishman D, Argos P (1995) Knowledge-based protein secondary structure assignment. Proteins 23(4):566–579. doi:10.1002/prot.340230412
12. Crooks GE, Hon G, Chandonia JM, Brenner SE (2004) WebLogo: a sequence logo generator. Genome Res 14(6):1188–1190. doi:10.1101/gr.849004

Chapter 11

Computational Protein Design Through Grafting and Stabilization

Cheng Zhu, David D. Mowrey, and Nikolay V. Dokholyan

Abstract

Computational grafting of target residues onto existing protein scaffolds is a powerful method for the design of proteins with novel function. In the grafting method side chain mutations are introduced into a preexisting protein scaffold to recreate a target functional motif. The success of this approach relies on two primary criteria: (1) the availability of compatible structural scaffolds, and (2) the introduction of mutations that do not affect the protein structure or stability. To identify compatible structural motifs we use the Erebus webserver, to search the protein data bank (PDB) for user-defined structural scaffolds. To identify potential design mutations we use the Eris webserver, which accurately predicts changes in protein stability resulting from mutations. Mutations that increase the protein stability are more likely to maintain the protein structure and therefore produce the desired function. Together these tools provide effective methods for identifying existing templates and guiding further design experiments. The software tools for scaffold searching and design are available at http://dokhlab.org.

Key words Scaffold search, Refinement, Stabilization, Mutation, Free energy, Protein design

1 Introduction

The goal of the protein design field is to engineer proteins with novel function with implications for developing new enzymes, biosensors, and therapeutics [1–4]. One approach to protein design is grafting, which has been successful in the developing of novel inhibitors [2, 5], biomarkers [6], enzymes [7], and antigens [8]. In the grafting approach one identifies an existing structural scaffold that can host a specific motif and replaces residues to match a desired active site or binding motif. This approach relies on the availability of potential scaffolds into which design mutations can be introduced. To this end, several methods have been developed to identify these scaffolds including Multigraft Match [9], GRAFTER [10], and MaDCaT [11]. These approaches rely on matching of backbone atoms, Cα-Cβ vectors, or Cα distance maps to identify potential scaffolds. In our approach we use the

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_11,

Erebus webserver, which allows greater user flexibility in matching user-specified atom types and residues, while maintaining high speed and accuracy [12] .

Having obtained a viable scaffold onto which target residues can be introduced, a further challenge is to determine a priori whether desired mutations will distort the original protein scaffold or even completely destabilize the protein structure. To improve thermodynamic stabilities of designed proteins, we use the protein stability prediction software Eris [13]. Eris has been shown to effectively predict effects of mutations, even for the more challenging case of small to large mutations, without the need for computationally intensive molecular dynamics simulations.

The overall workflow for our protein design protocol is composed of three major steps. In the first step we identify protein scaffolds for redesign using the Erebus. In the second step we submit the structure to the Chiron webserver for pre-relaxation prior to introducing mutations. In the final step we use the Eris webserver to identify potential redesign sequences. The theory and protocol for each of these methods are outlined in the following sections.

1.1 Identify Protein Scaffolds for Redesign

The identification of promising candidates for redesign makes use of the Erebus substructure search [12]. Provided with a query structure Erebus scans the protein data bank (PDB) for matching structural scaffolds [14]. Atom pairs of target structures in the structural database matching the name, residues, and distances in the query structure are collected to create the candidate substructures. The best substructures are selected based on their final weights representing the agreement between query and target structures. The final weights are calculated according to the following equation:

$$W = \left(\prod_{i=1}^{N} W_i\right)^{1/N} \prod_{j=1}^{M} (1 - w_j)$$

where the final weight (W) is the geometric mean of the weights (W_i) for each of the N atom pairs. The product is multiplied by an additional penalty (w_j) for each of the M missing atoms. Individual weights (W_i) for each atom pair are calculated according to the formula:

$$W_i = \mathrm{e}^{-\frac{\left(\Delta q_i - \Delta t_j\right)^2}{\sigma^2}}$$

Where the term $(\Delta q_i - \Delta t_j)$ represents the difference in distances between atom pairs i and j in the query (q) and target (t) structures, and σ^2 is a user-defined precision parameter.

1.2 Protein Preparation

As steric clashes are common structural artifacts observed in homology models and low-resolution crystal structures [15, 16], we first relax structures using Chiron before proceeding to Eris. In this method, a clash is defined as any atomic overlap resulting in energy greater than 0.3 kcal/mol (0.5 k_BT). Structural relaxation in Chiron is achieved by performing a series of short (~10 ps) discrete molecular dynamics simulations [17, 18], using a high heat exchange rate (5 fs) between the protein and the thermal bath. The exchange rate is used to effectively quench large atomic velocities resulting from large van der Waals clashes, which could result in broken bonds. To prevent large structural distortions, we also constrain backbone and Cβ atoms with a harmonic potential. The algorithm alternates between high temperature (0.7 ε/k_B, ~350 K) and low temperature (0.5 ε/k_BT, ~250 K) simulations until the overall clash score is less than 0.02 kcal/mol/contact. Pre-relaxing the structures in this way significantly improves sidechain packing of the protein core, which improves the accuracy of $\Delta\Delta G$ evaluations.

1.3 Protein Redesign

Protein redesign of preexisting scaffolds is accomplished using Eris [13]. Eris introduces a mutation or set of mutations into a protein structure and calculates free energies for both mutant (ΔG_{MUT}) and native (ΔG_{NAT}) structures. For free energy calculations rapid side-chain repacking and backbone relaxation are performed around the mutation site(s) using a Monte Carlo algorithm. The free energies are the result of a weighted sum of van der Waals forces, solvation, hydrogen bonding, and backbone-dependent statistical energies derived from the Medusa force field [19]. The final prediction of protein stability induced by mutations is expressed as the $\Delta\Delta G$ ($\Delta G_{MUT} - \Delta G_{NAT}$). Weighting parameters for free energy calculations were independently trained on 34 high-resolution X-ray protein structures and tested on a large dataset of 595 mutants where we found significant agreement between predicted and measured $\Delta\Delta G$ values ($R^2 = 0.75$) [13]. Furthermore, Eris can model the backbone flexibility, which is crucial for $\Delta\Delta G$ estimation of small-to-large mutations [20].

2 Software Requirements

The webservers of Erebus, Chiron, and Eris (ddg module) are freely available on our group page (http://dokhlab.org). The current version of Eris (ddg, scan and design module) also supports Linux/Unix-like platforms with the C and C++ compilers installed. It has been tested on Linux, Microsoft Windows (with the Linux port Cygwin), and Mac OS X. Our methods usually require a molecular viewer for preparation of crystal structures and analysis of results. For these purposes PyMol (http://pymol.org), an Open Source molecular viewer available on Windows, Mac OS X, and Linux, is used [21].

3 Methods

3.1 Erebus: Structure Search and Grafting

Erebus is a protein substructure search server (http://Erebus.dokhlab.org). It searches the entire PDB database for a match to a user-defined substructure, which can be any atoms from the backbone (N, Cα, O) or functional sites (see the following example). Erebus reads ATOM and HETATM records for atom coordinates in the PDB format and will only match atoms in a target substructure with atom names exactly matching those in the query structure [3, 4]. This feature is useful for identifying proteins that have the same catalytic sites, bind to similar small molecules or have the same backbone structures.

As an example, we used the copper-binding site in Cu/Zn superoxide dismutase (SOD1) to prepare the query structure and to find similar metal binding sites in PDB.

1. Create a file containing the atoms for the substructure query in PDB format. In our example these atoms are the Nδ or Nε atoms of H46, H48, H63, and H120 forming the copper-binding site in SOD1 (an example of the file is below). We saved this file as Cu_His.pdb:

ATOM 2 ND1 HIS A 46 11.519 -11.568 8.749 1.00 9.92 N

ATOM 4 NE2 HIS A 48 13.185 -15.110 8.862 1.00 11.95 N

ATOM 6 NE2 HIS A 63 10.975 -13.745 10.609 1.00 8.89 N

ATOM 8 NE2 HIS A 120 12.452 -13.024 6.807 1.00 2.00 N

END

2. Upload the query PDB file to the to Erebus server (Fig. 1).
3. After uploading the query PDB structure the user will have the option to adjust parameters for each atom in the search. Adjustable parameters include:

   'Residue Name': Under the 'Residue Name' column the user can specify the particular residue to match via a dropdown menu or may specify ANY to return matches from any residue.

   'Matching precision' or σ: This is a user-defined precision parameter (*see* Subheading 1.1). Smaller values for σ result in smaller deviations between the query and the targets.

   'Minimum weight' or *W*: This parameter measures how well the query and target structure match (*see* Subheading 1.1). Values for *W* range from 0 to 1, where smaller *W* means a worse match (the RMSD is bigger) and $W = 1$ is an identical match. The user can define the minimum acceptable *W*.

4. For this example Erebus finds over 100 matching structures in the PDB. These structures are ranked based on their RMSD to the query structure (Fig. 2). For each match the user can

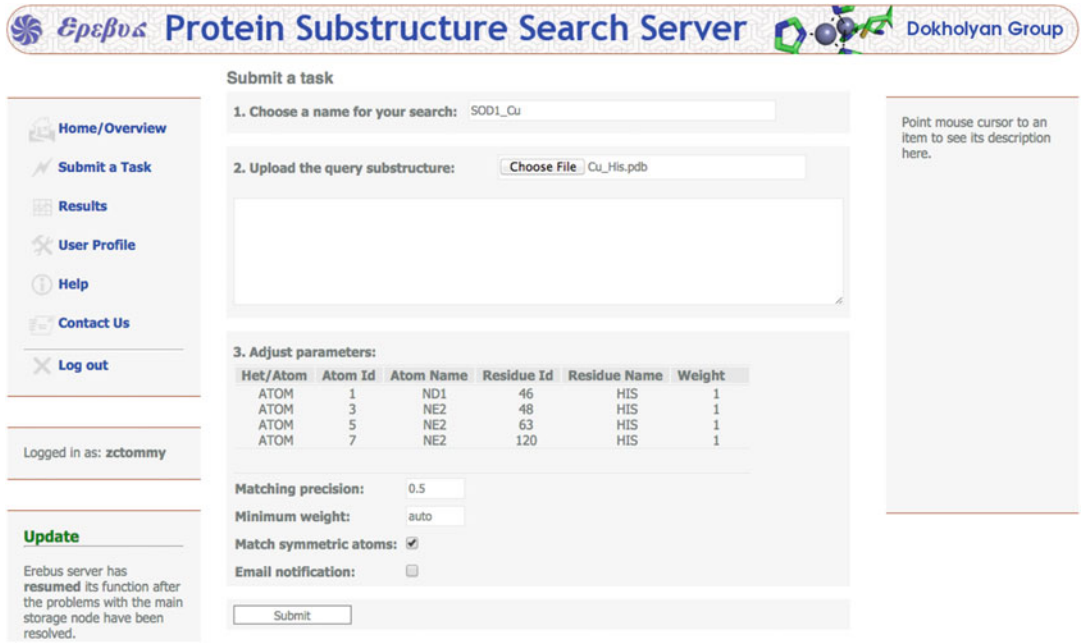


Fig. 1 The Erebus Web interface. Users may either upload a query PDB file or paste query coordinates into the field below in PDB format. Clicking a residue name under the 'Residue Name' column brings up a menu allowing the user to specify the particular residue to match. The 'ANY' selection allows for matches to any residue

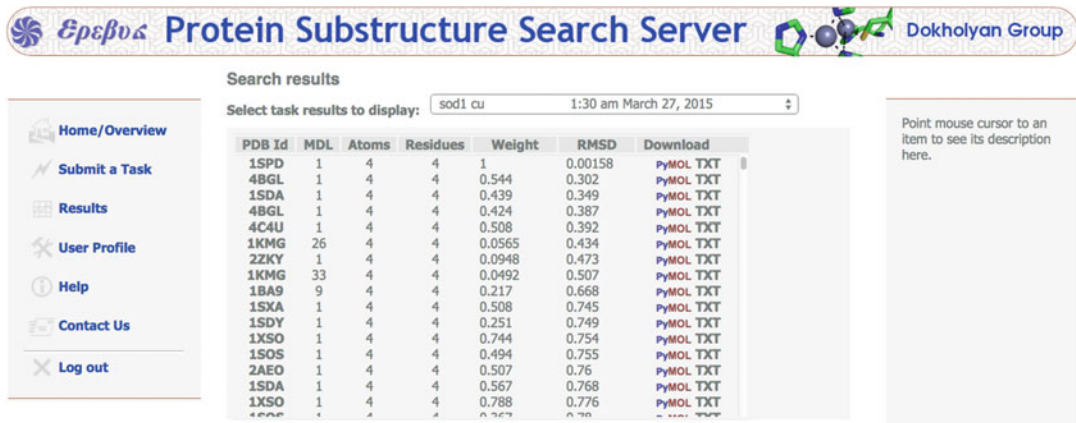


Fig. 2 Results from the Erebus scaffold search. The results are sorted by weight and root mean square deviation (RMSD) to the query structure

download a summary text file (TXT) and a structural model (PyMOL). The structural model file (.pml) is used to visualize the match between query and target structures (Fig. 3).

Erebus identified several protein families containing copper-binding site, including superoxide dismutase, laccase, and multi-copper oxidase. Several iron and zinc binding sites were also found.

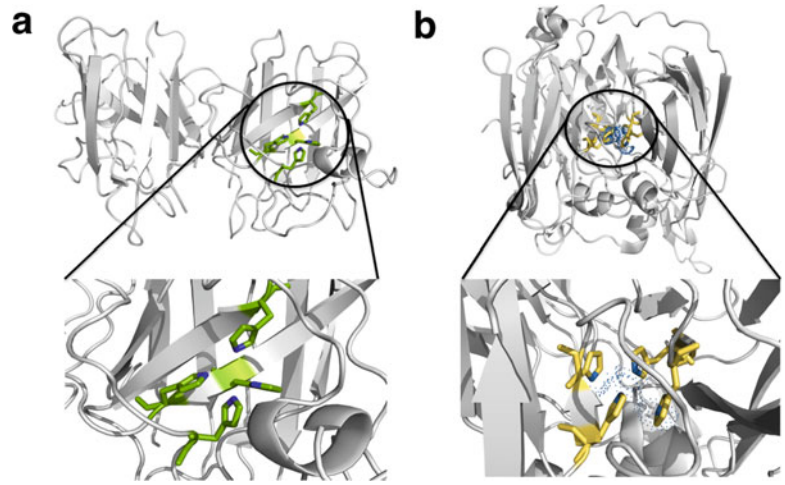


Fig. 3 Comparison of the copper binding site in the query structure (**a**) and one target structure (**b**). The query is based on the crystal structure of SOD1 (1SPD) and the target is a laccase (2XUW). The copper-interacting atoms (Nδ or Nε of histidine) are shown in *blue color*

3.2 Chiron: Preparation of Input Structures: Minimizes Steric Clashes Before Redesign

Chiron estimates the quality of a structure with respect to clashes and minimizes clashes using a series of short discrete molecular dynamics simulations [17, 18]. For homology models and low-resolution crystal structures in which structural artifacts often exist, Chiron is a useful refinement tool.

The Chiron webserver is freely available at http://chiron.dokhlab.org. After logging into the server, the user will be directed to the task submission page by clicking 'Submit Task' in the left panel. The user provides a structural model in PDB format or a PDB ID. During relaxation all backbone and Cβ atoms are constrained. To constrain all side-chain atoms not involved in steric clashes check the 'Constrain Sidechain' box (Fig. 4).

Results of the calculation can be accessed from the 'Home/Overview' page. Chiron outputs a refined structural model ({JOBID}.pdb) and the record of clashes ({JOBID}.py).

As an example, we submitted the crystal structure of Cu/Zn superoxide dismutase (SOD1, PDB ID: 1SPD) to the Chiron server. The resulting files are 12489.pdb and 12489.py. The following steps can be applied to visualize the clashes before and after the refinement:

1. Open 12489.pdb with PyMOL.
2. In the PyMOL command prompt, type 'run {Path}/ {JOBID}.py' and enter, omitting the quotation marks. The path to the .py should be indicated. In the provided example the command is 'run ~/user/12489.py'

The script generates two structural models: i-12489 is the structure of SOD1 before refinement and f-12489 is the structure

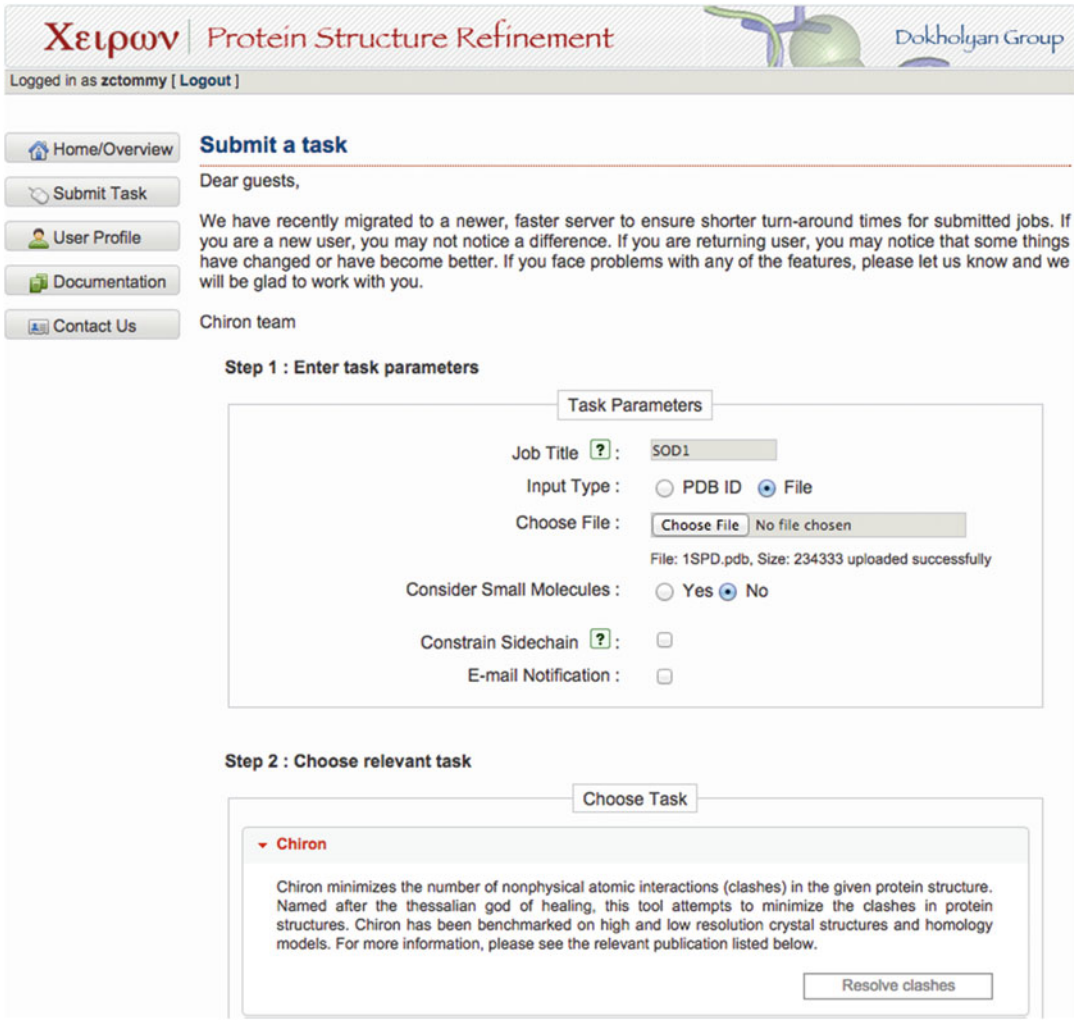


Fig. 4 Image of the Chiron Web interface

after refinement. The steric clashes are represented as color-coded (rainbow spectrum) cylinders of different radii. The clashes with higher repulsion energy are denoted as cylinders of larger radii. For SOD1, Chiron refinement successfully reduced the number of clashes and eliminated all major clashes (Fig. 5).

3.3 Eris: Identify Mutations That Stabilize a Protein Scaffold

Eris has three modules: ddg, scan, and design. The 'ddg' module exhaustively calculates the ΔΔG for individual mutations. The 'scan' module is used to rapidly search for stabilizing single mutations at a specified site. The 'design' function identifies the lowest-energy sequence for a given backbone, which can be a complete protein structure or a user-defined region in the whole structure. In the following section we elaborate the methods and procedures for each module.

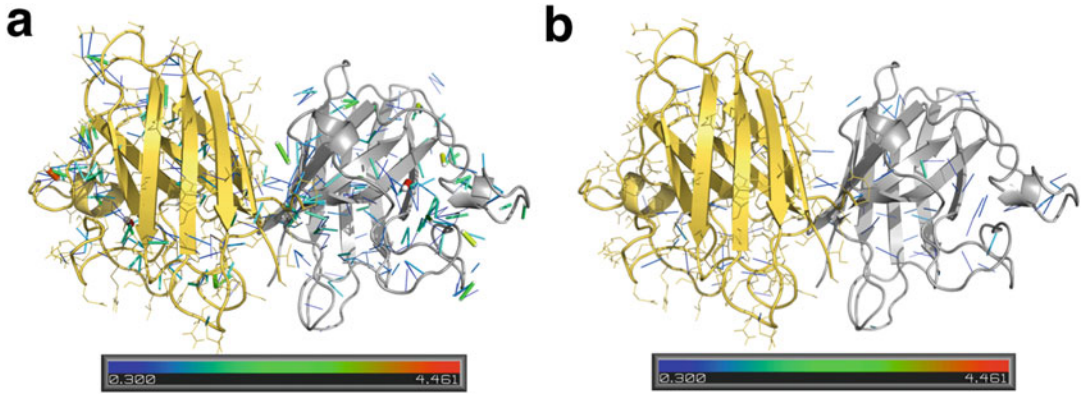


Fig. 5 Resolution of clashes using the Chiron webserver. Structures of SOD1 (PDB ID: 1SPD) are shown (**a**) before and (**b**) after refinement by Chiron. Colored cylinders connect atoms involved in clashes. The color and thickness of the cylinder denote the energy associated with the clash. Comparing panel **a** to panel **b** demonstrates that Chiron greatly reduces the number of clashes, and eliminates the large steric clashes

Table 1
Eris usage flags

Flag	Usage
-w	Specify the working directory
-j	Specify the 'JOBID'
-f	Flexible backbone (Not valid in 'scan' mode)
-r	Provide random seeds in Monte Carlo simulation
-m	Specify the mutation(s) for $\Delta\Delta G$ evaluations in 'ddg' mode
-s	Specify the site(s) in 'scan' mode
-d	Specify the path to a 'DesignTable' in 'design' mode
-h	Print the help, then exit
-v	Verbose output

3.3.1 Installation and Preparation for Input Files

The Eris package is available at http://moleculesinaction.com. Installation instructions are provided with the package (*see* **Note 1**).

The input file for all three modules should be in the PDB format. Currently, the Eris server will only read the first chain of a PDB file (*see* **Note 2**). Eris also renumbers the index of the first residue as 1. The common flags in Eris for command line usage are listed in Table 1.

3.3.2 ddg: An Estimation of ΔΔG for Given Mutations

For a single mutation or multiple substitutions, Eris-ddg repacks the sidechain 20 times using simulated annealing and computes stabilities by averaging all the conformations for both wild type and

mutant. The stability change, ΔΔG, is computed as the difference between the average stabilities of mutant and native structure ($\Delta G_{MUT} - \Delta G_{NAT}$).

As an example, we calculate the stability change induced by Y36E mutation in protein kinase B (PDB ID: 1UNP).

1. Submit the job:

   $ eris ddg –m {MUTATION} –w {DIRECTORY} –j {JOBID} INPUT.pdb

   MUTATION is a comma-delimited string of all mutations to be performed in a particular eris run in the format {native residue}{residue number}{target residue}. For example to change an alanine dipeptide to a serine dipeptide the input would be A1S, A2S.
   In our case we used the following command line and the results were kept in a folder called 'Eris_ddg':

   $ ~/Eris/eris ddg -m Y36E -w ~/Eris_ddg -j 1UNP_Y36E ~/Eris_ddg /1UNP.pdb

2. Analyzing the results:

   A summary of the results can be found in ~/Eris_ddg/eris_ddg.txt, which looks like:

   1UNP_Y36E ddg ~/Eris_ddg/1UNP.pdb Y36E 9.95
   9.95 7.11 0.11 2.84 0 0.81 0.05 0.40 0.46 -4.62 3.37

   The job (JOBID = 1UNP_Y36E) is the 'ddg' calculation for input structural model of 1UNP.pdb. The mutation (Y36E) is the substitution of Tyrosine at position 36 to Glutamic acid. The total ΔΔG equals to 9.95, i.e., destabilizing. In the second line the total stability change and its decomposition are illustrated (*see* **Note 3**).

   Structure files of both native and mutant proteins for each of the 20 rounds of calculation are stored in PDB format in the folder ~/Eris_ddg/1UNP_Y36E.

3.3.3 Scan: Search for Stabilizing Mutations

In the scan module a native amino acid at a specified site is substituted to all other 19 types of amino acids and only the stabilizing substitution ($\Delta\Delta G < 0$) are kept. If positions are not explicitly specified using '-s', then all the residues are scanned.

As an example, we used Eris-scan to find stabilizing mutations at position 37 in protein kinase B (PDB ID: 1UNP).

1. Submit the job:

   $ eris scan –s {SITE} –w {DIRECTORY} –j {JOBID} INPUT.pdb
   SITE is a comma-delimited string of integers specifying the residue positions to be scanned. Residue positions are determined from their order in the PDB file and Eris-scan

renumbers the index of the first residue as 0 ('-sN' means the scan is performed on the N+1 site).

In our case we used the following command line and the results were kept in a folder called 'Eris_scan':

$ ~/Eris/eris scan -s 36 -w ~/Eris_scan -j 1UNP ~/Eris_scan /1UNP.pdb

2. Analyzing the results:

The stabilizing mutations were listed in 'Eris_scan/1UNP/output/ddgStabilizing.dat', which looks like:

K37L -3.04264 -0.0944399 -0.0642849 -1.14021 0 -0.132146...
K37V -2.11104 0.0127475 -0.064736 -1.33434 0 0.0623585...
K37Q -0.111982 -0.878894 0.05282570.354524 0 0.0364916...
K37N -0.817968 0.106138 -0.10984 -0.245153 0 0.0981676...
...

Each line in the ddgStabilizing.dat specifies the stabilizing single mutations (*see* **Note 4**). In the same line, the numbers starting from the second column correspond to the total stability change and its decomposition.

The atomic structures of repacked conformations are stored in the same folder. The calculation results for all 19 substitutions were stored in ddgAll.dat.

3.3.4 Design: Find the Optimal Amino Acid Sequence for the Given Protein Backbone

In the 'design' module, users specify the protein segments to optimize and which subset of amino acids can be used to substitute the original one (polar, hydrophobic or user-defined subsets). The search can be performed using either a fixed backbone protocol (C, O, CA, and N positions fixed during design), or a flexible backbone protocol (allowing small adjustments of the backbone atoms to minimize energy). Eris-design then searches the lowest-energy sequence that satisfies the constraints listed in the design table.

Before submitting the 'design' job, the user should prepare a design table in .txt format. The design table consists of two columns: the first column (Index) specifies the mutation sites and the second column (Keyword) defines the subset of amino acids for substitution.

Values in the index column can be a single integer (m), a set of integers separated by commas (m,n,..), a range defined as m − n (m < n), or a mixture such as (a−b,c,d,e−f). The "DEFAULT" keyword can be used to represent all residues that have not been explicitly specified.

The keyword column takes as an argument one of a list of predefined flags. The flags and definitions are listed in Table 2.

For clarity an example design table for protein kinase B (PDB ID: 1UNP) is shown below:

Table 2
Keywords and definitions for Eris design table

Flag	
ALLAA	All available amino acids and the corresponding rotamers
NATAA	Fixed with native amino acid, but with all its available rotamers
NAROT	Fix the amino acid in its native rotamer but with sub-rotamer allowed
FIXNR	Fix the amino acid in its native rotamer without sub-rotamer motion
POLAR[a]	Polar amino acids and their rotamers
HYDPH[b]	Hydrophobic amino acids
AROMA[c]	Aromatic amino acids
PIKAA	User selected amino acids represented by single letter

[a]POLAR includes: SER, THR, GLN, GLU, ASN, ASP, LYS, ARG, HIS
[b]HYDPH includes: GLY, ALA, MET, VAL, LEU, ILE, PHE, TYR, TRP, PRO, CYS
[c]AROMA includes: PHE, TYR, TRP, HIS

#Index Keyword
DEFAULT NATAA
4-10 ALLAA
102, 112 PIKAA STYWL
50,69-76 HYDPH QEND

This design table will perform all-amino substitutions for residues 4 through 10, substitute residues 102 and 112 with Ser, Thr, Tyr, Trp, and Leu. And substitute residues 50 and 69 through 76 with hydrophobic amino acids. The QEND flag serves to denote the end of the design table

1. Submit the job:

   $ eris design –d {DesignTable} –w {DIRECTORY} –j {JOBID} STUCTURE.pdb

   In our case we used the following command line and the results were kept in a folder called 'Eris_design' (*see* **Note 5**):

   $ ~/Eris/eris design -d ~/DesignTable.txt -w ~/Eris_design -j 1UNP ~/Eris_design/1UNP.pdb

2. Analyzing the results
   The output of Eris-design is a PDB file of redesigned structural model and its free energy. In this module 20 rounds of Monte

Carlo simulations are performed. The results are given as design.run [00-19] and kept in Eris_design/1UNP/design. In each ".run" file, the first 20 lines record the temperature, Monte Carlo acceptance rate, total energy and its decompositions. The following lines are in PDB format so that the user can open it with PyMOL.

For example, design.run00 looks like:

```
0 10 0.763302 222.253 -409.689 371.847 319.997...
1 6.35799 0.70095 120.507 -402.49 196.314 296.97...
...
19 0.101625 0.0279097 -184.007 -464.69 8.64136 316.316...
ATOM 1 N ASP A 1 31.522 1.268 -6.333 1.00 0.00 N
ATOM 2 CA ASP A 1 30.972 2.648 -6.220 1.00 0.00 C
...
ATOM 1245 2HH2 ARG A 119 22.086 -5.596 -26.587 1.00 0.00 H
TER
```

3.3.5 An Online Server for 'ddg' Calculations

A Web-based Eris server for $\Delta\Delta G$ estimation is freely accessible online (http://eris.dokhlab.org). The users can follow the simple procedures listed below to submit their own task after registration:

1. Use the 'Submit a Task' bar on the left to submit the protein structure file. It can be a PDB ID or your own .pdb file. Eris only recognizes the first chain by reading the 'TER' line in a .pdb file.
2. Click on any residue site and choose the amino acids you want to substitute (Fig. 6).
3. Choose 'Fixed Backbone' or 'Flexible Backbone' and choose whether you want to include a pre-relaxation of backbone structure. Pre-relaxation remarkably improves the prediction accuracy when a high-resolution protein structure is not available. Alternatively the users can use Chiron to minimize steric clashes in the input structure (*see* **Note 6**).

4 Notes

1. After installation, typing "eris" without any command line arguments will display the brief help information. Typing "eris −h" will bring more detailed instructions.
2. The Eris webserver only reads the first chain of the provided PDB file. To modify the protein chains, the user can apply the 'alter chain' and 'alter resi' command in PyMOL for this modification (http://www.pymolwiki.org).

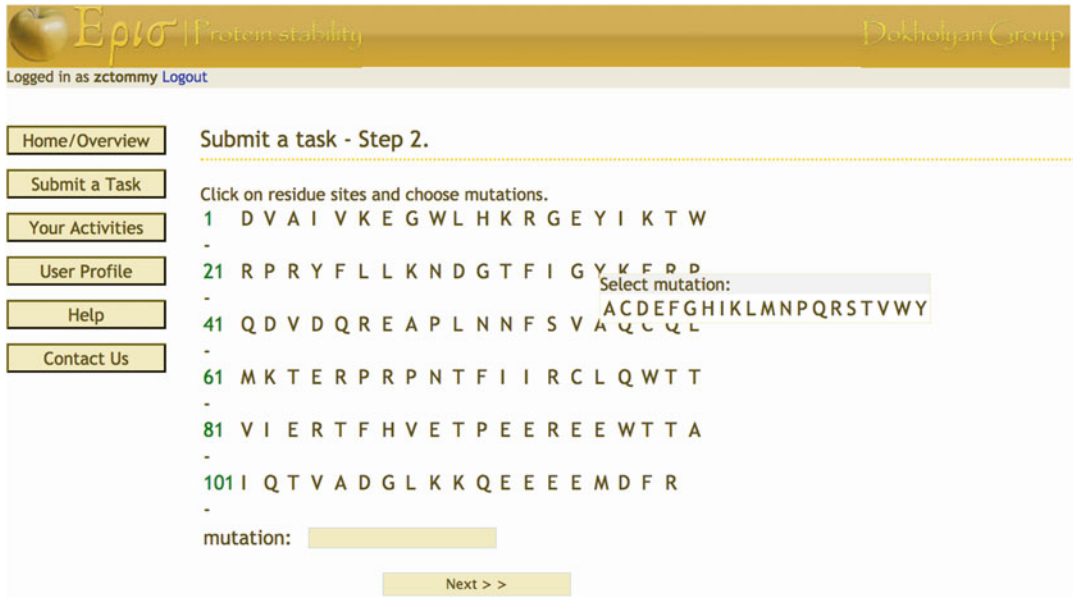


Fig. 6 Image of the Eris Web interface. Clicking on a residue results in a pop-up window allowing the user to select the target residue

3. The energy decomposition of total $\Delta\Delta G$ is composed of ten values. From left to right they are van der Waals attraction, van der Waals repulsion, solvation, backbone hydrogen bonds, backbone–side chain hydrogen bonds, side chain–side chain hydrogen bonds, backbone dependent statistical energy for amino acid, backbone-dependent statistical energy of the rotamer, reference energy of the unfolded states, and the correction for reference energy, respectively [19].
4. Eris-scan does not support flexible-backbone protocol and the Monte Carlo simulation is performed only once. Due to the limitations of the fixed-backbone method and sampling inefficiency, atomic clashes may not be resolved during structure minimization. These clashes can be identified by checking the van der Waals repulsion energy terms in the results. We find that the predictions are relatively more accurate for buried residues than exposed residues.
5. Eris uses a Monte Carlo algorithm for identifying changes in $\Delta\Delta G$. As such there is a certain amount of stochasticity in the results. We suggest that the user run the calculation multiple times to check for convergence in the distribution of energies. Performing multiple rounds of Eris calculation can be performed using the following bash script.

   CUR_DIR=‘pwd‘

   for i in {1..N}; do

```
~/eris ddg -m Y36E -w $CUR_DIR -j 1UNP_Y36E_${i} -r
    $RANDOM $CUR_DIR/1UNP.pdb
done
```

6. If there is difficulty with formatting or running the calculations it is suggested that the user submits the structural model to Chiron first. The server will both reformat the file to be compatible with Eris and fix clashes that could produce problems with the Eris calculations.

Acknowledgment

This work was supported by National Institutes of Health Awards R01GM080742 and R01AI102732.

References

1. Mandell DJ, Kortemme T (2009) Computer-aided design of functional protein interactions. Nat Chem Biol 5(11):797–807. doi:10.1038/nchembio.251
2. Martin L, Stricher F, Misse D, Sironi F, Pugniere M, Barthe P, Prado-Gotor R, Freulon I, Magne X, Roumestand C, Menez A, Lusso P, Veas F, Vita C (2003) Rational design of a CD4 mimic that inhibits HIV-1 entry and exposes cryptic neutralization epitopes. Nat Biotechnol 21(1):71–76. doi:10.1038/nbt768
3. Tinberg CE, Khare SD, Dou J, Doyle L, Nelson JW, Schena A, Jankowski W, Kalodimos CG, Johnsson K, Stoddard BL, Baker D (2013) Computational design of ligand-binding proteins with high affinity and selectivity. Nature 501(7466):212–216. doi:10.1038/nature12443
4. Schulze H, Vorlova S, Villatte F, Bachmann TT, Schmid RD (2003) Design of acetylcholinesterases for biosensor applications. Biosens Bioelectron 18(2-3):201–209
5. Sia SK, Kim PS (2003) Protein grafting of an HIV-1-inhibiting epitope. Proc Natl Acad Sci U S A 100(17):9756–9761. doi:10.1073/pnas.1733910100
6. Hao J, Serohijos AW, Newton G, Tassone G, Wang Z, Sgroi DC, Dokholyan NV, Basilion JP (2008) Identification and rational redesign of peptide ligands to CRIP1, a novel biomarker for cancers. PLoS Comput Biol 4(8), e1000138. doi:10.1371/journal.pcbi.1000138
7. Rothlisberger D, Khersonsky O, Wollacott AM, Jiang L, DeChancie J, Betker J, Gallaher JL, Althoff EA, Zanghellini A, Dym O, Albeck S, Houk KN, Tawfik DS, Baker D (2008) Kemp elimination catalysts by computational enzyme design. Nature 453(7192):190–195. doi:10.1038/nature06879
8. Drakopoulou E, Zinn-Justin S, Guenneugues M, Gilqin B, Menez A, Vita C (1996) Changing the structural context of a functional beta-hairpin. Synthesis and characterization of a chimera containing the curaremimetic loop of a snake toxin in the scorpion alpha/beta scaffold. J Biol Chem 271(20):11979–11987
9. Azoitei ML, Correia BE, Ban YE, Carrico C, Kalyuzhniy O, Chen L, Schroeter A, Huang PS, McLellan JS, Kwong PD, Baker D, Strong RK, Schief WR (2011) Computation-guided backbone grafting of a discontinuous motif onto a protein scaffold. Science 334(6054):373–376. doi:10.1126/science.1209368
10. Hearst DP, Cohen FE (1994) GRAFTER: a computational aid for the design of novel proteins. Protein Eng 7(12):1411–1421
11. Zhang J, Grigoryan G (2013) Mining tertiary structural motifs for assessment of designability. Methods Enzymol 523:21–40. doi:10.1016/B978-0-12-394292-0.00002-3
12. Shirvanyants D, Alexandrova AN, Dokholyan NV (2011) Rigid substructure search. Bioinformatics 27(9):1327–1329. doi:10.1093/bioinformatics/btr129
13. Yin S, Ding F, Dokholyan NV (2007) Eris: an automated estimator of protein stability. Nat Methods 4(6):466–467. doi:10.1038/nmeth0607-466
14. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, Shindyalov IN, Bourne PE (2000) The Protein Data Bank. Nucleic Acids Res 28(1):235–242

15. Hooft RW, Vriend G, Sander C, Abola EE (1996) Errors in protein structures. Nature 381(6580):272. doi:10.1038/381272a0
16. Ramachandran S, Kota P, Ding F, Dokholyan NV (2011) Automated minimization of steric clashes in protein structures. Proteins 79 (1):261–270. doi:10.1002/prot.22879
17. Ding F, Tsao D, Nie H, Dokholyan NV (2008) Ab initio folding of proteins with all-atom discrete molecular dynamics. Structure 16 (7):1010–1018. doi:10.1016/j.str.2008.03.013
18. Dokholyan NV, Buldyrev SV, Stanley HE, Shakhnovich EI (1998) Discrete molecular dynamics studies of the folding of a protein-like model. Fold Des 3(6):577–587. doi:10.1016/S1359-0278(98)00072-8
19. Ding F, Dokholyan NV (2006) Emergence of protein fold families through rational design. PLoS Comput Biol 2(7), e85. doi:10.1371/journal.pcbi.0020085
20. Yin S, Ding F, Dokholyan NV (2007) Modeling backbone flexibility improves protein stability estimation. Structure 15(12):1567–1576. doi:10.1016/j.str.2007.09.024
21. The PyMOL Molecular Graphics System, Version 1.7.4 Schrödinger, LLC

Chapter 12

An Evolution-Based Approach to De Novo Protein Design

Jeffrey R. Brender, David Shultis, Naureen Aslam Khattak, and Yang Zhang

Abstract

EvoDesign is a computational algorithm that allows the rapid creation of new protein sequences that are compatible with specific protein structures. As such, it can be used to optimize protein stability, to resculpt the protein surface to eliminate undesired protein-protein interactions, and to optimize protein-protein binding. A major distinguishing feature of EvoDesign in comparison to other protein design programs is the use of evolutionary information in the design process to guide the sequence search toward native-like sequences known to adopt structurally similar folds as the target. The observed frequencies of amino acids in specific positions in the structure in the form of structural profiles collected from proteins with similar folds and complexes with similar interfaces can implicitly capture many subtle effects that are essential for correct folding and protein-binding interactions. As a result of the inclusion of evolutionary information, the sequences designed by EvoDesign have native-like folding and binding properties not seen by other physics-based design methods. In this chapter, we describe how EvoDesign can be used to redesign proteins with a focus on the computational and experimental procedures that can be used to validate the designs.

Key words Protein design, Evolutionary profile, Protein structure modeling, Experimental protein validation, Recombinant expression, Circular dichroism, Nuclear magnetic resonance

1 Introduction

Computational protein design has expanded in recent years from the prediction of the effects of single site mutations to the complete redesign of entire proteins, including the alteration of protein-protein binding affinity and specificity [1–4], enzymatic activity [5, 6], and even the creation of new folds [7] and functions [8] that are not seen in nature. On the theoretical side, protein design has been used to find the sequence constraints necessary to generate specific folds or functions [9–11]. Through the use of these constraints, fundamental questions in protein evolution have been addressed by distinguishing what is physically possible from what is actually observed in evolution [10, 12].

However, full protein redesign beyond the mutation of a few hot spot residues, called *de novo* design, is computationally difficult,

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_12,

which is reflected in the relatively low successful percentage of successful designs. Most algorithms for de novo protein design approach the problem as reverse ab initio protein folding, evaluating the energy of the sequence according to all-atom physical potentials. Several problems become apparent in the naïve application of this approach: (1) A very large number of sequences must be considered, which limits the force field to only approximate energy terms that can be rapidly calculated; (2) there is a mismatch between the low-resolution models generated in the sequence search and the all-atom physical potentials used for evaluation. To make the design simulation computationally tractable, the possible conformations of the side-chains of the protein are restricted to a limited set of discrete rotamer conformations. The small steric clashes that necessarily result from this approximation force the use of dampened potentials that may miss subtle interactions that exist in the native protein [13, 14]; (3) the sequence search is considered only with the protein in isolation, not as the protein actually exists in the cellular context. This causes subtle problems in the real-life application of the designed proteins, particularly with respect to aggregation, as the highly hydrophobic sequences favored by folding energetics generally adopt highly compact sequences in silico but tend to aggregate in reality when actually expressed [15].

One approach to handle these challenges is to increase the accuracy of the design process by attempting to model physical reality at a higher resolution. In this spirit, design methodologies have been created that explicitly consider multiple conformations of the folded protein using ensemble techniques for multistate design [16–18] or that explicitly consider the unfolded state during the design process [18]. Alternatively, other design methodologies have been created that recognize the inherent inaccuracy of the force fields and attempt to diminish the effects of known inaccuracies. One example is the use of soft-core potentials that lessen repulsive interactions, preventing strongly unfavorable interactions that can be alleviated by small backbone motions from overriding the other terms [19]. Another example of this approach is the inclusion of additional terms in the force field that consider factors relevant to real proteins that are missing in the simulation, for example, the explicit consideration of inappropriate hydrophobic surfaces to limit aggregation in the designed sequences [18, 20]. The ongoing development of these methods has contributed greatly to the field and has led to some spectacular successes. However, complete de novo protein design is still a difficult process with routine application still in the future.

An alternative approach, based on hard-won knowledge from protein fold-recognition and structure prediction [21–24], is to recognize that evolution implicitly encodes information on protein folds and binding interactions that greatly exceeds our ability to

describe it through reductionist, physics-based methods. This evolution-based method approach to protein design differs from the physics-based methods in that most energy terms are not dependent on the full-atom representation of each tested sequence, whose inaccuracy is a significant source of error. Instead, the sequence space search is constrained by the sequence and structural profiles collected from structurally analogous families, assisted by neural network predictions of local structural features, including secondary structure, backbone torsion angle, and solvation [25, 26].

2 Methods

2.1 EvoDesign: Evolution-Based Method to Design Protein Folds and Interactions

The principle of EvoDesign follows the critical lessons learned from threading-based protein structure prediction methods, i.e., to use the reliable "finger-print" of nature of multiple proteins from the same family in the form of structural profile information to guide the simulation to the sequence search. It first collects a set of proteins with similar folds to the target scaffold structure from the PDB library by the structural alignment program TM-align [27], using a TM-score cutoff value to define structural similarity (Fig. 1) [28]. In the second step, this set of structurally similar folds is used to create a position specific scoring matrix $M(p, a)$ for evaluating potential sequences [29, 30].

To create the position specific scoring matrix, first a multiple sequence alignment (MSA) is generated according to the pair-wise structural alignments between the structural analogs identified in the first step and the target structure (Fig. 1). An $L \times 20$ matrix (where L = length of the protein) is then created according to

$$M(p, a) = \sum_{x=1}^{20} w(p, x) \times B(a, x) \tag{1}$$

where x represents a particular amino acid, $B(a, x)$ is the BLOSUM62 substitution matrix [31] for amino acid x to amino acid a, and $w(p, x)$ is the frequency of the amino acid x appearing at position p in the MSA created by TM-align. The matrix $M(p, a)$ serves as a structural profile to guide the sequences toward native-like sequences known to adopt structurally similar folds as the target (Fig. 1).

While the structural profile as given by the position specific scoring matrix $M(p, a)$ is efficient in guiding the global fold, optimization on the profile alone can result in singularities (i.e., disjointed "islands") in local sequences. To smoothen these singularities, back propagation neural network predictors are used to estimate the secondary structure (SS), solvent accessibility (SA), and torsion angles (φ/ψ) of the sequence. Unlike other predictors for these

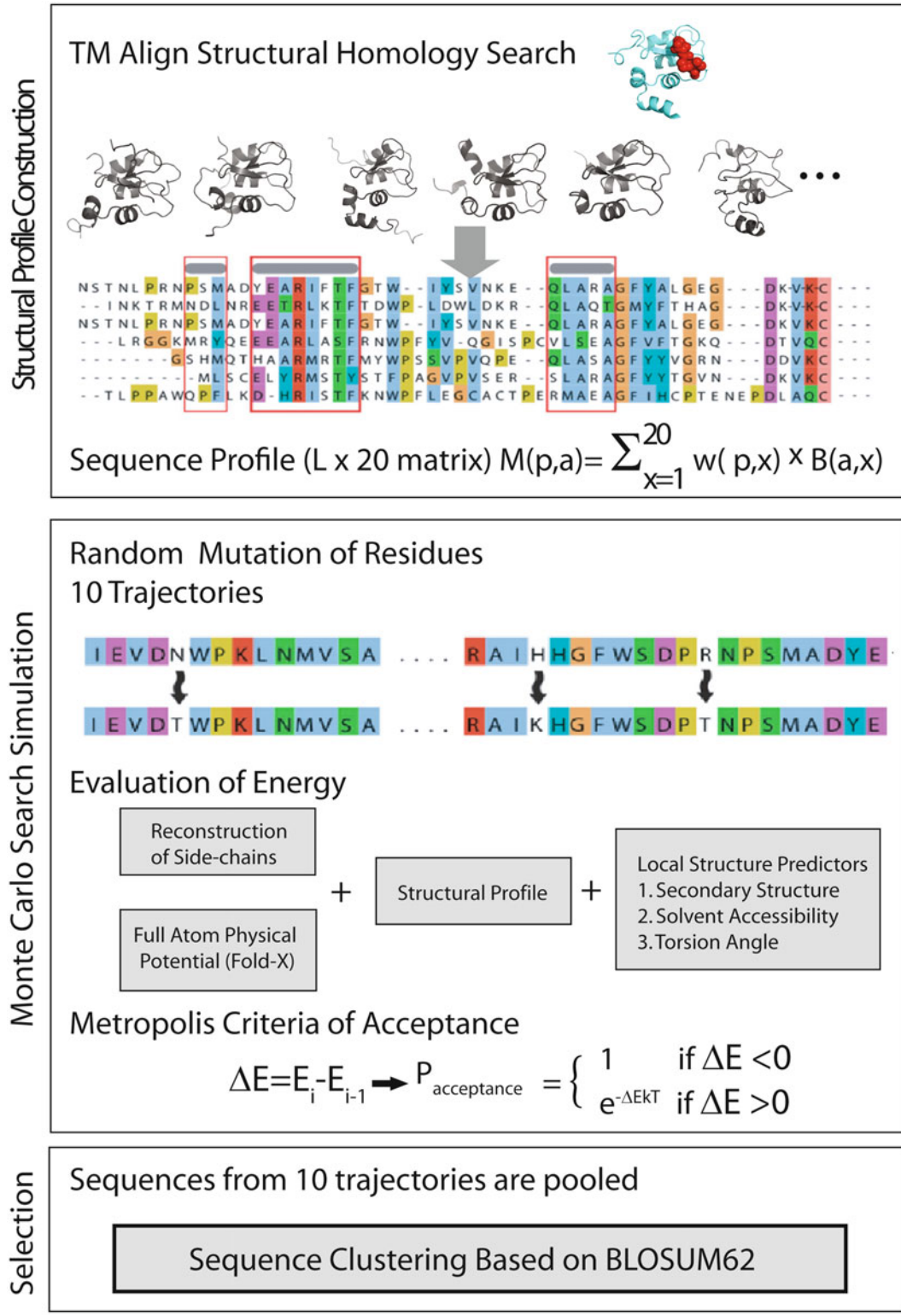


Fig. 1 Overview of the EvoDesign method showing the construction of the structural profile, the Monte Carlo search in sequence space, and the final selection of the sequences by sequence clustering

properties [32–34], these single-sequence-based predictors do not require a computationally expensive PSI-BLAST search, which considerably speeds up prediction at little cost in accuracy [25].

The evolutionary potential in EvoDesign is defined as the maximum score of the optimal alignment path between the decoy and target structure obtained by Needleman-Wunsch dynamic programming, giving the energy function:

$$E_{\text{evolution}} = \sum_{\max} [M(p,a) + w_1 \Delta\text{SS}(p) + w_2 \Delta\text{SA}(p) + w_3(\Delta\varphi(p) + \Delta\psi(p))], \tag{2}$$

where ΔSS, ΔSA, $\Delta\varphi$, and $\Delta\psi$ are the difference in secondary structure, solvent accessibility and torsion angles between the target assignments, and the predictions from the decoy sequences. The weighting factors (w_i) are decided by the relative accuracy of the single-sequence-based predictions for each term on a training set [25].

A physics-based potential can be used to predict potential favorable and unfavorable interactions among side-chains, such as steric interactions, which may be missed by the evolutionary-based terms defined above. While our computational benchmark results indicate the evolution-based energy function alone is sufficient to design protein sequences, adding a physics-based energy term from FoldX [35] improved the atomic packing of the local structures based on both computational structure prediction and experimental structure validations [25]. In this case, a full-atom representation of the sequence is needed which is created by SCWRL [36].

The final force field for single-chain protein design in EvoDesign is given by the weighted Z-scores of the evolution and physics-based terms:

$$E = w_4 \frac{E_{\text{evoluation}} - \langle E_{\text{evoluation}} \rangle}{\delta E_{\text{evoluation}}} + w_5 \frac{E_{\text{foldX}} - \langle E_{\text{foldX}} \rangle}{\delta E_{\text{foldX}}}, \tag{3}$$

where $\langle \ldots \rangle$ and δ indicate the average and standard deviation of the energy terms.

To actually generate the designed sequences, Monte Carlo searches are performed starting from 10 random sequences that are updated by random residue mutations (Fig. 1). Due to the imprecision of the force field, the lowest energy states do not always correspond to the best sequence design. Instead of simply focusing on the lowest energy sequence, the sequences from all 10 runs are pooled and the sequence with the maximum number of neighbors is identified using the SPICKER clustering algorithm [37] where the pair-wise distance between sequences is measured by the sum of the BLOSUM62 substitution scores [38].

The above procedure finds sequences compatible with the target structure. To introduce new or altered functionality into

the protein, the affinity of existing protein-protein interfaces can be improved by EvoDesign or new interfaces created though the optimization of non-native complexes created by docking. To modify interfaces, EvoDesign uses a multiscale approach incorporating a variety of features at different levels of structural resolution (Fig. 2).

Similar to the design of protein folds with EvoDesign, a key feature of the binding potential is the mixture of physics-based and evolutionary terms in the energy function [39]. For interface modification, the evolutionary terms are created from the structural alignment of similar interfaces from the nonredundant COTH structural library of dimeric proteins [40] by the IAlign program

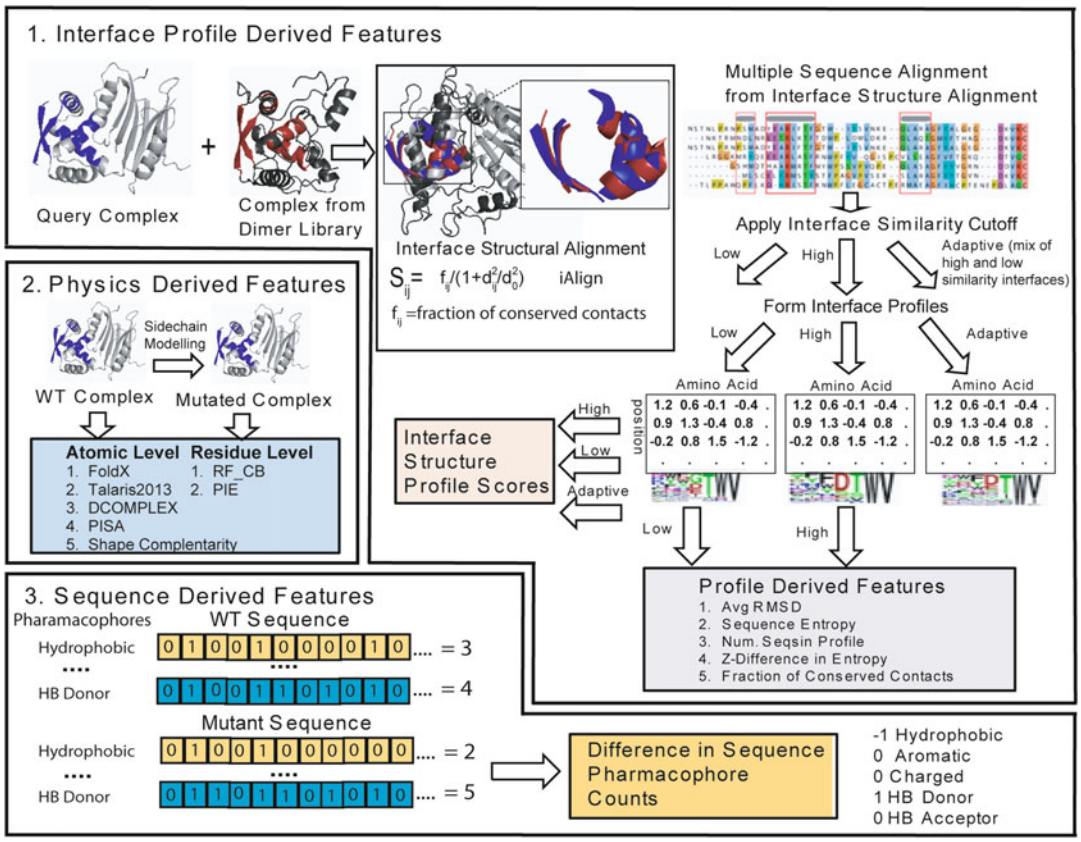


Fig. 2 Multiscale approach to predicting protein binding affinity using features derived from interface structural profiles, WT and mutant sequences, and physics-based scoring of the structures of the wild-type and mutant complexes. (*1*) Interface profile scores derived by structural alignment of structurally similar interfaces using an interface similarity cutoff to define the aligned sequences that are used to build the profile. (*2*) Physics-based scores are formed at the residue or atomic level formed by modeling the mutant monomeric protein and complex and evaluating the difference in energy. (*3*) Sequence features are formed by the difference between the WT and mutant sequences in the number of hydrophobic (V, I, L, M, F, W, or C), aromatic (Y, F, or W), charged (R, K, D, or E), hydrogen bond acceptors (D, E, N, H, Q, S, T, or Y), and hydrogen bond donating residues (R, K, W, N, Q, H, S, T, or Y) along with difference in amino acid volume calculated from the sequence

[41]. A series of interface similarity cutoffs has been used to define three separate interface structure profiles along with different metrics designed to assess the accuracy of the profiles relative to the other terms [39]. The interface profiles scores are then combined with physics-based all-atom and residue level docking scores. Finally, sequence-based scores based on phamacophore count differences between the native and designed sequences are calculated to complete the multiscale approach. A random forest method trained to predict the experimental affinity changes ($\Delta\Delta G$) associated with single and multiple mutations at the interface is used for the final interface energy score. This energy score has a correlation to the experimental $\Delta\Delta G$ values equivalent or superior to the best state-of-the-art mutation prediction programs (Pearson's correlation coefficient = 0.83 for a 5 fold cross validated set) but is fast enough to calculate the thousands of potential mutations necessary for protein design. The interface energy is then added to the regular EvoDesign scoring potential, using a user-defined weighting function to balance fold stability and protein-protein affinity.

2.2 Using the EvoDesign Server Design Program

The EvoDesign program can be used as a server at http://zhanglab.ccmb.med.umich.edu/EvoDesign. The only input to the server is a PDB format file of the target structure, which can be either a full-atomic or backbone only model. In either case, the backbone of the protein structure should be complete without breaks in the chain. Currently, the server is limited to design of one protein chain only.

There are three user-defined parameters to control the design simulation. The first parameter is the fold-similarity cutoff used for defining the structural profile (Eq. 1). By default, this is set to the relatively high value of a TM score of 0.7, which is relaxed if less than ten structural analogues are found in the PDB. This value can be adjusted to a higher or lower value; lower values incorporate more sequence and structural variability in constructing the profile while higher values incorporate less. The usual result is that higher cutoffs penalize deviations from the native sequence more strongly, which may or may not be desirable for the particular application. The second parameter controls whether the FoldX force field is used in the simulation or not. Inclusion of FoldX usually results in only a marginal improvement in the folding when validated by structure prediction (see the next section) [25], most likely due to the fact that the side-chains are modeled by a different force field from the SCWRL force field used for scoring. Including FoldX in the simulation requires that the full atomic model of each sequence be constructed, which is the most computationally demanding step in the simulation. For this reason, the FoldX force field is turned off by default. The last parameter does not affect the design simulation but controls whether structure prediction is performed for each of the designed sequences through the creation of I-TASSER models (*see* Subheading 2.3.1).

By default, the EvoDesign server operates without any residue restrictions on the design process. In many cases, it desirable to freeze certain residues in the design process, such as those involved in disulfide bond formation or in ligand binding. Taken further, in other cases, it is useful to redesign only the surface of the protein while keeping the inner core constant. An option is therefore provided to specify a set of residues (by residue number) which should be kept the same as in the input structure. It is also sometimes desirable to restrict the use of some residues completely or at certain positions. A prime example is cysteine residues on the surface, which can easily be oxidized to form intermolecular disulfide bonds that lead to a loss of activity through aggregation.

The output of the server is ten sequences in decreasing order of cluster size from the clusters generated by the SPICKER algorithm. For each sequence, the sequence identity to the native sequence is calculated along with the predicted normalized relative error for the secondary structure, solvent accessibility, and torsion angles. Each property is calculated by a high accuracy predictor using PSI-BLAST profiles along with neural network predictors (PSSPred for secondary structure prediction [42], ANGLOR for torsion angle prediction [32], and the method of SOLVE for solvent accessibility [43], respectively). The normalized relative error (NRE) is reported for each prediction, which is defined by [25].

$$\mathrm{NRE} = \frac{\mathrm{EDS} - \mathrm{ETS}}{\mathrm{ETS}}, \tag{4}$$

where *EDS* refers to "error of designed sequence," i.e., the mismatch between the predicted structure feature from the designed sequence and the target structure. *ETS* refers to "error of target sequence" that is defined similarly to *EDS* but for the target sequence. The *NRE* defined thus accounts for the uncertainty from the structure feature predictors. Finally, I-TASSER models of each of the designed sequences are provided if user selects the third option on I-TASSER modeling. The I-TASSER models represent a partial validation of the success of the design simulation as described below.

2.3 Computational Validation of Protein Designs

No computational design method is perfect, and validation remains an essential part of the design processes. Validating experimentally that the designed protein sequence successfully folds to the desired structure requires both successfully expressing the protein and successfully determining the structure. A full structure determination at the atomic level through either NMR spectroscopy or X-ray crystallography is a time-consuming and difficult task. Even simpler, less precise experimental methods for determining protein structure, such as comparing the secondary structure of the native and designed proteins through circular dichroism CD (*see*

Subheading 2.4.7) and recognition of the presence of folded tertiary structure through 1D NMR (*see* Subheading 2.4.8), still require that the protein be successfully expressed. Compared to computational techniques, protein expression is relatively expensive, limited in throughput, and in some cases may be challenging to achieve. Before expression, it is therefore desirable to know which designed sequences are most likely to fold to the target structure. The first step is to visually confirm that the design sequences are compatible with the structure. Specifically, it is a good idea to look for buried charges without salt-bridges and buried side-chains without hydrogen bonding partners before proceeding. The EvoDesign program uses a fixed backbone approximation in its calculations. High energies from van der Waals clashes can usually be relieved by small changes in the backbone [44, 45]. However, buried charges and missing hydrogen bonds are much harder to compensate for by small structural movements. Since even one missed hydrogen bond or buried charge is enough to completely destabilize a structure, any designs possessing these features should be eliminated from consideration.

It is, however, not possible to tell reliably if a protein will fold correctly by simple visual analysis. Accurate structure prediction of designed sequences is therefore central to the EvoDesign methodology, as it allows a much larger number and variety of sequences to be tested for correct folding than can be experimentally checked. EvoDesign currently employs I-TASSER, which is a hierarchical approach to protein structure modeling that constructs protein 3D models by reassembling continuous fragments excised from the multiple threading templates [43, 46–48]. I-TASSER has been extensively tested in both benchmarking [46, 47, 49] and blind tests [50–53]. In particular, the community-wide CASP (Critical Assessment of protein Structure Prediction) experiment is designed to benchmark the state-of-the-art of protein structure predictions every two years since 1994 [54–56]. I-TASSER was tested (as "Zhang-server") in the 7–11th CASP competitions in 2006–2015. Figure 3 shows the histogram of the *Z*-score of the GDT-score, which measures the significance of the model predictions by each group of automated structure predictors compared to the average performance, in the latest 11th CASP competition. The data shows the advantage of the I-TASSER in comparison to other state-of-the-art protein structure prediction methods, provided that the protein is already known to fold to a specific structure.

2.3.1 Estimating Structural Fidelity and Foldability of Designed Sequences Using I-TASSER

The I-TASSER-based structure prediction of designed sequences in EvoDesign seeks to answer two related but distinct questions. First, does the designed sequence fold to any structure at all or is it only partially or completely unfolded when expressed? Second, given that the protein folds, does it fold to the correct structure? If a designed sequence is known to fold, there is considerable evidence

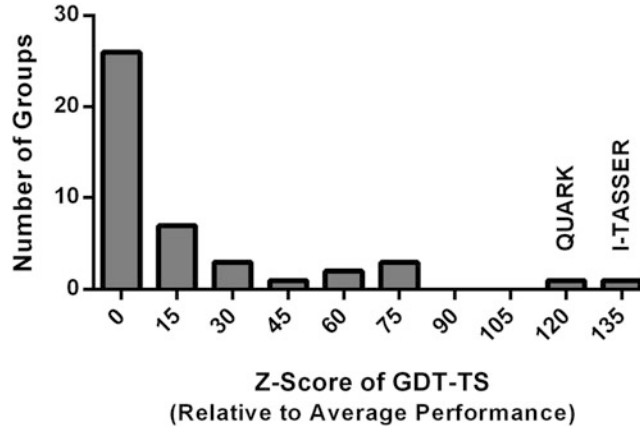


Fig. 3 Histogram of the *Z*-scores of all automated protein structure predictors in the CASP11 experiment. The first bin contains groups that have *Z*-score below 0. Data are taken from official CASP webpage at URL http:/www.predictioncenter.org/casp11/zscores_final.cgi?model_type=first&gr_type=server_only

from the benchmark and blind tests described above that I-TASSER could, with some confidence, tell if it will fold to its target structure. However, the ability of template-based protein structure programs to determine whether or not a given sequence can fold correctly to any structure at all has been tested much less extensively (*see* **Note 1**).

In an early test, I-TASSER was shown to cleanly distinguish native sequences from random sequences with similar sequence identity and secondary structural propensity [38]. For a more stringent benchmark test, we recently tested 16 successfully designed sequences that are known to match their target structure and 29 unsuccessful sequences that were known to either fold to a different structure or were unable to fold at all in the literature [25]. As shown in Fig. 4, I-TASSER successfully captured the deviation of the structures of the designed sequences from the target structure. Furthermore, the confidence level (*C*-score) [57] of the I-TASSER prediction is roughly correlated with the chance of success of the design: a C-score below −1.5 indicates an almost certain failure and a *C*-score above 0 indicates a very strong possibility of success. I-TASSER prediction on designed sequences can therefore allow a winnowing out of poorly designed sequences without resorting to the lengthy procedure of expressing and experimentally determining the structures of designed proteins at each step.

2.4 Experimental Validation of Designed Sequences

True validation of the designed protein requires that protein be characterized experimentally for structurally fidelity and activity. The processes listed below have been employed in the EvoDesign

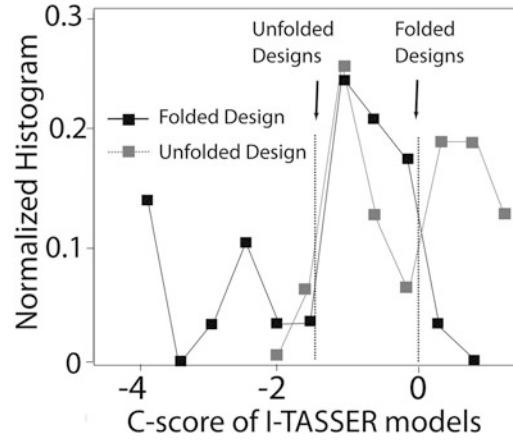


Fig. 4 Divergence in the confidence score of the I-TASSER models for successfully and unsuccessfully designed sequences. Approximate cutoff values are indicated by the arrows. A *C*-score < -1.5 indicates a high probability that the design will not be folded correctly and a *C*-score > 0 indicates a high probability that the design will fold to the target structure

studies [25, 58], aiming to ensure that the designed proteins are thermodynamically stable, soluble, and adopt the desired fold. In all cases, the same tests should be performed with the wild-type protein as well for a control.

2.4.1 Expression and Purification of Designed Proteins

Before a protein can be characterized experimentally, the pure protein must be generated in sufficient quantities for the experiments. This is done through a process called recombinant expression, which involves incorporating the DNA sequence of the designed protein into the genome of another organism and using that organism's protein production process to generate the target protein. Since there are many variations on the technique and the specifics of the process can vary with the protein being produced, a comprehensive description of the technique is not given here. Instead, key considerations are outlined in a basic manner for those unfamiliar with process. For further, more depth treatment readers are encouraged to consult several excellent reviews on this topic [59].

2.4.2 Choice of Host Cell

The first decision that must be made in setting up a recombinant protein expression system is the choice of the host cell whose protein synthesis machinery will produce the target protein. This choice is one of the most critical ones as the choice of the expression organism defines the scope of the project, the reagents and equipment needed, and the final outcome of the expression process [59]. Each protein expression has advantages and disadvantages. In most cases, bacterial expression systems are favored as they are low cost,

easy to manipulate genetically, scale easily from small- to large-scale expression, and can easily incorporate isotopic labels for NMR studies. The main disadvantage of bacterial expression is that eukaryotic posttranslational modifications such as glycosylation and phosphorylation are not performed. In the case that these posttranslational modifications are essential, a eukaryotic host cell such as yeast or insect cells must usually be used and the process becomes considerably more complex.

Disulfide bond formation is also more difficult in bacteria, although this may be overcome in most cases by selecting a bacterial strain such as the Orgami cell line that have mutations in the thioredoxin reductase and glutathione reductase genes, which creates an oxidative environment that greatly enhances disulfide bond formation in the cytoplasm [60]. Expression can vary greatly for different bacterial strains. For this reason, different specialized strains of bacteria have been created to optimize the expression of recombinant proteins. Most specialized bacterial strains for expression start with the BL21 genetic background that is deficient in the Ion and ompT proteases that can lead to improper cleavage of the protein product. Other bacterial strains attempt to minimize the difference in codon usage between the natural codon usage of the bacteria and the codon usage required to express the protein.

Recombinant expression of proteins can lead to a high demand for specific tRNAs that are normally produced in only small amounts by the bacteria. Depletion of these low abundance tRNAs can cause translation to stall on the ribosome, leading to premature release from the ribosome and the generation of truncated versions of the protein [61]. From our studies [25, 58, 62], we recommend for routine use of the Rosetta 2 bacterial cell line that combines the protease mutations found in the BL21 strain along with additional modifications that allow the bacteria to generate low abundance tRNAs more efficiently and mutations that allow tunable expression through mutations in the Lac permease gene (see below). However, alternate strains may be considered in certain situations such as the Rosetta-gami strain, which adds the disulfide-bond promoting mutations of the Orgami strain to the Rosetta background.

2.4.3 Selection of Expression Vector

Once the host cell is selected, the next step is to create the vector that introduces the foreign DNA into host cell. This is typically a bacterial plasmid that contains several elements besides the DNA encoding the target protein. The first element is a gene for antibiotic resistance which provides a growth selection mechanism for discovery; only those bacteria that have incorporated the plasmid into their genome can grow in the presence of the antibiotic. The second is the promoter system, which ties the expression of the target protein to another protein whose expression is essential for

the cell and whose expression can be readily induced at a specific time. Triggering expression at a specific time is essential as bacteria continue to grow during incubation and the time at which the protein is lysed determines the overall yield and final purity of the product. If the cell density is too low, the yield of expressed protein is naturally low. On the other hand, too high of cell density can also result in decreased yields and purity from loss of the plasmid from the bacteria [63], metabolism of the antibiotic within the medium, and death of the bacteria from lack of dissolved oxygen [64]. Typically, this is done through the use of the Lac operon, in which protein expression can be induced at a specific time period during growth with the lactose analog isopropyl β-D-1-thiogalactopyranoside (IPTG).

2.4.4 Purification of Expressed Protein

Once expressed, the expressed protein still needs to be purified from the other proteins in the bacterial cell. Although this may be accomplished using the sequence of the designed protein without modification using multiple steps of column chromatography, it is easier to fuse the designed sequence to other protein domains to make purification easier. In many cases, the expressed protein is not soluble at the very high concentrations generated during expression. In this situation, the expressed protein accumulates in an insoluble form in the bacteria as particles known as inclusion bodies. The formation of inclusion bodies can make purification easier or more difficult. The inclusion bodies generally contain the expressed protein in highly pure form with only a small amount of the other proteins of the host cell mixed in, a clear advantage for the purification process. On the other hand, proteins within inclusion bodies must be first disaggregated and then refolded with urea, which may prove a difficult process [65]. If the stability of the protein is unknown, such as the case with designed proteins, it is often easier to try to purify already folded, soluble proteins.

To enhance the solubility of proteins during purification, a solubility tag such as the Mocr domain [66] can be fused to the target protein. This domain is usually fused N-terminal to the designed sequence. Since it is localized to the N-terminus, the Mocr domain is therefore synthesized first and folds into its native form before the translation of the designed sequence, stabilizing the designed domain's folding process. Moreover, the high negative charge on the Mocr domain increases the solubility during the purification process by preventing self-association by electrostatic repulsion. Along with the solubility tag, another sequence that specifically binds a particular column can be incorporated to assist purification. A common choice is the His tag, six consecutive histidine residues that strongly bind nickel (Ni) columns. A protease cleavage site is often placed between the Mocr domain with the His tag and the sequence of the designed protein so that the two

domains can be separated. The expressed protein with the Mocr/His tag will bind the Ni column; most other bacterial proteins will not. The Mocr/His domain is then cleaved from the target sequence by the addition of a protease specific to the cleavage site and passed through the Ni column again. This time, the target protein does not bind the Ni column but all other nickel-binding proteins will remain bound to the column. The end result of this process is a highly pure protein in a soluble form.

2.4.5 Confirmation of Protein Solubility

In addition to adopting a stable folded conformation, many proteins must be soluble in water to perform their biological function. This requirement constrains the design process, as sequences that are optimized only for stability of the folded conformation may not be optimized for solubility. A key advantage of the EvoDesign method is that the structural profiles implicitly include all the constraints involved in determining the sequences that are compatible with a specific fold, not just those concerned with fold stability. As a result, sequences designed by EvoDesign are significantly more native-like in composition than those designed by physics only methods [25], which tend to overemphasize hydrophobic residues on the surface more than is found in native proteins [20, 38, 67]. Consequently, aggregation by the coalescence of exposed hydrophobic patches is a common source of failure in physics-based design [20].

As aggregation generally makes a protein useless for most applications, the oligomeric state of the protein should be determined before proceeding at the highest concentration used for the other biophysical experiments. Typically, this is around 100 μM for a 100-residue domain. The limiting factor is usually sensitivity of the 1D NMR experiment for tertiary structure estimation and sensitivity of the urea denaturation experiment used for the determination of protein stability (*see* **Note 2**). An approximate concentration range may be established by measuring the signal-to-noise ratio at different concentrations of the native protein. The signal of both experiments is actually more sensitive to the total concentration by weight than the molar concentration. The 100 μM value may need to be adjusted upward or downward for proteins significantly shorter or longer than 100 residues.

The presence of aggregation is most readily determined quantitatively by dynamic light scattering, which measures the hydrodynamic radius of proteins in solution, or from a correctly calibrated analytical size exclusion column. In the absence of either instrument, aggregation may be measured semiquantitatively by the absorbance at 400 nm. At this wavelength range, the protein does not absorb light and increases in absorbance are due to Raleigh scattering, which is proportional to the sixth power of the particle radius. A comparison to the corresponding absorbance at 400 nm

of the native protein provides a qualitative estimate of the amount of aggregation in the sample (*see* **Note 3**).

2.4.6 Confirmation of Structural Fidelity

X-ray crystallography remains the gold standard for confirming whether a protein design has the desired structure. However, not all well-folded proteins crystallize and the expense of X-ray crystallography severely restricts the number of designs that can be studied. From a functional perspective, absolute structural fidelity is not necessary in many cases and small changes on the atomic scale are tolerated if the protein is stable, soluble, and functional. To test a larger number of sequences, faster low-resolution biophysical techniques can be used to eliminate obviously badly designed sequences [68, 69].

2.4.7 Confirmation of Secondary Structure

Secondary structure is the most basic building block of protein structure. The existence of severely incorrect secondary structure in the designed protein therefore very strongly implicates a failed design. Since each secondary structure element (α-helix, β-sheet, and random coil) has a distinct circular dichroism (CD) spectra, the relative fractions of each in a protein can be estimated from a CD spectra by fitting to a reference set of proteins with known CD spectra and secondary structure [70]. The accuracy of this procedure is typically around ±5 %, with α-helical content determined more precisely than either random coil or beta sheet content. If available, infrared (IR) spectra can also be used in a similar manner to characterize the secondary structure, as it has been shown that IR and CD are largely complementary and a combination of the two techniques gives a more accurate picture of the secondary structure than either technique alone [71].

2.4.8 Confirmation of Existence of Tertiary Structure

The existence of tertiary structure has traditionally been defined in a qualitative way from the appearance of the 1D ^{1}H NMR spectra of the protein. A protein that is poorly folded, without extensive contacts within the protein core, has a distinctive 1D NMR spectra characterized by the lack of highly shielded peaks in the region of the spectra from −1 to 0.5 ppm and poor dispersion of the signal within the amide region (*see* Fig. 5) [72, 73]. While this method is standard in the protein design field [68, 69], it is subjective and qualitative. A more objective and quantitative method is to use the autocorrelation of a 1D ^{1}H [74] or unassigned 3D ^{15}N NOESY-HSQC NMR spectrum [75], which have been shown to accurately distinguish folded and unfolded proteins. A comparison of the binding of the dye SYPRO Orange, which binds to exposed hydrophobic surfaces, to the native sequence can provide an additional test for a misfolded protein structure [76].

2.4.9 Confirmation of Fold Stability

The free energy of folding can be measured using chemical denaturation with urea, with denaturation measured by the decrease in secondary structure as determined by CD [25]. As the

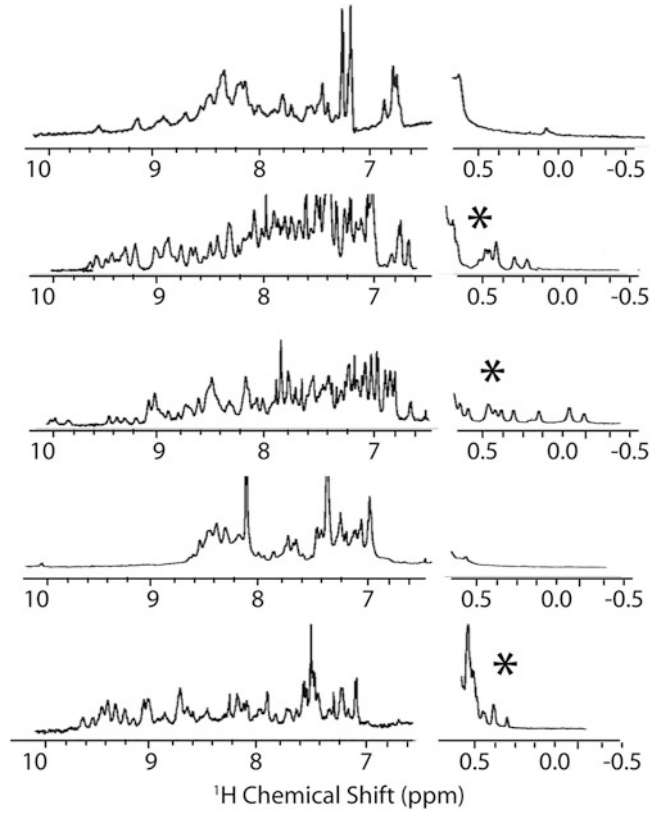


Fig. 5 NMR spectra of folded (with *asterisk*) and unfolded designed proteins. The folded designs have a wider range of chemical shift values in the amide region of the spectrum (7–10 ppm) and have chemical shift values below 0.5 ppm indicating side-chains strongly shielded from solvent, as would be expected in a well-packed protein core

concentration of urea is increased, the protein unfolds, in most cases by a two-step process without a significant population of partially unfolded intermediates. The first step of determining the stability is to measure the CD signal without denaturant (CD_{folded}), where it is assumed to be completely folded, and at a high concentration of denaturant, where it is assumed to be completely folded ($CD_{unfolded}$). If unfolding is a two-step process, the CD signal as a function of the urea concentration is [77]:

$$CD(\text{urea}) = f_{\text{unfolded}}(\text{urea})CD_{\text{unfolded}} + f_{\text{folded}}(\text{urea})CD_{\text{folded}}, \quad (5)$$

where $f_{\text{folded}}(urea)$ and $f_{\text{unfolded}}(urea)$ refer to the fractions of folded and unfolded proteins respectively, at a given urea concentration. Since the equilibrium constant can be calculated directly from fraction of folded and unfolded proteins, the Gibbs free

energy of unfolding can be calculated for each urea concentration [77]:

$$K(\text{urea}) = \frac{f_{\text{unfolded}}(\text{urea})}{1 - f_{\text{unfolded}}(\text{urea})} \tag{6}$$

$$\Delta G(\text{urea}) = -\text{RT ln } K(\text{urea}) = -RT\ln\left(\frac{f_{\text{unfolded}}(\text{urea})}{1 - f_{\text{unfolded}}(\text{urea})}\right) \tag{7}$$

The relevant free energy is the free energy of unfolding in the absence of denaturant, which can be obtained by linear extrapolation of the free energy to zero urea concentration.

3 Conclusions

Using an evolution-based approach, we have successfully designed, expressed, and experimentally characterized a number of single domain proteins [25, 58]. In the first benchmark test, we used EvoDesign to redesign 87 globular proteins randomly collected from the PISCES server. I-TASSER was then used to test the fidelity of the predicted structure to the target. Although all homologous templates have been excluded from the I-TASSER template library, out of the 87 designed sequences, 80 % were predicted to fold to structure with an RMSD of <2.0 Å to the target scaffold, and 42.5 % were predicted to fold to an essentially identical structure with an RMSD < 1.0 Å. This was a clear difference from designed sequences created using only the FoldX force field, for which only 54 % of the predicted structures have an RMSD < 2.0 Å to the target structure, and only 31 % have an RMSD < 1.0 Å.

In a separate test, we redesigned five globular proteins by EvoDesign and used the experimental validation procedures described in Subheading 2.4 to confirm the success of the designs. All five proteins were successfully expressed using the expression system in Subheading 2.4.3 and were soluble to at least 70 μM. Further, all five designed proteins have secondary structure consistent with the target protein (<12 % difference). Three out of the five had a compact tertiary structure confirmed by NMR (Subheading 2.4.8, Fig. 5), for an overall success rate of 60 %. One of the three, the Phox homology domain of the cytokine-independent survival kinase (CISK-PX), could be crystallized and its structure compared to the native protein [78]. Despite having only 32 % sequence identity, the structure of the designed protein showed a very close similarity to the target with a RMSD of 1.54 Å and a TM score of 0.90 to the target template. The RMSD and TM score between the I-TASSER model and the X-ray crystal structure of

CISK-PX are 1.32 Å and 0.91, respectively. Most of the difference between the two structures was in a loop that is disordered in the original structure.

Finally, we have shown that EvoDesign can be used to create functional complexes for the X-linked inhibitor of apoptosis proteins (XIAP) with improved properties by designing a peptide-protein complex involved in apoptosis inhibition [58]. The XIAP protein inhibits apotosis by binding caspase-9, an activity that is in turn regulated by the second mitochondria-derived activator of caspases (SMAC). The designed XIAP protein by EvoDesign binds SMAC but does not possess affinity for caspase-9. As such, the designed protein can serve as a SMAC sink, altering the normal protein-protein interaction network involved in cell death. The circular dichroism and isothermal calorimetry data showed that the designed XIAP domain was more stable than WT-XIAP and bound the SMAC derived peptide with a K_d of 167 ± 67 nM, which compares favorably with the 80 ± 25 nM K_d found for WT-XIAP. Interestingly, a designed version of XIAP with native interface residues actually showed worse binding (K_d of 352 ± 79 nM) and stability than the fully designed sequence, highlighting the efficiency of evolution-based full protein design.

4 Notes

1. The distinction between these two questions becomes clear when the nature of the benchmarks is considered. Due to the experimental requirements of structure determination, the benchmark test largely consists of proteins that can be successfully expressed, successfully purified, and are stable for a prolonged period of time at high concentration. In addition, the protein also must be crystallized in the case of X-ray structures, which is a rather severe restriction for proteins with large unfolded regions as the disordered regions have poor crystal contacts which interferes with the crystallization process [79]. Even if the protein can be crystallized, the disordered regions will have poor electron density and will therefore not be resolved in the structure. Similarly, the structure of unfolded proteins is difficult to determine by NMR due to the lack of long-range NOE constraints and poor chemical shift dispersion [80]. These experimental constraints suggest that though the PDB library is largely complete with respect to the possible universe of monomeric folded domains [81, 82], it is still biased toward compact folded structures, as proteins that are intrinsically unstable or unfolded are difficult to observe. The PDB library should therefore not be considered as completely representative of the conformational ensembles, folded or not, that all protein sequences can adopt.

2. The signal-to-noise ratio in an NMR experiment depends on a number of factors including the field strength of the NMR spectrometer (higher magnetic fields give higher resolution spectra and hence higher signal-to-noise ratios), the size of the protein (larger proteins give rise to broader signals), and other factors such as conformational exchange (transitions between conformations under certain timescales give rise broader signals). The signal-to-noise ratio in a CD spectrum also depends on a variety of factors, including the transparency of the buffer in the far UV region of the spectrum (180–260 nm), the pathlength of the cuvette, and the age of the xenon lamp used to acquire the spectrum. Of these factors, the transparency of the buffer usually has the most impact. A buffer strongly absorbing in the UV serves as an inner filter that attenuates the incoming light reaching the protein. Phosphate buffers are optimal for CD due to their transparency in the far UV region of the spectrum, although Tris buffers are nearly as good. Chloride ions absorb in this region and the proteins in NaCl solutions should be dialyzed against an equivalent of concentration of NaF. Finally, many additives used to stabilize proteins, such as glycerol, arginine, and Triton-X, absorb strongly in the UV and are incompatible with CD spectroscopy for this reason.
3. An alternative wavelength can be used if the protein possesses a cofactor such as FAD or FMN that absorbs in the visible light range.

Acknowledgment

The project is supported in part by the National Institute of General Medical Sciences (GM083107).

References

1. Karanicolas J, Kuhlman B (2009) Computational design of affinity and specificity at protein-protein interfaces. Curr Opin Struct Biol 19(4):458–463
2. Kortemme T, Joachimiak LA, Bullock AN, Schuler AD, Stoddard BL, Baker D (2004) Computational redesign of protein-protein interaction specificity. Nat Struct Mol Biol 11 (4):371–379
3. Shifman JM, Mayo SL (2003) Exploring the origins of binding specificity through the computational redesign of calmodulin. Proc Natl Acad Sci U S A 100(23):13274–13279
4. Lopes A, Busch MSA, Simonson T (2010) Computational design of protein-ligand binding: modifying the specificity of asparaginyl-tRNA synthetase. J Comput Chem 31 (6):1273–1286
5. Procko E, Hedman R, Hamilton K, Seetharaman J, Fleishman SJ, Su M, Aramini J, Kornhaber G, Hunt JF, Tong L, Montelione GT, Baker D (2013) Computational design of a protein-based enzyme inhibitor. J Mol Biol 425(18):3563–3575
6. Jiang L, Althoff EA, Clemente FR, Doyle L, Rothlisberger D, Zanghellini A, Gallaher JL, Betker JL, Tanaka F, Barbas CF 3rd, Hilvert D, Houk KN, Stoddard BL, Baker D (2008) De novo computational design of retro-aldol enzymes. Science 319(5868):1387–1391
7. Kuhlman B, Dantas G, Ireton GC, Varani G, Stoddard BL, Baker D (2003) Design of a

novel globular protein fold with atomic-level accuracy. Science 302(5649):1364–1368

8. Siegel JB, Smith AL, Poust S, Wargacki AJ, Bar-Even A, Louw C, Shen BW, Eiben CB, Tran HM, Noor E, Gallaher JL, Bale J, Yoshikuni Y, Gelb MH, Keasling JD, Stoddard BL, Lidstrom ME, Baker D (2015) Computational protein design enables a novel one-carbon assimilation pathway. Proc Natl Acad Sci U S A 112(12):3704–3709
9. Ollikainen N, Kortemme T (2013) Computational protein design quantifies structural constraints on amino acid covariation. PLoS Comput Biol 9(11), e1003313
10. Fromer M, Linial M (2010) Exposing the co-adaptive potential of protein-protein interfaces through computational sequence design. Bioinformatics 26(18):2266–2272
11. McLaughlin RN, Poelwijk FJ, Raman A, Gosal WS, Ranganathan R (2012) The spatial architecture of protein function and adaptation. Nature 491(7422):138–142
12. Schaefer C, Schlessinger A, Rost B (2010) Protein secondary structure appears to be robust under in silico evolution while protein disorder appears not to be. Bioinformatics 26 (5):625–631
13. Ollikainen N, Smith CA, Fraser JS, Kortemme T (2013) Flexible backbone sampling methods to model and design protein alternative conformations. Methods Enzymol 523:61–85
14. Kellogg EH, Leaver-Fay A, Baker D (2011) Role of conformational sampling in computing mutation-induced changes in protein structure and stability. Proteins 79(3):830–838
15. Chiti F, Stefani M, Taddei N, Ramponi G, Dobson CM (2003) Rationalization of the effects of mutations on peptide and protein aggregation rates. Nature 424(6950):805–808
16. Smith CA, Kortemme T (2011) Predicting the tolerated sequences for proteins and protein interfaces using RosettaBackrub flexible backbone design. PLoS One 6(7)
17. Harbury PB, Plecs JJ, Tidor B, Alber T, Kim PS (1998) High-resolution protein design with backbone freedom. Science 282 (5393):1462–1467
18. Pokala N, Handel TM (2005) Energy functions for protein design: adjustment with protein-protein complex affinities, models for the unfolded state, and negative design of solubility and specificity. J Mol Biol 347 (1):203–227
19. Li Z, Yang Y, Zhan J, Dai L, Zhou Y (2013) Energy functions in de novo protein design: current challenges and future prospects. Annu Rev Biophys 42:315–335
20. Jacak R, Leaver-Fay A, Kuhlman B (2012) Computational protein design with explicit consideration of surface hydrophobic patches. Proteins 80(3):825–838
21. Bowie JU, Luthy R, Eisenberg D (1991) A method to identify protein sequences that fold into a known three-dimensional structure. Science 253(5016):164–170
22. Soding J (2005) Protein homology detection by HMM-HMM comparison. Bioinformatics 21(7):951–960
23. Wu S, Zhang Y (2008) MUSTER: improving protein sequence profile-profile alignments by using multiple sources of structure information. Proteins 72(2):547–556
24. Zhang Y (2008) Progress and challenges in protein structure prediction. Curr Opin Struct Biol 18(3):342–348
25. Mitra P, Shultis D, Brender JR, Czajka J, Marsh D, Gray F, Cierpicki T, Zhang Y (2013) An evolution-based approach to de novo protein design and case study on Mycobacterium tuberculosis. PLoS Comput Biol 9(10), e1003298
26. Mitra P, Shultis D, Zhang Y (2013) EvoDesign: de novo protein design based on structural and evolutionary profiles. Nucleic Acids Res 41(W1):W273–W280
27. Zhang Y, Skolnick J (2005) TM-align: a protein structure alignment algorithm based on the TM-score. Nucleic Acids Res 33 (7):2302–2309
28. Xu J, Zhang Y (2010) How significant is a protein structure similarity with TM-score = 0.5? Bioinformatics 26(7):889–895
29. Gribskov M, Homyak M, Edenfield J, Eisenberg D (1988) Profile scanning for 3-dimensional structural patterns in protein sequences. Comput Appl Biosci 4(1):61–66
30. Gribskov M, Mclachlan AD, Eisenberg D (1987) Profile analysis – detection of distantly related proteins. Proc Natl Acad Sci U S A 84 (13):4355–4358
31. Henikoff S, Henikoff JG (1992) Amino acid substitution matrices from protein blocks. Proc Natl Acad Sci U S A 89(22):10915–10919
32. Wu ST, Zhang Y (2008) ANGLOR: a composite machine-learning algorithm for protein backbone torsion angle prediction. PLoS One 3(10)
33. Chen HL, Zhou HX (2005) Prediction of solvent accessibility and sites of deleterious mutations from protein sequence. Nucleic Acids Res 33(10):3193–3199
34. Faraggi E, Zhang T, Yang YD, Kurgan L, Zhou YQ (2012) SPINE X: improving protein secondary structure prediction by multistep

learning coupled with prediction of solvent accessible surface area and backbone torsion angles. J Comput Chem 33(3):259–267
35. Schymkowitz J, Borg J, Stricher F, Nys R, Rousseau F, Serrano L (2005) The FoldX web server: an online force field. Nucleic Acids Res 33(Web Server issue):382–388
36. Krivov GG, Shapovalov MV, Dunbrack RL (2009) Improved prediction of protein side-chain conformations with SCWRL4. Proteins 77(4):778–795
37. Zhang Y, Skolnick J (2004) SPICKER: a clustering approach to identify near-native protein folds. J Comput Chem 25(6):865–871
38. Bazzoli A, Tettamanzi AGB, Zhang Y (2011) Computational protein design and large-scale assessment by I-TASSER structure assembly simulations. J Mol Biol 407(5):764–776
39. Brender JR, Zhang Y (2015) Recognizing mutations on protein-protein binding interactions through structure-based interface profiles. PLoS Comput Biol (in press)
40. Mukherjee S, Zhang Y (2011) Protein-protein complex structure predictions by multimeric threading and template recombination. Structure 19(7):955–966
41. Gao M, Skolnick J (2010) iAlign: a method for the structural comparison of protein-protein interfaces. Bioinformatics 26(18):2259–2265
42. Zhang Y (2012) http://zhanglab.ccmb.med.umich.edu/PSSpred
43. Yang J, Yan R, Roy A, Xu D, Poisson J, Zhang Y (2015) The I-TASSER suite: protein structure and function prediction. Nat Methods 12 (1):7–8
44. Davis IW, Arendall WB, Richardson DC, Richardson JS (2006) The backrub motion: how protein backbone shrugs when a sidechain dances. Structure 14(2):265–274
45. Smith CA, Kortemme T (2008) Backrub-like backbone simulation recapitulates natural protein conformational variability and improves mutant side-chain prediction. J Mol Biol 380 (4):742–756
46. Roy A, Kucukural A, Zhang Y (2010) I-TASSER: a unified platform for automated protein structure and function prediction. Nat Protoc 5(4):725–738
47. Wu S, Skolnick J, Zhang Y (2007) Ab initio modeling of small proteins by iterative TASSER simulations. BMC Biol 5:17
48. Zhang Y (2007) Template-based modeling and free modeling by I-TASSER in CASP7. Proteins 69(S8):108–117
49. Zhang Y (2008) I-TASSER server for protein 3D structure prediction. BMC Bioinformatics 9:40
50. Kopp J, Bordoli L, Battey JN, Kiefer F, Schwede T (2007) Assessment of CASP7 predictions for template-based modeling targets. Proteins 69(Suppl 8):38–56
51. Cozzetto D, Kryshtafovych A, Fidelis K, Moult J, Rost B, Tramontano A (2009) Evaluation of template-based models in CASP8 with standard measures. Proteins 77(Suppl 9):18–28
52. Montelione GT (2012) Template based modeling assessment in CASP10. Paper presented at the 10th community wide experiment on the critical assessment of techniques for protein structure prediction, Gaeta, Italy, 9–12 Dec 2012
53. Lee BK (2012) Template free modeling assessment in CASP10. Paper presented at the 10th community wide experiment on the critical assessment of techniques for protein structure prediction, Gaeta, Italy
54. Moult J, Pedersen JT, Judson R, Fidelis K (1995) A large-scale experiment to assess protein structure prediction methods. Proteins 23 (3):2–5
55. Moult J, Fidelis K, Kryshtafovych A, Rost B, Tramontano A (2009) Critical assessment of methods of protein structure prediction-round VIII. Proteins Struct Funct Bioinf 77:1–4
56. Moult J (2005) A decade of CASP: progress, bottlenecks and prognosis in protein structure prediction. Curr Opin Struct Biol 15 (3):285–289
57. Zhang Y (2008) I-TASSER server for protein 3D structure prediction. BMC Bioinformatics 9
58. Shultis D, Mitra P, Aslam N, Gray F, Piper C, Chinnaswamy K, Stuckey J, Cierpicki T, Wang S, Lei M, Zhang Y (2015) Redesigning the fold and binding specificity of BIR3 domain of X-linked inhibitor of apoptosis proteins using evolutionary profiles (submitted)
59. Rosano GL, Ceccarelli EA (2014) Recombinant protein expression in Escherichia coli: advances and challenges. Front Microbiol 5:172
60. Prinz WA, Aslund F, Holmgren A, Beckwith J (1997) The role of the thioredoxin and glutaredoxin pathways in reducing protein disulfide bonds in the Escherichia coli cytoplasm. J Biol Chem 272(25):15661–15667
61. Buchan JR, Stansfield I (2007) Halting a cellular production line: responses to ribosomal pausing during translation. Biol Cell 99 (9):475–487
62. Shultis D, Czajka J, Marsh D, Gray F, Brender JR, Mitra P, Cierpicki T, Zhang Y. Structural validation of computational protein designed through evolutionary methods (in preparation)

63. Baneyx F (1999) Recombinant protein expression in Escherichia coli. Curr Opin Biotechnol 10(5):411–421
64. Jana S, Deb JK (2005) Strategies for efficient production of heterologous proteins in Escherichia coli. Appl Microbiol Biotechnol 67 (3):289–298
65. Burgess RR (2009) Refolding solubilized inclusion body proteins. Methods Enzymol 463:259–282
66. DelProposto J, Majmudar CY, Smith JL, Brown WC (2009) Mocr: a novel fusion tag for enhancing solubility that is compatible with structural biology applications. Protein Expr Purif 63(1):40–49
67. Dantas G, Kuhlman B, Callender D, Wong M, Baker D (2003) A large scale test of computational protein design: folding and stability of nine completely redesigned globular proteins. J Mol Biol 332(2):449–460
68. Koga N, Tatsumi-Koga R, Liu GH, Xiao R, Acton TB, Montelione GT, Baker D (2012) Principles for designing ideal protein structures. Nature 491(7423):222
69. Socolich M, Lockless SW, Russ WP, Lee H, Gardner KH, Ranganathan R (2005) Evolutionary information for specifying a protein fold. Nature 437(7058):512–518
70. Sreerama N, Woody RW (2000) Analysis of protein CD spectra: comparison of CONTIN, SELCON3, and CDSSTR methods in CDPro software. Biophys J 78(1):334
71. Oberg KA, Ruysschaert JM, Goormaghtigh E (2004) The optimization of protein secondary structure determination with infrared and circular dichroism spectra. Eur J Biochem 271 (14):2937–2948
72. Rehm T, Huber R, Holak TA (2002) Application of NMR in structural proteomics: screening for proteins amenable to structural analysis. Structure 10(12):1613–1618
73. Scheich C, Leitner D, Sievert V, Leidert M, Schlegel B, Simon B, Letunic I, Bussow K, Diehl A (2004) Fast identification of folded human protein domains expressed in *E. coli* suitable for structural analysis. BMC Struct Biol 4:4
74. Hoffmann B, Eichmuller C, Steinhauser O, Konrat R (2005) Rapid assessment of protein structural stability and fold validation via NMR. Methods Enzymol 394:142
75. Schedlbauer A, Coudevylle N, Auer R, Kloiber K, Tollinger M, Konrat R (2009) Autocorrelation analysis of NOESY data provides residue compactness for folded and unfolded proteins. J Am Chem Soc 131(17):6038
76. Niesen FH, Berglund H, Vedadi M (2007) The use of differential scanning fluorimetry to detect ligand interactions that promote protein stability. Nat Protoc 2(9):2212–2221
77. Pace CN, Scholtz JM (1997) Measuring the conformational stability of a protein. In: Creighton TE (ed) Protein structure: a practical approach. Oxford University Press, New York, NY, pp 299–321
78. Shultis D, Dodge G, Zhang Y (2015) Crystal structure of designed PX domain from cytokine-independent survival kinase and implications on evolution-based protein engineering (submitted)
79. Price WN 2nd, Chen Y, Handelman SK, Neely H, Manor P, Karlin R, Nair R, Liu J, Baran M, Everett J, Tong SN, Forouhar F, Swaminathan SS, Acton T, Xiao R, Luft JR, Lauricella A, DeTitta GT, Rost B, Montelione GT, Hunt JF (2009) Understanding the physical properties that control protein crystallization by analysis of large-scale experimental data. Nat Biotechnol 27(1):51–57
80. O'Hare B, Benesi AJ, Showalter SA (2009) Incorporating 1H chemical shift determination into 13C-direct detected spectroscopy of intrinsically disordered proteins in solution. J Magn Reson 200(2):354–358
81. Zhang Y, Hubner IA, Arakaki AK, Shakhnovich E, Skolnick J (2006) On the origin and highly likely completeness of single-domain protein structures. Proc Natl Acad Sci U S A 103(8):2605–2610
82. Brylinski M, Gao M, Skolnick J (2011) Why not consider a spherical protein? Implications of backbone hydrogen bonding for protein structure and function. Phys Chem Chem Phys 13(38):17044–17055

Chapter 13

Parallel Computational Protein Design

Yichao Zhou, Bruce R. Donald, and Jianyang Zeng

Abstract

Computational structure-based protein design (CSPD) is an important problem in computational biology, which aims to design or improve a prescribed protein function based on a protein structure template. It provides a practical tool for real-world protein engineering applications. A popular CSPD method that guarantees to find the global minimum energy solution (GMEC) is to combine both dead-end elimination (DEE) and A* tree search algorithms. However, in this framework, the A* search algorithm can run in exponential time in the worst case, which may become the computation bottleneck of large-scale computational protein design process. To address this issue, we extend and add a new module to the OSPREY program that was previously developed in the Donald lab (Gainza et al., Methods Enzymol 523:87, 2013) to implement a GPU-based massively parallel A* algorithm for improving protein design pipeline. By exploiting the modern GPU computational framework and optimizing the computation of the heuristic function for A* search, our new program, called gOSPREY, can provide up to four orders of magnitude speedups in large protein design cases with a small memory overhead comparing to the traditional A* search algorithm implementation, while still guaranteeing the optimality. In addition, gOSPREY can be configured to run in a bounded-memory mode to tackle the problems in which the conformation space is too large and the global optimal solution cannot be computed previously. Furthermore, the GPU-based A* algorithm implemented in the gOSPREY program can be combined with the state-of-the-art rotamer pruning algorithms such as iMinDEE (Gainza et al., PLoS Comput Biol 8:e1002335, 2012) and DEEPer (Hallen et al., Proteins 81:18–39, 2013) to also consider continuous backbone and side-chain flexibility.

Key words Protein design, A*, Dead-end elimination, GPGPU, CUDA, Parallel computing

1 Introduction

1.1 Structure-Based Computational Protein Design

Computational structure-based protein design (CSPD) is an important task in computational biology. In this problem, we want to find new amino acid sequences that have the prerequisite features to perform certain desired functions by substituting a number of residues from a wild-type protein structure with new amino acids. CSPD has many exciting real-world applications in protein engineering, such as design of protein–protein interactions [1], drug design [2], drug resistance prediction [3, 4], vaccine development [5, 6], and enzyme synthesis [7].

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_13,

The CSPD problem can be formulated into finding a sequence of side-chain conformations based on a given energy function so that the energy of the resulting protein structure is minimized. Such an optimal solution is often called the *global minimum energy conformation (GMEC)*. In an ideal case, we hope to consider all possible backbone positions and continuous side-chain conformations for searching the GMEC solution. However, it is almost impossible to sample over all these parameters with high precision because of the huge computational burden. Therefore, simplified protein design models with reasonable assumptions are often used. In practice, we often ignore the displacement of the backbone structure to assume a rigid backbone, and limit the rotational degrees of freedom of side-chain conformations to a set of common discrete conformations, called *rotamer library*.

Having the rigid backbone structure and discrete side-chain conformation assumptions, the protein design problem can be formulated into a combinatorial optimization problem. Equation 1 defines the objective function of this problem:

$$E_{\mathrm{T}}(A) = E_0 + \sum_{i_{\mathrm{r}} \in A} E_1(i_{\mathrm{r}}) + \frac{1}{2} \sum_{i_{\mathrm{r}} \in A} \sum_{j_{\mathrm{s}} \in A} E_2(i_{\mathrm{r}}, j_{\mathrm{s}}), \qquad (1)$$

where A represents a conformation in the search space, i.e., a set of discrete side-chain rotamers of all residues, $E_{\mathrm{T}}(A)$ represents the total energy of conformation A, E_0 represents the backbone energy, $E_1(i_{\mathrm{r}})$ represents the self-energy term of residue i_{r} which is the sum of its intra-energy and residue-to-backbone energy, and $E_2(i_{\mathrm{r}}, j_{\mathrm{s}})$ represents the pairwise energy between rotamer i_{r} and j_{s}.

Unfortunately, even under the rigid backbone and discrete side-chain conformation assumptions, finding the GMEC solution has still been proven as an NP-hard problem [8, 9], which means that most likely there does not exists an algorithm that can guarantee to solve it in polynomial time. The solutions to this issue can be divided into two categories. One common scheme is to apply heuristic algorithms in hopes of generating high-quality solutions [10–13]. The weakness of this scheme is that these algorithms often provide no guarantee of solution quality, as they may be trapped into local optima.

The alternatives to the heuristic algorithms are the provable algorithms, which can assure to output the GMEC solution. Examples are integer linear programming [14], branch-and-bound [15, 16], tree decomposition [17], dead-end elimination [18, 19], and A* tree search [20–22]. Among them, the combination of dead-end elimination and A* tree search has been popularly used to solve the design problem [23]. The major advantage of this pipeline is that it not only guarantees to find the global minimum energy conformation solution, but also is to output all the suboptimal solutions in a gap-free sorted order in an efficient way. This is an

important feature because suboptimal solutions are necessary to fight against the errors in the energy functions and from the model assumptions (e.g., rigid backbone structure) for real-world applications.

In our protein design pipeline, a set of predefined dead-end elimination criteria [24, 25] is first applied to prune all the rotamers that can be proved not to be in the GMEC solution. After that, A* tree search algorithm is applied to traverse the remaining conformation space to find the GMEC solution (and other suboptimal solutions within a given energy cutoff from the GMEC solution). Although A* search guided by an admissible heuristic function usually only needs to visit a small portion of search space to find the optimal solution, it can still run in exponential time in the worst-case scenario due to the difficulty of this problem. Our tool `gOSPREY` provides an effective method to address this computational bottleneck by fully exploiting the massively parallel computational power on the GPU platform to accelerate the computational protein design process.

1.2 General-Purpose Computing on a Graphic Processing Unit

In recent years, the general-purpose computing on a graphic processing unit is becoming popular in numerous scientific computation scenarios. The main difference between a traditional CPU computational framework and a GPU computation framework lies in the way they deal with a computational task. CPUs are often optimized to efficiently execute the input instructions one by one with little parallelism, while GPUs are designed to process thousands of similar tasks simultaneously in an efficient fashion.

Because of such design difference, each core of a GPU is simpler and more efficient than that of a CPU. Therefore, the overall throughput of a GPU platform can be much higher than a CPU platform if the intrinsic parallelism has been fully exploited [26, 27]. Furthermore, a GPU platform usually has its own memory system with high memory bandwidth that is independent of the normal memory system used by the CPU platform. Such a design scheme may require data to be transferred back and forth between CPU and GPU through a slow interface called the PCI-E bus. Our algorithm implemented in `gOSPREY` is a pure GPU algorithm, thus we only need to transfer the initial input data and the final output results between CPU and GPU, in which the data transferring time is generally a negligible overhead compared to the large amount of time spent in the floating-point arithmetic operations require by A* search.

2 Materials

2.1 Hardware

gOSPREY requires a NVIDIA's CUDA-capable GPU to enable its GPU acceleration feature. The minimum requirement of the CUDA compute-capability of the GPU is version 1.2.

2.2 Software

Currently gOSPREY depends on the following software environment:

- Linux operation system.
- NVIDIA CUDA SDK.
- Java Development Kit 1.7.
- gcc-4.7 or newer.
- CMake 2.8 or newer.

2.3 Installation

In order to install gOSPREY, the user needs to enter the working directory and execute the following commands:

```
$ git clone https://github.com/zhou13/gOSPREY.git
$ cd gOSPREY
$ mkdir build
$ cd build
$ cmake -DCMAKE_INSTALL_PREFIX=/usr ..
$ make
$ sudo make install
```

After that, if everything goes smoothly, the user should be able to find that `osprey.jar` has been generated under the `build` directory and the library `libMSAStar.so` has been installed to the system's library directory.

3 Methods

3.1 The Algorithm

In order to find the global minimum energy conformation (GMEC) solution, we often need to search over a large conformational space. To reduce the search space and speed up the design process, our search scheme follows a popular protein design pipeline in the literature [20]: First, a set of dead-end elimination criteria is applied to prune the rotamers that are provably not part of the optimal solution and thus can significantly reduce the magnitude of the search space. Then, a combinatorial optimization algorithm, namely A* tree search, is used to traverse the remaining conformational space and guarantees to find the GMEC solution.

In the traditional A*-based tree search algorithm for protein design, an A* search tree is visited for searching the global optimal

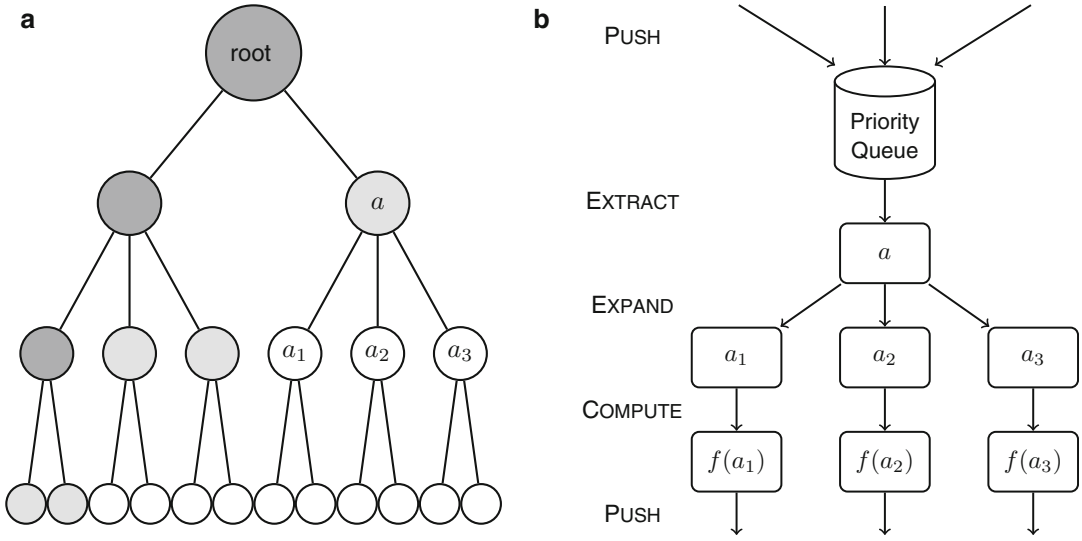


Fig. 1 An illustration of the traditional A* tree search algorithm. (**a**) An example of the A* search tree at a certain time. Nodes in dark shade represent the nodes that have already been expanded and visited. Nodes in light shade represent the frontier nodes which are stored in the priority queue and waiting to be expanded. The node labeled with *a* is the node with minimum heuristic function value in the priority queue and thus is currently being expended. (**b**) The workflow of the node expansion operation for current node *a*

solution. An example of the A* search tree is shown in Fig. 1a. Each internal node of the tree represents a partial conformation in which the side-chain conformations of some residues have not been decided yet, while each leaf node represents a full conformation, in which the side-chain rotamer conformations of all residues have been determined. Thus, computing the global minimum energy conformation is equivalent to finding the leaf node with the minimum energy function value in the last layer.

In order to traverse this search tree efficiently, the A* search algorithm with an admissible heuristic function is usually used [28]. The traditional A* search algorithm uses a priority queue to decide which node should be visited next. In the priority queue, nodes are sorted according to the following heuristic function:

$$f(x) = g(x) + h(x), \tag{2}$$

where $g(x)$ represents the energy term among the residues that have already been decided and $h(x)$ represents the lower bound of the energy term that involves the undecided residues. Intuitively, the heuristic function $f(x)$ provides a quantified estimation about whether the children of node x have low energy values. Therefore, A* search can give higher priority to these nodes with lower $f(x)$ values during the node expansion process. In our protein design problem, the functions of $g(x)$ and $h(x)$ are defined as follows:

$$g(x) = E_0 + \sum_{i \in D(x)} E_1(i_s) + \frac{1}{2} \sum_{i \in D(x)} \sum_{j \in D(x)} E_2(i_r, j_s)$$
$$h(x) = \sum_{i \in U(x)} \min_s \left(E_1(i_s) + \sum_{j \in D(x)} E_2(i_r, j_s) + \sum_{k \in U(x)} \min_u E_2(i_s, k_u) \right), \tag{3}$$

where $D(x)$ represents the set of residues whose side-chain rotamer conformations have already been decided and $U(x)$ represents the set of residues whose side-chain rotamer conformations are still undecided.

In the A* search process, the root node is placed in the priority queue initially. In each round, the A* algorithm extracts the node with the minimum $f(x)$ value, expands its child nodes, computes their heuristic function values, and pushes them back to the priority queue. These steps are repeated until a leaf node is extracted. If the heuristic function is *admissible*, which is the case for Eq. 3, we can prove that the A* search algorithm can find the global optimal solution in our protein design problem [28]. Figure 1b gives an example of the node expansion operation in the traditional A* search algorithm.

From Fig. 1b, we know that the calculation of heuristic function $f(x)$ for each expanded node is simply a series of independent arithmetic operations, which can be directly parallelized on a GPU platform. However, the degree of the parallelism is still limited by the number of children for each node, which is equal to the number of rotamers for each residue in the protein design problem. In general, this number is far smaller than the number of cores in a normal GPU processor, thus the parallelization of the heuristic function calculation alone does not fully exploit the parallelism of a GPU.

To further speedup the search process, `gOSPREY` creates another level of parallelism to exploit the computational power of a GPU platform. Instead of using only one priority queue to perform node expansion, the parallel A* search algorithm allocates hundreds of priority queues in parallel to accelerate the A* search process. Figure 2 provides an example of the parallel node expansion operations in our algorithm.

Our GPU-based A* search algorithm first launches k threads to extract the nodes with the minimum $f(\cdot)$ values from k independent priority queues in parallel, where k is a parameter that can be set by the user. Then each thread expands the child nodes of each extracted node in parallel. After that, the GPU-based A* algorithm launches p threads to compute the heuristic function values for each expanded node, where p is the number of total expanded nodes. Finally, the algorithm launches k threads to push these expanded nodes with the computed $f(\cdot)$ values back to the k priority queues.

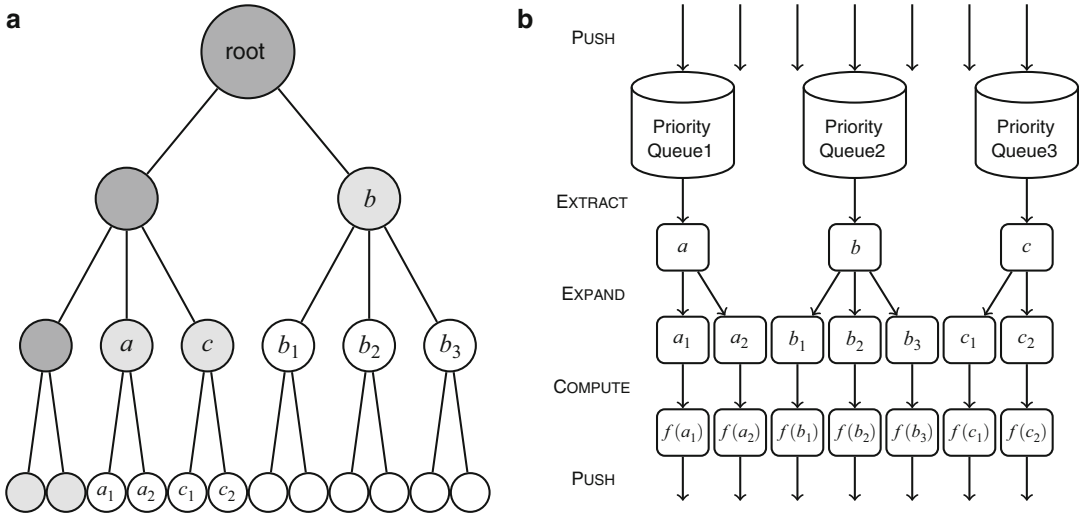


Fig. 2 The illustration of the parallel node expansion operations in our parallel A* search algorithm. (**a**) An example of the A* search tree at a certain time. The meaning of **a** is as same as that of Fig. 1a except that now the algorithm is expanding nodes *a*, *b*, and *c* in parallel. The nodes labeled with *a*, *b*, and *c* are the nodes with the minimum heuristic function values in their respective priority queues. (**b**) The workflow on how multiple nodes are expanded simultaneously

The parameter k can affect the performance of gOSPREY. If we increase the value of k, we can further exploit the parallelism of the GPU platform. However, it may also increase the number of nodes that need to be expanded before finding the global optimal solution, which may lead to an extra memory overhead. In Subheading 3.3, we will provide a simple method to choose a suitable k for individual GPU platforms.

3.2 Performance Evaluation

Here, we cite the test results from [29] to show the performance of our GPU-based protein design algorithm in gOSPREY. In this test, we used the CPU Intel Xeon E5-1620 3.6 GHz with 16GB memory and the GPU Tesla K20c with 2496 CUDA cores and 4.8G global memory. We evaluated both running time and memory usage of gOSPREY on several native sequence recovery problems. Figure 3 shows the comparison results between the traditional single-thread CPU-based A* search algorithm and the massively parallel GPU-based A* search algorithm implemented in gOSPREY. For more details about the comparison results with other protein design frameworks, please refer to the original paper [29].

As shown in Fig. 3, the GPU acceleration achieved by gOSPREY was remarkable. Our GPU-based A* search algorithm was about 40 times faster than the traditional single-thread A* search algorithm on the large design problems. In addition, our benchmark tests showed that the GPU-based A* search had good scalability.

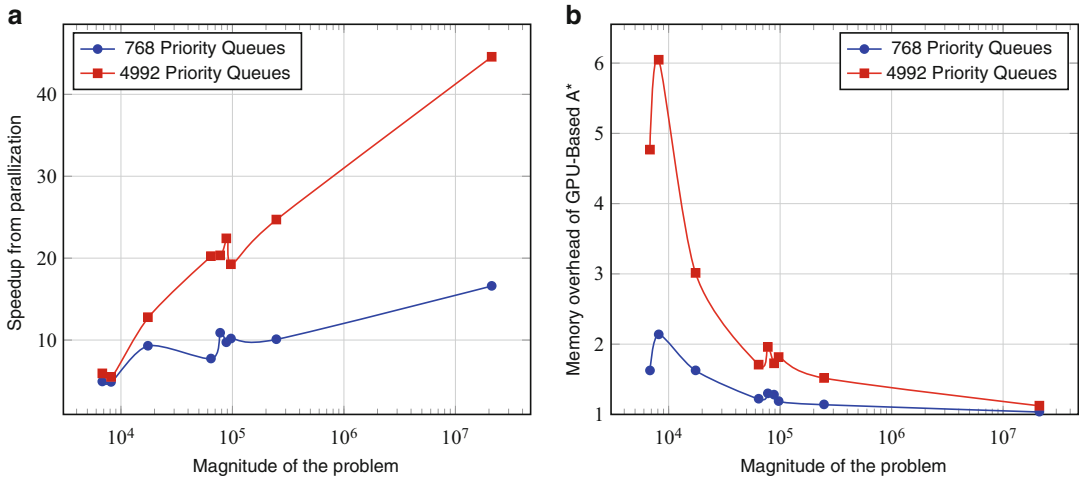


Fig. 3 Semi-log plots about the ratios of speedups and memory overhead of our GPU-based A* search algorithm. The x axes represent the magnitude of the design problem, while the y axes of **a** and **b** represent the ratios of speedups and memory overhead of our GPU-based algorithm comparing to the traditional sequential A* search algorithm, respectively. The circle and square marks represent the results of our GPU-based A* search algorithm with 768 and 4992 parallel priority queues, respectively

The larger the size of the problem, the more speedup that GPU-based A* algorithm can achieve. For small design problems, launching the GPU code and copying the input and output data caused considerable overhead and thus the power of parallelism in the GPU platform was not fully exploited.

The result about the memory overhead of our GPU-based protein design algorithm was also promising. On small problems, the GPU-based A* search algorithm with 4992 priority queues used about six times more memory then the single-thread A* implementation. However, when the magnitude of the problems scaled up, the memory overhead gradually diminished. In the largest problem, the massively parallel GPU-based A* search algorithm only expanded 0.12 times more nodes than the CPU-based algorithm. The significant improvement in running time and the negligible memory overhead shown in this test made the GPU acceleration in `gOSPREY` an appealing and practical tool for solving the large protein design problems.

As shown in Fig. 3b, the GPU-based A* search algorithm with 4992 priority queues expanded more nodes than the A* search algorithm with 768 priority queues. This meant that the number of parallel priority queues should not be set to be too large, as this may result in an unacceptable memory overhead. In the next section, we will describe how to choose the appropriate number of parallel priority queues based on the available hardware environment.

3.3 Usage Examples

As gOSPREY is built based on OSPREY originally developed from the Donald lab [20], most of the configuration of gOSPREY is as same as that of OSPREY. Therefore, in this section we will mainly focus on the feature of gOSPREY, i.e., how to use the GPU acceleration feature of gOSPREY. For the guide and tutorial about how to set up other parameters, please refer to the original manual of OSPREY which is stored in doc/manual.pdf.

1. The first thing that the user needs to do is to know the hardware specification of the GPU system. This can be queried by a simple program called deviceQuery in the NVIDIA CUDA Samples, which is usually installed with the CUDA SDK. The deviceQuery program can be found in $PATH_TO_CUDA_SAMPLE/1_Utilities/deviceQuery. The user may need to compile it manually using the make command. Here is an example of the deviceQuery result:

```
$ ./deviceQuery
deviceQuery Starting...
 CUDA Device Query (Runtime API) version (CUDART static linking)
Detected 1 CUDA Capable device(s)
Device 0: "Tesla K20m"
CUDA Driver Version / Runtime Version 5.5 / 5.5
CUDA Capability Major/Minor version number: 3.5
Total amount of global memory: 5120
MBytes (5368512512 bytes)
  (13) Multiprocessors, (192) CUDA Cores/MP: 2496 CUDA Cores
GPU Clock rate:  706 MHz
(0.71 GHz)
Memory Clock rate:  2600 Mhz
Memory Bus Width:  320-bit
L2 Cache Size:  1310720
bytes
.........
    Concurrent copy and kernel execution: Yes with 2
copy engine(s)
 Run time limit on kernels: No
 Integrated GPU sharing Host Memory: No
 Support host page-locked memory mapping: Yes
 Alignment requirement for Surfaces: Yes
 Device has ECC support:  Disabled
 Device supports Unified Addressing (UVA): Yes
 Device PCI Bus ID / PCI location ID: 2 / 0
 Compute Mode:
< Default (multiple host threads can use ::cudaSetDevice() with
device simultaneously) >
 deviceQuery, CUDA Driver = CUDART, CUDA Driver Version = 5.5,
CUDA Runtime Version = 5.5, NumDevs = 1, Device0 = Tesla K20m
 Result = PASS
```

Here we need to care about:

(a) "(13) Multiprocessors, (192) CUDA Cores/MP" tells us that this GPU has 13 multi-processors, and each of them has 192 CUDA cores. In total, we have 13* 192=2496 CUDA cores.

(b) "Total amount of global memory" tells us the size of the global memory in the GPU.

2. The user also needs to configure the parameter files of `gOSPREY` in order to use the GPU acceleration. An `OSPREY` workspace usually contains three configuration files: `KStar.cfg`, `System.cfg`, and `DEE.cfg`. For GPU acceleration, the user needs to modify `KStar.cfg`. Here is a list of options that are new to `OSPREY` and the user needs to append them to `KStar.cfg`:

```
enableAStarJava false
enableAStarNativeC false
enableAStarCUDA true
maxNativeCPUMemory 5032706048
maxNativeGPUMemory 5032706048
numGPUWorkGroup 26
numGPUWorkItem 192
numGPUWorkItem2 192
shrinkRatio 0.5
```

There are three different A* engines in the `gOSPREY` program: `enableAStarJava` indicates whether the original Java A* engine from `OSPREY` is enabled; `enableAStarNativeC` indicates whether our A* engine with the optimized computation of the heuristic function implemented in C programming language, which should be hundreds of times faster than the original Java A* engine, is enabled; `enableAStarCUDA` indicates whether our GPU-based A* engine is enabled, the performance benchmark of which is shown in Subheading 3.2.

In `KStar.cfg`, `maxNativeCPUMemory` and `maxNativeGPUMemory` set the maximum CPU and GPU memory that `gOSPREY` can occupy, respectively. For `maxNativeGPUMemory`, setting the value to 80 % of the global memory size of the GPU indicated by the `deviceQuery` program is a safe choice.

The parameters `numGPUWorkGroup` and `numGPUWorkItem` together determine the number of parallel priority queues used in `gOSPREY`. In most case, it is reasonable to set `numGPUWorkItem` to be the number of CUDA cores per multi-processor. `numGPUWorkGroup` can be set to be one to two times the number of multiprocessors in the GPU platform. Parameter `numGPUWorkItem2` determines the block size of CUDA when computing the

heuristic functions in parallel. In general, it can be set to be the same value as `numGPUWorkItem`.

Finally, `shrinkRatio` determines the fraction of frontier nodes kept in the memory-bounded A* search. After the memory occupied by the GPU-based A* search algorithm exceeds `maxNativeCPUMemory` or `maxNativeGPUMemory`, `gOSPREY` will discard a percentage of unpromising nodes in the priority queues depending on the value of `shrinkRatio`. Setting this parameter to be less than 1 enables this feature so that A* can continue to run even after the number of expanded nodes exceeds the global memory.

4 Notes

1. Our GPU-based A* search algorithm can also output all the suboptimal solutions within a given energy cutoff from the GMEC solution in a gap-free sorted order, using the same setting as in `OSPREY` [20].
2. `gOSPREY` can be combined with the iMinDEE [18] and DEEPer [30] to further consider continuous backbone and sidechain flexibility, using the same framework as in `OSPREY` [20].
3. In `gOSPREY`, the GPU-based A* search algorithm only performs the single-precision floating-point arithmetic operations and typically runs for a limited period of time. Thus, the features of NVIDIA's expensive video cards Titan and Tesla, such as ECC memory and high-performance double-precision floating-point support, are not so useful to the program. Therefore, if the budget is the main concern, the high-end GeForce GTX video cards are a great choice for `gOSPREY`, and should have comparable or even better performance than the expensive Tesla and GeForce Titan cards.

Acknowledgments

This work is supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61361136003. This work is supported by a grant to B.R.D. from the National Institutes of Health (R01 GM-78031).

References

1. Roberts KE, Cushing PR, Boisguerin P, Madden DR, Donald BR (2012) Computational design of a PDZ domain peptide inhibitor that rescues CFTR activity. PLoS Comput Biol 8(4):e1002477
2. Gorczynski MJ, Grembecka J, Zhou Y, Kong Y, Roudaia L, Douvas MG, Newman M, Bielnicka I, Baber G, Corpora T, Shi J, Sridharan M, Lilien R, Donald BR, Speck NA, Brown ML, Bushweller JH (2007) Allosteric

inhibition of the protein-protein interaction between the leukemia-associated proteins Runx1 and CBF$\\beta$. Chem Biol 14 (10):1186–1197

3. Frey KM, Georgiev I, Donald BR, Anderson AC (2010) Predicting resistance mutations using protein design algorithms. Proc Natl Acad Sci 107(31):13707–13712
4. Reeve SM, Gainza P, Frey KM, Georgiev I, Donald BR, Anderson AC (2015) Protein design algorithms predict viable resistance to an experimental antifolate. Proc Natl Acad Sci 112(3):749–754
5. Reardon PN, Sage H, Dennison SM, Martin JW, Donald BR, Alam SM, Haynes BF, Spicer LD (2014) Structure of an HIV-1-neutralizing antibody target, the lipid-bound gp41 envelope membrane proximal region trimer. Proc Natl Acad Sci 111(4):1391–1396
6. Rudicell RS, Kwon YD, Ko S-Y, Pegu A, Louder MK, Georgiev IS, Wu X, Zhu J, Boyington JC, Chen X, Shi W, Yang ZY, Doria-Rose NA, McKee K, O'Dell S, Schmidt SD, Chuang GY, Druz A, Soto C, Yang Y, Zhang B, Zhou T, Todd JP, Lloyd KE, Eudailey J, Roberts KE, Donald BR, Bailer RT, Ledgerwood J, NISC Comparative Sequencing Program, Mullikin JC, Shapiro L, Koup RA, Graham BS, Nason MC, Connors M, Haynes BF, Rao SS, Roederer M, Kwong PD, Mascola JR, Nabel GJ (2014) Enhanced potency of a broadly neutralizing HIV-1 antibody in vitro improves protection against lentiviral infection in vivo. J Virol 88(21):12669–12682
7. Chen C-Y, Georgiev I, Anderson AC, Donald BR (2009) Computational structure-based redesign of enzyme activity. Proc Natl Acad Sci 106(10):3764–3769
8. Chazelle B, Kingsford C, Singh M (2004) A semidefinite programming approach to side chain positioning with new rounding strategies. INFORMS J Comput 16(4):380–392
9. Pierce NA, Winfree E (2002) Protein design is NP-hard. Protein Eng 15(10):779–782
10. Kuhlman B, Baker D (2000) Native protein sequences are close to optimal for their structures. Proc Natl Acad Sci 97 (19):10383–10388
11. Marvin JS, Hellinga HW (2001) Conversion of a maltose receptor into a zinc biosensor by computational design. Proc Natl Acad Sci 98 (9):4955–4960
12. Shah PS, Hom GK, Mayo SL (2004) Preprocessing of rotamers for protein design calculations. J Comput Chem 25(14):1797–1800
13. Street AG, Mayo SL (1999) Computational protein design. Structure 7(5):R105–R109
14. Kingsford CL, Chazelle B, Singh M (2005) Solving and analyzing side-chain positioning problems using linear and integer programming. Bioinformatics 21(7):1028–1039
15. Hong E-J, Lippow SM, Tidor B, Lozano-Pérez T (2009) Rotamer optimization for protein design through MAP estimation and problem-size reduction. J Comput Chem 30 (12):1923–1945
16. Zhou Y, Wu Y, Zeng J (2015) Computational protein design using AND/OR branch-and-bound search. J Comput Biol 23(6):439–451. doi:10.1089/cmb.2015.0212
17. Xu J, Berger B (2006) Fast and accurate algorithms for protein side-chain packing. JACM) 53(4):533–557
18. Gainza P, Roberts KE, Donald BR (2012) Protein design using continuous rotamers. PLoS Comput Biol 8(1), e1002335
19. Desmet J, Maeyer MD, Hazes B, Lasters I (1992) The dead-end elimination theorem and its use in protein side-chain positioning. Nature 356(6369):539–542
20. Gainza P, Roberts KE, Georgiev I, Lilien RH, Keedy DA, Chen C-Y, Reza F, Anderson AC, Richardson DC, Richardson JS, Donald BR (2013) OSPREY: protein design with ensembles, flexibility, and provable algorithms. Methods Enzymol 523:87
21. Leach AR, Lemon AP (1998) Exploring the conformational space of protein side chains using dead-end elimination and the A* algorithm. Proteins 33(2):227–239
22. Lippow SM, Tidor B (2007) Progress in computational protein design. Curr Opin Biotechnol 18(4):305–311
23. Donald BR (2011) Algorithms in structural molecular biology. The MIT Press, Cambridge, MA
24. Georgiev I, Lilien RH, Donald BR (2006) Improved pruning algorithms and divide-and-conquer strategies for dead-end elimination, with application to protein design. Bioinformatics 22(14):e174–e183
25. Georgiev I, Lilien RH, Donald BR (2008) The minimized dead-end elimination criterion and its application to protein redesign in a hybrid scoring and search algorithm for computing partition functions over molecular ensembles. J Comput Chem 29(10):1527–1542
26. Gregg C, Hazelwood K (2011) Where is the data? Why you cannot debate CPU vs. GPU performance without the answer. In: 2011 I.E. International Symposium on Performance Analysis of Systems and Software (ISPASS), pp. 134–144

27. Lee VW, Kim C, Chhugani J, Deisher M, Kim D, Nguyen AD, Satish N, Smelyanskiy M, Chennupaty S, Hammarlund P, Singhal R, Dubey P (2010) Debunking the 100X GPU vs. CPU myth: an evaluation of throughput computing on CPU and GPU. ACM SIGARCH Comput Archit News 38 (3):451–460
28. Hart PE, Nilsson NJ, Raphael B (1968) A formal basis for the heuristic determination of minimum cost paths. IEEE Trans Syst Sci Cybern 4(2):100–107
29. Zhou Y, Xu W, Donald BR, Zeng J (2014) An efficient parallel algorithm for accelerating computational protein design. Bioinformatics 30(12):i255–i263
30. Hallen MA, Keedy DA, Donald BR (2013) Dead-end elimination with perturbations (DEEPer): a provable protein design algorithm with continuous sidechain and backbone flexibility. Proteins 81(1):18–39

Chapter 14

BindML/BindML+: Detecting Protein-Protein Interaction Interface Propensity from Amino Acid Substitution Patterns

Qing Wei, David La, and Daisuke Kihara

Abstract

Prediction of protein-protein interaction sites in a protein structure provides important information for elucidating the mechanism of protein function and can also be useful in guiding a modeling or design procedures of protein complex structures. Since prediction methods essentially assess the propensity of amino acids that are likely to be part of a protein docking interface, they can help in designing protein-protein interactions. Here, we introduce BindML and BindML+ protein-protein interaction sites prediction methods. BindML predicts protein-protein interaction sites by identifying mutation patterns found in known protein-protein complexes using phylogenetic substitution models. BindML+ is an extension of BindML for distinguishing permanent and transient types of protein-protein interaction sites. We developed an interactive web-server that provides a convenient interface to assist in structural visualization of protein-protein interactions site predictions. The input data for the web-server are a tertiary structure of interest. BindML and BindML+ are available at http://kiharalab.org/bindml/ and http://kiharalab.org/bindml/plus/.

Key words Protein-protein interaction, Protein docking, Interface residues, Protein binding site prediction, Bioinformatics, Protein interaction design, Protein interaction propensity

1 Introduction

Protein-protein interactions (PPIs) are critical for mediating many biological functions in the cell. The plethora of knowledge divulged by the complexity of new PPI networks are continuing to be unraveled [1, 2] and tertiary structures of protein complexes are progressively determined and accumulated in databases [3]. However, the rapid accumulation and availability of sequence and structural data for individual proteins makes computational prediction of PPIs, including protein docking structure prediction [4] and prediction of PPI sites [5–7], invaluable for investigating a large number of interacting proteins that do not have solved structures of complexes. PPI site prediction is useful in guiding protein

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_14, © Springer Science+Business Media New York 2017

docking prediction [8] and for artificially designing of protein-protein interfaces [9].

We have previously developed BindML (Binding site prediction by Maximum Likelihood), a method to predict protein-protein interaction sites using phylogenetic substitution models [10]. BindML takes a protein structure and multiple sequence alignment (MSA) information to predict protein-protein interaction sites of a given protein surface. Protein-protein interaction site is predicted based on amino acid substitutions observed at a local region around a surface amino acid in question. Through a large performance benchmark, we demonstrated that BindML performed favorably against other existing methods.

Furthermore, we developed an extended framework named BindML+ [11], which utilizes mutation patterns specific for permanent and transient interaction sites to distinguish these two types. Proteins interact with each other with different affinities for specific functional reasons. Some protein pairs, for example oligomeric enzyme complex structures, interact tightly and permanently, while other proteins that are involved in signaling pathways have a mechanism for dissociation after binding, which helps to regulate protein activity at specific times (transient interaction). Distinguishing between the two interaction types provides clues for functions of interacting proteins and has important implications for furthering the understanding of the functional diversity exhibited in PPI networks. Being able to distinguish permanent and transient interaction will be the basis for controlling interaction affinity of designing protein interactions.

BindML and BindML+ are unique in that they use solely interaction site specific mutation patterns, i.e., evolutionary information, in comparison with existing methods that consider features of amino acids, including physicochemical properties [12–15], geometric features of surface shape [15, 16]. Our methods are also unique among methods that use a MSA of a query protein to identify structurally or functionally important regions, because most of such methods are based on the traditional principle that important regions of a protein are conserved in its MSA. BindML and BindML+ use mutation patterns observed in a MSA, i.e., regions in a MSA which do not exhibit apparent conservation and identify hidden structures of mutation events in protein sequences. In this sense, BindML and BindML+ are in common in their philosophy with correlated mutation analyses, which are used for predicting physically contacting residues [17–19] or functional residues [20] in proteins.

In this chapter, we present a web-based graphical user interface for BindML and BindML+ that assists in the prediction and structural visualization of protein-protein interactions sites. The web server provides convenient interactive tools to help identify protein-protein interaction site predictions and to intuitively locate

and associate top scoring predictions to an evaluated protein structure. BindML and BindML+ are freely available online as interactive web servers at http://kiharalab.org/bindml/ and http://kiharalab.org/bindml/plus/.

2 Algorithms

In this section, we briefly explain the essence of the BindML [10] and BindML+ [11] algorithms. For more details, please refer to the original papers.

2.1 BindML Algorithm

A structure of the target protein in the PDB format and corresponding MSA of its family including the target sequence are taken as the input for the BindML algorithm (Fig. 1). The main BindML algorithm starts with generating patches on the protein surface. For each surface residue, a patch is defined as neighboring residues within a 15.0 Å radius sphere. The β-carbon

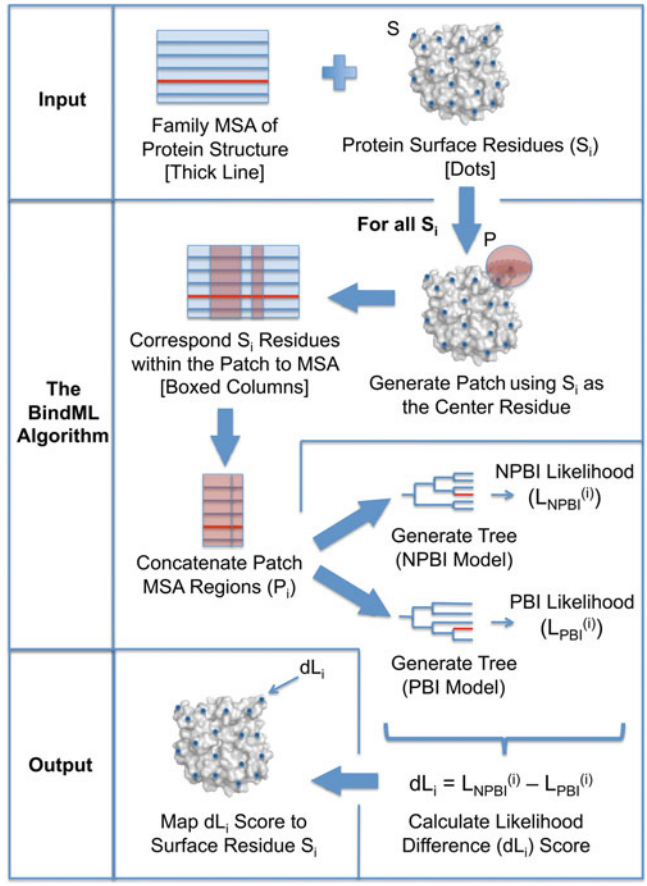


Fig. 1 The steps of the BindML algorithm

of a given amino acid (α-carbon is used for glycine) is selected as the representative point when computing the distance between amino acids. For a patch, all corresponding residues in MSA are concatenated together. The essence of the BindML algorithm is to concatenate surface amino acid residues in a surface patch into a "mini-"MSA (a patch MSA) and judge whether the patch MSA is more likely to occur at protein binding interface or not. A modified version of the PHYML (ver. 2.4.5) program [21] computes the likelihood that a patch MSA comes from protein binding interface (PBI) and non-protein binding interface (NPBI) by constructing phylogenetic trees using amino acid similarity matrices computed for residues at PBI and NPBI, respectively. More concretely, PHYML computes the likelihood of having the input patch MSA following the PBI amino acid similarity matrix (Eq. 1) or NPBI amino acid similarity matrix (Eq. 2) given the initial tree topology. Finally, the difference of the likelihood under PBI and NPBI models provides a score used to predict PPI sites (Eq. 3). For a patch MSA, P_i, which has residue i at the center,

$$L_{\mathrm{NPBI}} = \log\left\{\mathrm{Prob}\left(P_i,\ T_i^{\mathrm{NPBI}} \middle| M_{\mathrm{NPBI}}\right)\right\} \tag{1}$$

$$L_{\mathrm{PBI}} = \log\left\{\mathrm{Prob}\left(P_i,\ T_i^{\mathrm{PBI}} \middle| M_{\mathrm{PBI}}\right)\right\} \tag{2}$$

$$\mathrm{dL} = L_{\mathrm{NPBI}} - L_{\mathrm{PBI}} \tag{3}$$

where M_{NPBI} and M_{PBI} are the amino acid similarity matrices of NPBI and PBI, respectively, and T_i^{NPBI} and T_i^{PBI} are tree generated with M_{NPBI} and M_{PBI}, respectively, for the input patch MSA. The distance likelihood (dL) score is the difference between the log likelihood of the patch MSA being NPBI and PBI. Once all dL scores are calculated, these scores are recast into Z-scores and mapped to corresponding residues. Lower negative scores indicate higher likelihood of PBI mutation patterns, while higher scores show a smaller likelihood.

2.2 BindML+ Algorithm

BindML+ is an extension of BindML, which further predicts whether a predicted PBI site in a query protein performs permanent or transient interaction (Fig. 2). The first step of BindML+ is to predict PBI in the protein surface using BindML as described in the previous section. Then, in the subsequent step, the identified PBI site is classified into either permanent or transient interface. In the first step, once dL scores (Eq. 3) for all surface patches are calculated, these scores are recast into Z-scores and a threshold (0 is used for the website) is placed. A lower (negative) Z-score indicates larger likelihood of PBI mutation patterns. In BindML+, any center residue of a patch with a score that is equal to or smaller than the given threshold value is included in a PBI site for the subsequent step.

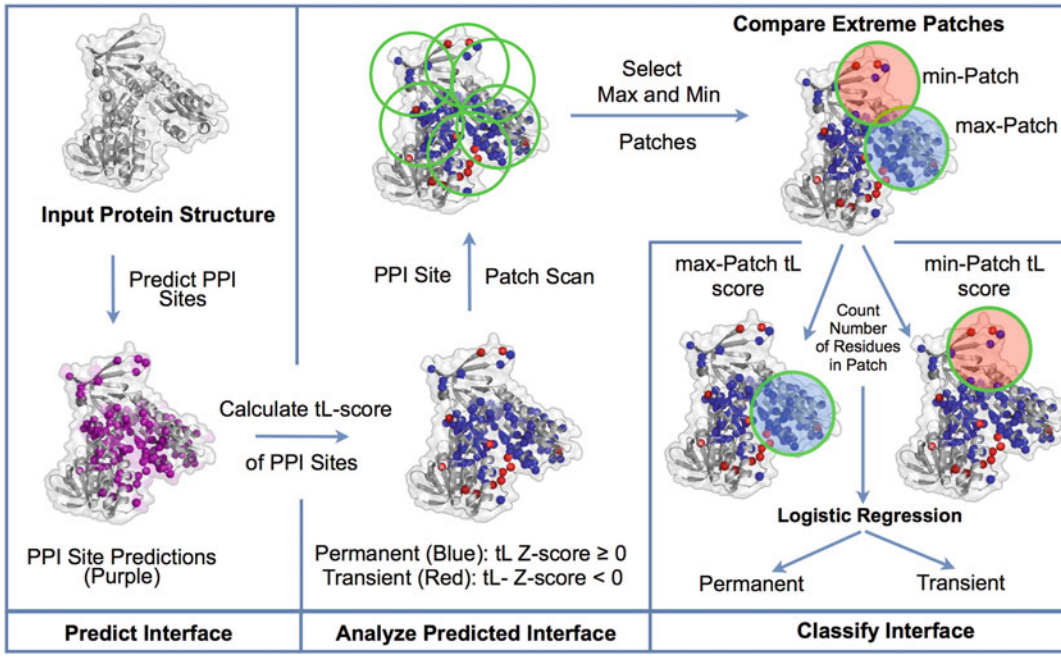


Fig. 2 The steps of BindML+ algorithm

Next, each predicted PBI residue is classified into either permanent or transient, using an amino acid substitution matrix computed from MSAs observed at permanent interaction sites (PERM) and another matrix computed from MSAs at transient interaction sites (TRAN). These two matrices capture characteristic amino acid substitution patterns at permanent and transient interaction sites, respectively. Using these two matrices, similar to Eq. 2, the likelihood that each patch-MSA centered at residue i in a predicted PBI site is from permanent or transient interface ($L_{\mathrm{PERM}}(i)$ and $L_{\mathrm{TRAN}}(i)$) is computed, respectively, and the difference between $L_{\mathrm{PERM}}(i)$ and $L_{\mathrm{TRAN}}(i)$ score, which is named the interface type likelihood (tL) score, is computed:

$$\mathrm{tL}(i) = L_{\mathrm{PERM}}(i) - L_{\mathrm{TRAN}}(i) \tag{4}$$

For a residue with a tL Z-score above zero it is more likely to be permanent, whereas a lower value below zero suggests that it is more likely to be transient.

Then, BindML+ will discriminate the interaction type of the query protein into either the permanent or the transient type using a logistic regression model (LRM). LRM is a binary classifier that tries to fit a set of features using a logit function. Features used in the LRM are based on the tL score and additional related scores of residues at the predicted PBI site. For the details, please refer to the original paper [11].

3 Input and Output of the Servers

3.1 Input Data

Both BindML and BindML+ need four input data to execute. Figure 3 shows the screen capture of the input windows at the top page of BindML.

1. User's email address: An email address is needed for receiving notifications when a submission is processed and completed. An email with the result page URL will be sent to this address.
2. A target PDB file: A query of BindML/BindML+ is a protein tertiary structure in the PDB format. The PDB file can contain chains that are not the target of binding site prediction because in the next step the chain ID of the target will be specified.
3. Specify a chain ID: users need to specify the chain ID in the PDB file. If there are no chain IDs in your PDB, put the underscore "_" instead.
4. Upload a MSA file: This is an optional input to use when users want to use their own MSA. A MSA file to upload must be in the FASTA format and the query PDB sequence is needed to be included in the MSA. Example input files are provided at the bottom of the submission page. If the MSA filed is left empty, the server will execute a search against the Pfam database [22]. Two Pfam databases will be searched. First, Pfam-A will be searched and Pfam-B will only be searched when Pfam-A does not return a match to the query protein sequence. If neither of them matches, an HMMER search [23] will be used to include weak matches from the Pfam database. Finally, the server automatically generates the MSA with the MUSCLE multiple sequence alignment program [24] using the full-length sequences of proteins included in the retrieved Pfam profile.

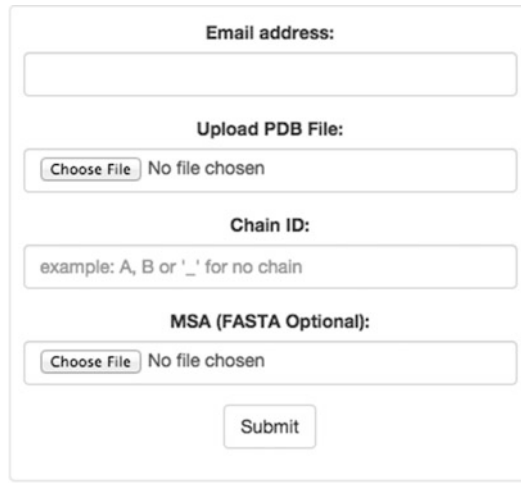


Fig. 3 Input data submission window for BindML

3.2 Output Page with Case Studies

After a submission, an email will be sent to the user when the computation is completed, which includes a link to the result page of the query. Computatation takes typically a few minutes but can take longer depending mainly on the size (length) of the query protein and the number of sequences in the MSA of the query. Below we explain how the results are presented.

3.2.1 BindML Output Page

The interactive BindML result page consists of an integrative structural-level view and a residue-level table with associated prediction scores (Fig. 4). The left panel shows the query protein structure with the JSmol structure viewer (http://wiki.jmol.org/index.php/Jmol_Javascript_Object), where residues are colored based on the *Z*-score of the dL score (Eq. 3). The color ranges from red to blue, where red indicates strong predictions of binding site residues while blue represents residues predicted to be at non-protein binding surface. To visualize prediction, the dL *Z*-score of each residue is written at the B-factor column in the PDB file of the query protein, and JSmol colors residues by reading the dL *Z*-score as B-factor values. The modified PDB file can be downloaded. The last line of the structure panel shows the MSA found for the query protein in the Pfam database. The structure can be rotated and zoomed in/out. All options of different visualization offered by JSmol are available by right click on the structure panel.

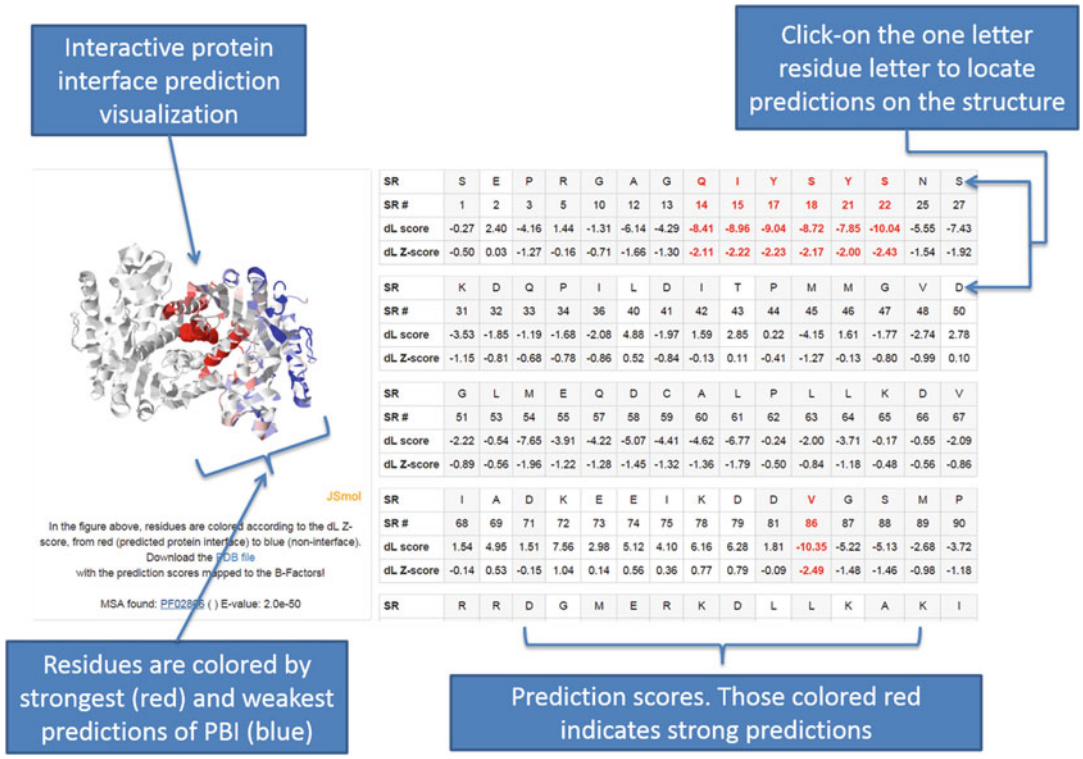


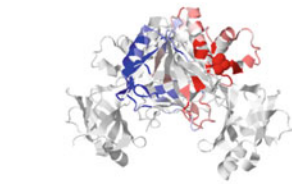
Fig. 4 Example of result page of BindML. PDB entry, 4MDH chain A was used

The right panel shows the detailed list of prediction scores for each residue. Only surface residues are listed. The third row, the dL score, corresponds to Eq. 3, and the final prediction is determined using the dL *Z*-score. Predicted protein binding site residues, i.e., residues that have negative dL *Z*-scores, are highlighted in gray, and those with a high confident score, i.e., dL *Z*-score < -2.0, are colored in red. Amino acid residues in the first row can be clicked and mapped on the left panel with a volume representation.

The example shown in Fig. 4 is the prediction computed for cytoplasmic malate dehydrogenase, A-chain (PDB code: 4MDH). The structure panel visualizes structures of a complex of chains A and B that are contained in the PDB file, but the prediction was computed only for chain A, the colored chain on the right-hand side, without considering the docking conformation with chain B. Apparently, the prediction captures protein binding interface residues of chain A very well with high confidence (red) and surface residues that are not involved in interaction are correctly captured (blue). The area under the curve (AUC) value of this prediction is 0.826. In the left panel, TYR17 is shown in a volume representation, which was invoked by clicking the residue in the table. The MSA used for this prediction is a Pfam entry, PF02866, as indicated at the bottom of the left panel. The match of the query to the Pfam entry is significant with a very low *E*-value of 2.0×10^{-50}. The entry ID is linked to its Pfam page where users can retrieve sequences in the MSA and related information. PF02866 is a MSA for lactate/malate dehydrogenase, alpha/beta C-terminal domain, which agrees with the name of the query protein.

3.2.2 *BindML+ Output Page*

The BindML+ result page essentially shares the same layout with the BindML result page (Fig. 5). The additional information



JSmol

In the figure above, residues are colored according to the tL Z-score, from red (predicted permanent interaction) to blue (predicted transient interaction). Download the PDB file with the predicted scores (dL Z-scores) mapped to the B-Factors and tL z-scores mapped to the occupancy!

MSA found: PF05543 (Peptidase_C47) E-value: 1.1e-85

SR	E	V	S	E	Q	D	P	N	C	A	T	F	P	N	Q
SR #	273	274	275	276	277	278	280	281	282	283	284	285	286	287	288
dL score	-0.07	-0.44	0.25	-0.65	-0.23	-0.49	-0.91	-1.31	-1.00	-2.41	-1.37	-1.86	-0.98	-0.93	-1.63
dL Z-score	-0.33	-0.60	-0.11	-0.75	-0.45	-0.63	-0.93	-1.21	-0.99	-2.00	-1.26	-1.60	-0.98	-0.94	-1.44
tL score	-0.78	0.04	-0.58	-0.13	-0.39	0.11	0.35	0.59	0.61	1.06	0.28	0.68	0.43	-0.51	1.30
tL Z-score	0.11	0.85	0.29	0.70	0.46	0.91	1.13	1.35	1.36	1.77	1.07	1.43	1.20	0.35	1.99

SR	I	E	K	S	Q	G	R	D	H	Y	Q	E	G	V	S
SR #	290	291	294	295	296	297	298	299	301	302	303	304	305	306	308
dL score	0.14	-0.63	-0.34	0.32	0.49	-0.06	0.35	0.34	1.61	0.51	-0.07	-0.74	-0.51	0.93	2.97
dL Z-score	-0.19	-0.73	-0.53	-0.06	0.06	-0.33	-0.04	-0.04	0.86	0.08	-0.33	-0.81	-0.65	0.38	1.82
tL score	-0.81	0.20	-0.19	-0.07	-0.62	0.19	-0.70	-0.65	-1.34	-1.47	-0.78	-0.65	-0.58	-2.75	-2.98
tL Z-score	0.08	0.99	0.64	0.75	0.25	0.98	0.18	0.23	-0.40	-0.51	0.11	0.23	0.29	-1.67	-1.88

SR	Y	N	Q	D	Q	L	K	D	N	V	Q	S	V	S	Q
SR #	309	310	311	313	314	315	317	318	319	320	327	328	329	330	331
dL score	2.92	2.27	1.45	3.49	2.22	1.65	1.90	2.20	2.47	1.97	-0.94	-0.61	0.04	0.11	1.04
dL Z-score	1.79	1.33	0.74	2.19	1.29	0.89	1.06	1.28	1.47	1.11	-0.95	-0.72	-0.26	-0.21	0.45
tL score	-2.98	-2.52	-1.69	-3.64	-2.05	-1.69	-1.54	-1.68	-1.39	-1.86	0.14	1.18	-0.06	-0.98	-1.19
tL Z-score	-1.88	-1.46	-0.71	-2.47	-1.04	-0.71	-0.58	-0.70	-0.44	-0.87	0.94	1.88	0.76	-0.07	-0.26

Fig. 5 Example of BindML+ prediction

predicted by BindML+, prediction of permanent or transient interactions, is shown in the two more rows (tL score and tL *Z*-scores) in the right-hand side table. As described in Subheading 2.2, predicted interface resisdues are classified into either permanent or transent types by their tL *Z*-scores. Residues with tL *Z*-scores greater than or equal to zero correpond to permanent binding site predictions, while tL *Z*-scores below zero represent transient binding site predictions. In the table, predicted protein binding site residues, which have negative dL *Z*-score values, are shown in gray columns. When the binding site predictions are confident (dL *Z*-score < -2.0), letters in the colums are colored in the same way as the BindML output page. Classification of permanent and transient interactions for residues at predicted binding sites (i.e., those highlighted in gray columns) is colored with red or blue, for permanent or transient interactions, respectively.

The top of the BindML+ result page shows the overall prediction of interaction types, either permanent or transient with a confidence score. Note that this overall classification of interaction is computed using information of predicted interaction types of individual residues thus, it is possible that individual residues have different predicted types than the overall interaction type. The overall classification has a score that ranges from 0.0 to 1.0 with 1.0 being the highest confidence. This score is based on the output of the logistic regression used in the interaction type classification.

In a BindML+ page, the structural view on the left of the page visualizes predicted interaction types of individual residues in colors, permanent (red) to transient (blue), according to the tL *Z*-score in the table. The source of the visualized structure in the PDB format, which can be downloaded, has the predicted binding interface scores (dL *Z*-scores) mapped to the B-Factors and the interaction type score, tL *Z*-scores, mapped to the occupancy field.

The protein used as an example in Fig. 5 is staphostatin-staphopain complex (PDB ID: 1pxv). This protein has a permanent interaction. BindML+ correctly predicted its interaction type as permanent with a score of 0.274. The structure panel in Fig. 4 shows binding interface residues of staphostatin (chain on the left) to its inhibitor, staphopain (smaller gray structure on the right side of the complex), are almost all predicted to have permanent interaction (red), while the opposite side of the residues is predicted to have transient interaction properties (blue). Ala283 is emphasized in volume representation. This is a successful example of prediction with the area under the curve (AUC) value of 0.84 for binding residue prediction with 63 predicted binding interface residues out of 73 predicted to have permanent interaction.

4 Conclusion

BindML and BindML+ provide prediction of residues at protein binding interface for a query protein structure entirely from evolutionary information embedded in the MSA of the protein. The algorithms are based on a novel idea of constructing a phylogenetic tree of mini-MSA of local surface regions of the query protein. The performance of these two methods were rigorously benchmarked and compared favorably with related existing methods [10, 11]. The methods can be applied for experimentally solved high-resolution structures, computationally modeled structures, and artificially designed proteins. Also, the methods will be useful in designing protein-protein interactions at desired sites in the query protein and controlling strength of interactions.

Acknowledgments

The authors thank Lyman Monroe for proofreading the manuscript. This work has been supported by grants from the National Institutes of Health (R01GM075004 and R01GM097528), National Science Foundation (IIS1319551, DBI1262189, IOS1127027), and National Research Foundation of Korea Grant funded by the Korean Government (NRF-2011-220-C00004).

References

1. Rajagopala SV, Sikorski P, Kumar A, Mosca R, Vlasblom J, Arnold R, Franca-Koh J, Pakala SB, Phanse S, Ceol A, Hauser R, Siszler G, Wuchty S, Emili A, Babu M, Aloy P, Pieper R, Uetz P (2014) The binary protein-protein interaction landscape of Escherichia coli. Nat Biotechnol 32(3):285–290. doi:10.1038/nbt.2831
2. Hauser R, Ceol A, Rajagopala SV, Mosca R, Siszler G, Wermke N, Sikorski P, Schwarz F, Schick M, Wuchty S, Aloy P, Uetz P (2014) A second-generation protein-protein interaction network of Helicobacter pylori. Mol Cell Proteomics 13(5):1318–1329. doi:10.1074/mcp.O113.033571
3. Mosca R, Ceol A, Aloy P (2013) Interactome3D: adding structural details to protein networks. Nat Methods 10(1):47–53. doi:10.1038/nmeth.2289
4. Venkatraman V, Yang YD, Sael L, Kihara D (2009) Protein-protein docking using region-based 3D Zernike descriptors. BMC Bioinformatics 10:407. doi:10.1186/1471-2105-10-407
5. La D, Kihara D (2008) Predicting binding interfaces of protein-protein interactions. In: Li XL, Ng SK (eds) Biological data mining in protein interaction netrworks. IGI-Global, Hershey, PA, pp 64–79
6. Zhou HX, Qin S (2007) Interaction-site prediction for protein complexes: a critical assessment. Bioinformatics 23(17):2203–2209. doi:10.1093/bioinformatics/btm323
7. Tress M, Ezkurdia I, Grana O, Lopez G, Valencia A (2005) Assessment of predictions submitted for the CASP6 comparative modeling category. Proteins 61(Suppl 7):27–45
8. Li B, Kihara D (2012) Protein docking prediction using predicted protein-protein interface. BMC Bioinformatics 13:7. doi:10.1186/1471-2105-13-7
9. Fleishman SJ, Whitehead TA, Strauch EM, Corn JE, Qin S, Zhou HX, Mitchell JC,

Demerdash ON, Takeda-Shitaka M, Terashi G, Moal IH, Li X, Bates PA, Zacharias M, Park H, Ko JS, Lee H, Seok C, Bourquard T, Bernauer J, Poupon A, Aze J, Soner S, Ovali SK, Ozbek P, Tal NB, Haliloglu T, Hwang H, Vreven T, Pierce BG, Weng Z, Perez-Cano L, Pons C, Fernandez-Recio J, Jiang F, Yang F, Gong X, Cao L, Xu X, Liu B, Wang P, Li C, Wang C, Robert CH, Guharoy M, Liu S, Huang Y, Li L, Guo D, Chen Y, Xiao Y, London N, Itzhaki Z, Schueler-Furman O, Inbar Y, Potapov V, Cohen M, Schreiber G, Tsuchiya Y, Kanamori E, Standley DM, Nakamura H, Kinoshita K, Driggers CM, Hall RG, Morgan JL, Hsu VL, Zhan J, Yang Y, Zhou Y, Kastritis PL, Bonvin AM, Zhang W, Camacho CJ, Kilambi KP, Sircar A, Gray JJ, Ohue M, Uchikoga N, Matsuzaki Y, Ishida T, Akiyama Y, Khashan R, Bush S, Fouches D, Tropsha A, Esquivel-Rodriguez J, Kihara D, Stranges PB, Jacak R, Kuhlman B, Huang SY, Zou X, Wodak SJ, Janin J, Baker D (2011) Community-wide assessment of protein-interface modeling suggests improvements to design methodology. J Mol Biol 414 (2):289–302. doi:10.1016/j.jmb.2011.09.031

10. La D, Kihara D (2012) A novel method for protein-protein interaction site prediction using phylogenetic substitution models. Proteins 80(1):126–141. doi:10.1002/prot.23169
11. La D, Kong M, Hoffman W, Choi YI, Kihara D (2013) Predicting permanent and transient protein-protein interfaces. Proteins 81 (5):805–818. doi:10.1002/prot.24235
12. Fernandez-Recio J, Totrov M, Skorodumov C, Abagyan R (2005) Optimal docking area: a new method for predicting protein-protein interaction sites. Proteins 58(1):134–143. doi:10.1002/prot.20285
13. Xu D, Tsai CJ, Nussinov R (1997) Hydrogen bonds and salt bridges across protein-protein interfaces. Protein Eng 10(9):999–1012
14. Tjong H, Qin S, Zhou HX (2007) PI2PE: protein interface/interior prediction engine. Nucleic Acids Res 35(Web Server issue): W357–W362. doi:10.1093/nar/gkm231
15. Jones S, Thornton JM (1997) Analysis of protein-protein interaction sites using surface patches. J Mol Biol 272(1):121–132. doi:10.1006/jmbi.1997.1234
16. Jones S, Thornton JM (1997) Prediction of protein-protein interaction sites using patch analysis. J Mol Biol 272(1):133–143. doi:10.1006/jmbi.1997.1233
17. Morcos F, Hwa T, Onuchic JN, Weigt M (2014) Direct coupling analysis for protein contact prediction. Methods Mol Biol 1137:55–70. doi:10.1007/978-1-4939-0366-5_5
18. Jones DT, Buchan DW, Cozzetto D, Pontil M (2012) PSICOV: precise structural contact prediction using sparse inverse covariance estimation on large multiple sequence alignments. Bioinformatics 28(2):184–190. doi:10.1093/bioinformatics/btr638
19. Gobel U, Sander C, Schneider R, Valencia A (1994) Correlated mutations and residue contacts in proteins. Proteins 18(4):309–317
20. Kuipers RK, Joosten HJ, Verwiel E, Paans S, Akerboom J, van der Oost J, Leferink NG, van Berkel WJ, Vriend G, Schaap PJ (2009) Correlated mutation analyses on super-family alignments reveal functionally important residues. Proteins 76(3):608–616. doi:10.1002/prot.22374
21. Guindon S, Gascuel O (2003) A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood. Syst Biol 52 (5):696–704
22. Finn RD, Bateman A, Clements J, Coggill P, Eberhardt RY, Eddy SR, Heger A, Hetherington K, Holm L, Mistry J, Sonnhammer EL, Tate J, Punta M (2014) Pfam: the protein families database. Nucleic Acids Res 42(Database issue):D222–D230. doi:10.1093/nar/gkt1223
23. Finn RD, Clements J, Eddy SR (2011) HMMER web server: interactive sequence similarity searching. Nucleic Acids Res 39(Web Server issue):W29–W37. doi:10.1093/nar/gkr367
24. Edgar RC (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Res 32 (5):1792–1797

Chapter 15

OSPREY Predicts Resistance Mutations Using Positive and Negative Computational Protein Design

Adegoke Ojewole*, Anna Lowegard*, Pablo Gainza, Stephanie M. Reeve, Ivelin Georgiev, Amy C. Anderson, and Bruce R. Donald

Abstract

Drug resistance in protein targets is an increasingly common phenomenon that reduces the efficacy of both existing and new antibiotics. However, knowledge of future resistance mutations during pre-clinical phases of drug development would enable the design of novel antibiotics that are robust against not only known resistant mutants, but also against those that have not yet been clinically observed. Computational structure-based protein design (CSPD) is a transformative field that enables the prediction of protein sequences with desired biochemical properties such as binding affinity and specificity to a target. The use of CSPD to predict previously unseen resistance mutations represents one of the frontiers of computational protein design. In a recent study (Reeve et al. Proc Natl Acad Sci U S A 112(3):749–754, 2015), we used our OSPREY (Open Source Protein REdesign for You) suite of CSPD algorithms to prospectively predict resistance mutations that arise in the active site of the dihydrofolate reductase enzyme from methicillin-resistant *Staphylococcus aureus* (SaDHFR) in response to selective pressure from an experimental competitive inhibitor. We demonstrated that our top predicted candidates are indeed viable resistant mutants. Since that study, we have significantly enhanced the capabilities of OSPREY with not only improved modeling of backbone flexibility, but also efficient multi-state design, fast sparse approximations, partitioned continuous rotamers for more accurate energy bounds, and a computationally efficient representation of molecular-mechanics and quantum-mechanical energy functions. Here, using SaDHFR as an example, we present a protocol for resistance prediction using the latest version of OSPREY. Specifically, we show how to use a combination of positive and negative design to predict active site escape mutations that maintain the enzyme's catalytic function but selectively ablate binding of an inhibitor.

Key words OSPREY, Computational protein design, Positive and negative design, Antibiotic resistance prediction

*Corresponding authors.

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_15,

1 Introduction

Antibiotic resistance is an unfortunate consequence of evolutionary pressures on drug targets. In particular, selective pressures from competitive inhibitors that target enzymes elicit single nucleotide polymorphisms that give rise to amino acid changes that preserve catalytic function in the target but disrupt inhibitor binding. Dihydrofolate reductase (DHFR) in *Staphylococcus aureus* is a clinically important example of this mode of resistance. A single amino acid polymorphism in DHFR confers resistance to trimethoprim, a commonly prescribed antibiotic [1]. This and other drug-resistant strains—collectively referred to as methicillin-resistant *Staphylococcus aureus* (MRSA)—cause pneumonia as well as skin, bloodstream, and surgical site infections. Additional mutations in MRSA DHFR (SaDHFR) result in even higher levels of drug resistance.

Successfully predicting resistance-conferring SaDHFR mutations before they emerge can enable the development of more robust inhibitors. However, because 20 amino acids can occur at every residue position, the combinatorially large number of candidate sequences that must be evaluated for resistance far exceeds the capabilities of current experimental methods. Fortunately, computational structure-based protein design (CSPD) is a practical alternative strategy to predict drug resistance over a large set of mutations.

OSPREY (Open Source Protein REdesign for You) [2, 3, 4, 5, 6, 7, 8, 9] is a state-of-the-art, free, and open source suite of computational protein design algorithms. To date, a number of research groups have successfully used OSPREY to perform biomedically important protein designs. For example, we previously used OSPREY to predict escape mutations in SaDHFR that confer resistance to a lead inhibitor [10]. More recently, we used OSPREY to predict escape mutations that grant SaDHFR resistance to a different experimental inhibitor, compound 1; we showed that two novel, predicted mutants (V31L and V31G) were selected in resistance selection experiments along with an additional compensating mutation (F98Y) [11]. Additionally, we used OSPREY to alter the specificity of Gramicidin S Synthetase A [12, 13], to design epitope-specific HIV antibody probes [14], to design peptides to inhibit the interaction between the protein CAL and cystic fibrosis transmembrane conductance regulator (CFTR) [15], and to screen inhibitors of a leukemia-associated protein–protein interaction [16]. Furthermore, the Vaccine Research Center (VRC) used OSPREY to design HIV antibodies that are easier to induce [17]. In [18], we collaborated with the VRC to use OSPREY to design broader and more potent anti-HIV antibodies. Finally, Bailey-Kellogg and colleagues used OSPREY to optimize stability and immunogenicity of therapeutic proteins [19, 20, 21].

OSPREY is based on the following principles:

(a) *Accurate modeling of flexibility in the protein (backbone and side-chains) and ligand captures conformational changes induced by amino acid mutations.* Other CSPD algorithms typically represent amino acid side-chain rotational isomers (rotamers) as discrete points in χ-angle space, resulting in sub-optimal design predictions [4, 6]. OSPREY overcomes the limitations imposed by discrete rotamers by implementing *continuous rotamers*: continuous regions of χ-angle space that more accurately reflect empirically observed side-chain placements [2, 4, 15]. In contrast to protein designs using discrete rotamers, those using continuous rotamers find lower energy conformations and different sequences, leading to more accurate biological predictions [4, 6].

(b) *Ensemble-based design enables more accurate predictions of binding free energy.* Traditional protein design methods focus on locating the global minimum energy conformation (GMEC). However, a protein in solution exists not as a single low-energy structure but as a thermodynamic ensemble of conformations. Since a thermodynamic ensemble of low-energy conformations governs protein-ligand binding [22], models that only consider the GMEC may incorrectly predict binding [15]. OSPREY improves upon GMEC-based protein design by using the K^* algorithm [2, 3], which efficiently approximates the association constant, K_a, of a protein-ligand complex using structural ensembles. In particular, K^* only considers the most probable low-energy conformations and discards the high energy conformations that are rarely populated by either the protein or the ligand.

(c) *Mathematical guarantees of accuracy.* Because CSPD algorithms must search vast sequence and conformation spaces, computational complexity remains a limiting factor in protein design. Accordingly, CSPD programs must rely on a simplified input model, which defines a computationally tractable simplification of the protein design space. Briefly, the input model consists of the initial protein structure(s), the permitted set of mutations to the wild type structure, the allowed protein flexibility, and an energy function to rank the generated conformations. Nevertheless, protein design remains NP-hard [23]. Because of this complexity, heuristic search methods based on stochastic optimization, such as Monte Carlo [24, 25], are often used. However, these methods cannot guarantee to find the lowest energy conformations nor sequences. In contrast, OSPREY uses provable algorithms to determine the lowest energy conformations satisfying the input model. As a consequence, OSPREY

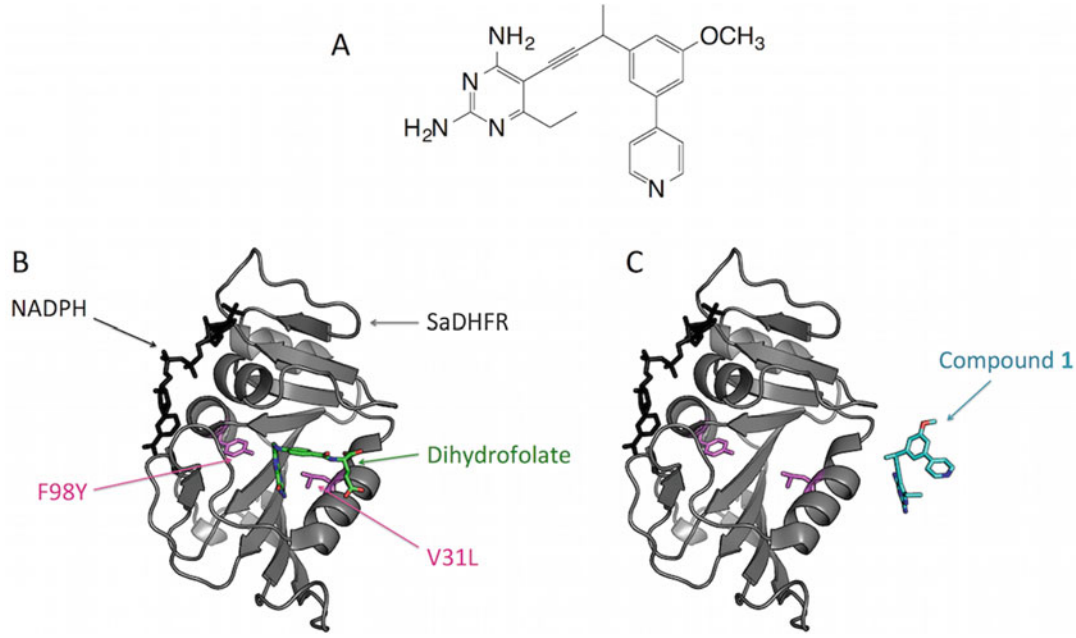


Fig. 1 Positive design to maintain SaDHFR:dihydrofolate binding and negative design to destabilize SaDHFR: compound 1 binding using OSPREY. (**a**) Compound 1, an experimental SaDHFR inhibitor. (**b**) OSPREY positive design objective. OSPREY predicts mutations (*pink*) of SaDHFR (*gray*) that maintain binding of dihydrofolate (*green*) in the SaDHFR active site. These mutations allow SaDHFR to preserve its catalytic activity. The co-factor NADPH is shown in *black*. (**c**) OSPREY negative design objective. OSPREY predicts mutations that destabilize the binding of an inhibitor (compound 1) to SaDHFR. OSPREY predicts SaDHFR candidate escape mutations that bind dihydrofolate but selectively disrupt binding of compound 1

determines protein sequences that satisfy the design objective with mathematical guarantees of accuracy (up to the accuracy of the input model). Crucially, this means that discrepancies between experimental results and predictions by OSPREY are attributable solely to errors in the input model; when using OSPREY any such discrepancies are substantially easier to resolve by making corrections to the input model. On the other hand, the causes of erroneous design predictions are much more difficult to ascertain when using heuristic methods.

Below, we describe the specific application of OSPREY to predict novel, viable resistance mutations that arise in SaDHFR in response to our novel propargyl-linked antifolate inhibitor, compound 1 (Fig. 1a) [26, 11]. The combination of positive and negative design (to maintain native substrate binding and to abrogate inhibitor binding, respectively) in OSPREY is sufficient to predict novel escape mutations in this system (Fig. 1b, c). We use this specific example to illustrate the more general problem of predicting resistance in drug targets in other systems. These extensions may require the

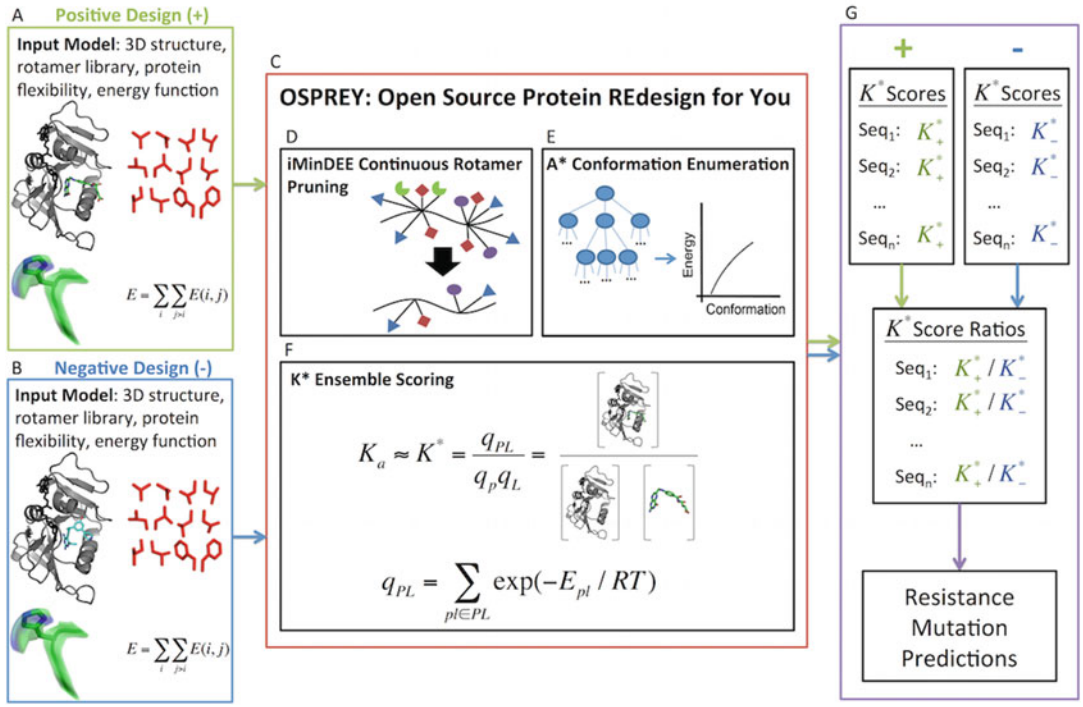


Fig. 2 Processing of positive and negative design input models in OSPREY. (**a**) Input model for positive design. The 3D structure is a model of SaDHFR bound to dihydrofolate and NADPH. (**b**) Input model for negative design. The 3D structure is a model of SaDHFR bound to compound 1 (Fig. 1a) and NADPH. (**c**) Pruning, search, and ensemble scoring algorithms in OSPREY. (**d**) iMinDEE continuous rotamer pruning removes rotamers that provably do not belong to the ensemble of lowest energy conformations. (**e**) A^* conformation enumeration generates an ensemble of conformations in a gap-free, energetically increasing order. (**f**) K^* ensemble scoring approximates Boltzmann-weighted partition functions for the bound and unbound states and subsequently approximates the association constant, K_a, with mathematical guarantees of accuracy relative to the input model. (**g**) Prediction of resistance mutations in OSPREY. (*Top Left*) Positive design K^* scores, $K_+{}^*$, generated by OSPREY for each sequence. (*Top Right*) Negative design K^* scores, $K_-{}^*$, generated by OSPREY for each sequence. (*Middle*) A ratio of the positive design score to the negative design score, K_+^*/K_-^*, for each sequence. (*Bottom*) Sequences are sorted in decreasing order of K^* score ratios. The top predicted mutants, which have the highest ratio of scores, are evaluated experimentally

modeling of backbone flexibility [5], multi-state specificity [7], faster energy functions [9], or efficient sparse approximations [8], all of which are available in OSPREY.

We begin with a detailed description of the input model for OSPREY's positive and negative design steps. The input model consists of 3D structures (determined by nuclear magnetic resonance, X-ray crystallography, or homology modeling), the allowable set of mutations, protein and ligand flexibility parameters, and an energy function (Fig. 2a, b). To predict candidate resistance mutations [i.e., those that bind SaDHFR's natural substrate but not compound 1 (Fig. 1a)], we perform positive (Fig. 2a) and negative (Fig. 2b) designs using structures of SaDHFR:dihydrofolate:NADPH and

SaDHFR:compound 1:NADPH, respectively. Since crystal structures of these complexes were unavailable, we created the respective homology models from [1] and PDB ID 3FQC [27]. Having constructed these models, we considered a sequence space consisting of the most prevalent modes of mutational resistance: single nucleotide polymorphisms to active site residues [28]. These residues are also subject to OSPREY's flexibility model, which specifies the empirically determined set of energetically favorable protein side-chain and ligand rotational isomers in a rotamer library [29]. For improved prediction accuracy, OSPREY's continuous rotamer model extends this rigid definition of a rotamer to a bounded, yet continuously flexible region of side-chain conformation space [4]. Ligands (dihydrofolate and compound 1), which are also modeled using continuous rotamers, are further allowed rigid body rotational and translational degrees of freedom within the active site. Together, the 3D structures, allowable mutations, and protein and ligand flexibility parameters define the conformation space for all candidate resistant mutants. The fourth component of the input model, a computationally efficient all-atom residue-pairwise energy function, is used to evaluate structures in this conformation space. Several energy functions are available in OSPREY [6], but usually, and for this example, the energy function consists of the Amber96 [30] energy function for van der Waals, electrostatic, and dihedral energies and the EEF1 solvation model [31].

Having presented the components of the input model, we now describe the use of OSPREY to predict novel SaDHFR escape mutations. For each mutation defined in the input model, OSPREY performs a positive design step to predict the mutant's binding affinity for SaDHFR's natural substrate (dihydrofolate) and a negative design step to predict its affinity for compound 1. Mutants with both tight binding affinity for dihydrofolate and poor binding affinity for compound 1 are selected as the best candidate mutants. We discuss OSPREY's procedure to predict binding affinity below.

Positive design and negative design are performed and scored separately for each candidate mutant using the iMinDEE [4], A^* [2, 32], and K^* [2, 3, 15] algorithms in OSPREY (Fig. 2c). In a preprocessing step, the iMinDEE algorithm (Fig. 2d) efficiently prunes rotamers that are provably incompatible with the ensemble of lowest energy conformations. Importantly, iMinDEE extends the provable guarantees of the original dead-end elimination algorithms [33, 34] to OSPREY's continuous rotamer model, allowing both biophysically accurate protein modeling and an exponential reduction in the size of the conformation space. Subsequently, the A^* algorithm (Fig. 2e) enumerates the remaining conformations in gap-free energetically increasing order, starting from the global minimum energy conformation (GMEC). The K^* module (Fig. 2f) of OSPREY approximates a Boltzmann-weighted partition

function, q, from this energetically ordered ensemble, S, of conformations:

$$q = \sum_{s \in S} \exp(-E_s/RT)$$

where E_s is the energy of conformation $s \in S$, T is the temperature in Kelvin, and R is the gas constant. To efficiently approximate the full partition function q defined over all conformations in S, K^* halts A^* conformation enumeration when the partial partition function q^*, computed from the ensemble of lowest energy conformations in S, is provably within a factor ε of q. The user specifies ε ahead of time as part of the input model. In practice, K^* achieves a provably accurate ε-approximation to q using only a small fraction of the lowest energy conformations in S. Subsequently K^* approximates the association constant, K_a, for a protein-ligand complex as the ratio of ε-approximated partition functions for the bound and unbound states:

$$\frac{q^*_{PL}}{q^*_P q^*_L}$$

where PL, P, and L represent the protein-ligand complex, the unbound protein, and unbound ligand, respectively. For each candidate mutant, separate positive and negative design K^* scores are computed (Fig. 2g, Top). Since a higher K^* score denotes tighter predicted binding affinity, a resistant mutant would have a high positive design score (for dihydrofolate) and a low negative design score (for compound 1). Therefore, mutants were ranked by their ratio of positive to negative design scores. Mutants with both a higher rank than the wild type and a good positive design score relative to the wild type were considered candidate resistant mutants. Among this set of mutants, a higher ratio of scores indicates a greater degree of predicted resistance to compound 1 (Fig. 2g, Middle). On the other hand, mutants such as L20F, which have high positive to negative design score ratios but low positive design scores, are not considered viable, due to low predicted affinity for dihydrofolate. The top-ranked predicted resistance mutations according to our protocol were recommended for creation and experimental testing.

In summary, we combined positive and negative protein design with the state-of-the-art algorithms in OSPREY to predict viable mutations in SaDHFR that confer resistance to our potent competitive inhibitors [11, 26]. Table 1 shows predictions and experimental characterizations for wild type SaDHFR (Sa(WT)DHFR) and OSPREY's four top-ranked resistance mutations. Each of these mutants (V31L, V31I, L5I, and L5V) had not only higher positive to negative design K^* score ratios than Sa(WT)DHFR, but also a comparable or tighter predicted binding affinity for dihydrofolate than Sa(WT)DHFR. To test our top resistance predictions, we

Table 1
K^* resistance prediction (columns 1–5) and experimental characterization (columns 6–7) of wild type and mutant SaDHFR enzymes from [11]

Enzyme	K^* ratio rank	K^* positive-to-negative design ratio	K^* positive design (dihydrofolate) score	K^* negative design (compound 1) score	k_{cat}/K_M	Fold loss (K_i^{mut}/K_i^{wt}) compound 1
Sa(WT)DHFR	18	1.96 E+06	7.16 E+42	3.66 E+36	6.1±0.3	N/A
Sa(V31L)DHFR	1	7.11 E+21	2.16 E+41	3.04 E+19	1.60±0.06	58
Sa(V31I)DHFR	2	5.95 E+21	4.87 E+36	8.18 E+14	1.74±0.07	36
Sa(L5I)DHFR	3	1.71 E+15	6.06 E+39	3.54 E+24	2.24±0.1	4.4
Sa(L5V)DHFR	4	1.16 E+14	4.01 E+44	3.44 E+30	1.8±0.1	1.9

created these SaDHFR SNP mutants using site-directed mutagenesis. An evaluation of Michaelis–Menten kinetics confirmed that our top four predicted mutant enzymes are catalytically competent, exhibiting small losses in k_{cat}/ K_M. Furthermore, the resistance of our top four mutants, as measured by fold loss in K_i relative to the wild type, correlates perfectly with our predicted K^* ratio rank (*see* [11] for details).

Since these predictions were made in [11], we have substantially improved OSPREY's capabilities with the following algorithmic enhancements: improved backbone flexibility [5], multi-state specificity [7], fast sparse approximations [8], partitioned rotamers for improved energy bounds [35], and a computationally efficient representation of molecular-mechanics and quantum-mechanical energy functions [9]. In the following Materials and Methods sections, with this system as an example, we present a protocol to predict the same SaDHFR escape mutations using the most recent release of OSPREY. The Methods section describes how to install and set up OSPREY (Subheading 3.1), how to perform positive and negative design in OSPREY (Subheading 3.2), how to predict resistant mutants using OSPREY's positive and negative design scores (Subheading 3.3.1), and how to visualize the PDB files that represent OSPREY's structural ensemble predictions (Subheading 3.3.2). Importantly, the paradigm described here is applicable to the prediction of novel escape mutations to any antibacterial, antiviral, or antineoplastic drug. In all these cases, the combination of positive and negative design in OSPREY can be used to model selective pressure by inhibitors on other protein targets.

2 Materials

2.1 Operating System Environment

1. An operating system that supports the Java programming language.
2. Java Runtime Environment (JRE) 7.0 or later.
3. Python version 2.7 (required for post-processing scripts).

2.2 Input Files

The input files can be downloaded at: http://www.cs.duke.edu/donaldlab/Supplementary/mimb2015/OSPREY-V2.2B-MIMB2015.zip and consist of the following:

1. Homology model for positive design: structure of SaDHFR in complex with dihydrofolate, SaDHFR:DHF:NADPH (*see* **Notes 1**, **2**, **3**, and **4**).
2. Homology model for negative design: structure of SaDHFR in complex with compound 1, SaDHFR:compound 1:NADPH (*see* **Notes 1**, **2**, **3**, and **4**).
3. Two expanded amino acid rotamer libraries:

   `LovellRotamer-wt-pos.dat` and `LovellRotamer-wt-neg.dat`
4. Two generic rotamer libraries for non-amino acids:

   `GenericRotamers-fol.dat` and `GenericRotamers-pye.dat` for dihydrofolate and Compound1, respectively.
5. Shell scripts necessary to run software.
6. A Python script to analyze the output.
7. Other default data files also found in the OSPREY software package.

2.3 OSPREY Suite of Algorithms

1. OSPREY 2.2 software package, available at http://www.cs.duke.edu/donaldlab/osprey.php

2.4 Other Software

1. PyMOL 1.6 or later, available at http://www.pymol.org/
2. AmberTools (*see* **Note 3**), available at http://ambermd.org/AmberTools14-get.html

3 Methods

3.1 OSPREY Installation

1. Download the OSPREY version 2.2 suite of protein design algorithms (Subheading 2.3, **item 1**).
2. After downloading the OSPREY software package from the above source, unzip the file to a desired location using the following command:

   ```
   # tar -xvfz OSPREY.tar.gz
   ```

3. Next, add the third-party libraries provided with OSPREY to your classpath:

```
# libpath=/whatever/OSPREY/lib
# export CLASSPATH=$CLASSPATH:$libpath/architec-ture-rules-3.0.0-M1.
jar:$libpath/commons-logging-1.1.1.jar:$libpath/colt-1.2.0.jar:$lib
path/commons-math3-3.0.jar:$libpath/commons-beanutils-1.6.jar:$libpath
/jdepend-2.9.1.jar:$libpath/commons-collections-2.1.jar:$libpath/jop
timizer.jar:$libpath/commons-digester-1.6.jar:$libpath/junit-3.8.1.
jar:$libpath/commons-io-1.4.jar:$libpath/log4j-1.2.14.jar:$libpath/
commons-lang-2.5.jar:$libpath/xml-apis-1.0.b2.jar
```

4. Now, change directories to the `OSPREY` directory and create a new directory, `bin`.
5. Finally, change directories to the `src` directory and run the following command:

```
# javac -d ../bin *.java
```

3.2 Design

In this section, we describe how to run positive and negative design in OSPREY. Nine active site residues were chosen to be continuously flexible within 9° of the rotamers in the Penultimate Rotamer Library [29] and mutable up to one nucleotide substitution: L5{L/V/I/R/Q}, V6{V/A/L/I/F/D/G}, L20{L/V/I/F/S}, L28{L/V/M/W/F/S}, V31{V/A/I/F/L/D/G}, T46{T/A/R/I/K/S}, I50{I/V/L/M/F/N/S/T}, L54{L/R/Q/V}, and F92{F/V/L/I/Y/S/C}. We also apply this flexibility model to rotamers of the ligands (i.e., dihydrofolate and compound 1), whose motions also include rigid body translations and rotations in the active site. To empirically determine a ligand rotamer library for compound 1, we began by modeling roughly 10,000 of its binding conformations to SaDHFR. Next, we used OSPREY's MinDEE/A* algorithm [2] to determine the lowest energy binding conformations beneath a steric threshold. This process yielded 1660 binding poses for compound 1 (see `GenericRotamers-pye.dat` in the OSPREY negative design directory in Subheading 3.2.1). The collection of mutable and flexible residues, including the ligands, resulted in a total of 47 sequences. This set of sequences is used in the following positive and negative designs.

3.2.1 Obtaining Input Files for Design

1. Download the required files for this section, described in Subheading 2.2.
2. Extract the file to create the project directory:

```
# unzip OSPREY-V2.2B-MIMB2015.zip
```

The base directory created is `OSPREY-V2.2B-MIMB2015`. Its sub-directory, `OSPREY-INPUT`, is the parent directory for the positive design directory, `OSPREY-INPUT/pos-design`, and the negative design directory, `OSPREY-INPUT/neg-design`.

3.2.2 Running Positive Design in OSPREY

This section describes how to run the provided scripts (Subheading 2.2, **item 5**) to complete the positive design. The PDB file `pos-design.pdb` (Subheading 2.2, **item 1**) consists of all amino acids within an 8 Å radius of dihydrofolate, `DHF` (*see* **Note 5**).

1. Change to the directory where the files for positive design are located:

   ```
   OSPREY-INPUT/pos-design
   ```

2. Run the provided shell script for positive design.

   ```
   #./runPositiveDesign.sh
   ```

3.2.3 Running Negative Design in OSPREY

This section describes how to run the provided scripts (Subheading 2.2, **item 5**) to complete the negative design. The PDB file `neg-design.pdb` (Subheading 2.2, **item 2**) consists of all amino acids within an 8 Å radius of compound 1, `PYE` (*see* **Note 5**).

1. Change to the directory where the files for positive design are located:

   ```
   OSPREY-INPUT/negative-design
   ```

2. Run the provided shell script for negative design.

   ```
   #./runNegativeDesign.sh
   ```

3.3 OSPREY Output

3.3.1 Predicting Resistance from the Ratio of OSPREY Positive to Negative Design Scores

This section describes how to rank sequences by their predicted resistance to compound 1. A python script is provided to complete this process (Subheading 2.2, **item 5**).

1. Move to the `OSPREY-INPUT` directory.
2. Run the provided Python script:

   ```
   # python summarizeResults.py
   ```

Each row of output is formatted as follows: mutation, positive design score (log scale), negative design score (log scale), and ratio of design scores (log scale). The mutations are ordered by increasing order of score ratios. So, the mutation in the last line of the output has the highest positive to negative design ratio. (*See* **Note 6** for the interpretation of a positive or negative design K^* score of 0). From this list, the top candidate resistant mutants are those with both a high positive design score (i.e., high predicted binding affinity for dihydrofolate relative to the wild type) and a high positive to negative design score ratio (*see* **Note 7**).

3.3.2 Structural Analysis of OSPREY Output

The script in Subheading 3.3.1 ranks sequences by increasing order of positive to negative design score ratios. Candidate resistant mutants, which have high positive design scores and high score ratios, can be identified visually in this list. Below, we describe how to view the lowest energy structures from each sequence.

After completing positive and negative design (Subheadings 3.2, **step 2** and 3.2, **step 3**) OSPREY outputs the PDB files for the top ten conformations for each sequence. This section describes these PDB files and how to view them. Each PDB file name takes on one of the following formats:

*n*_aaaaaaaaa_0_*m*.pdb
*n*_X_1_*m*.pdb
*n*_aaaaaaaaaX_2_*m*.pdb

where *n* is an index assigned to each sequence and *m* is a three digit number ranking one sequence's set of ten conformations from lowest to highest energy. Each string of a's corresponds to an amino acid sequence (e.g., LVLLVTILF). X represents the non-amino acid ligand (i.e., dihydrofolate or compound 1). The first format corresponds to SaDHFR unbound to the ligand (either dihydrofolate for the positive design or compound 1 for the negative design). The second format corresponds to the ligand unbound to SaDHFR. Finally, the third format corresponds to SaDHFR in complex with the ligand.

1. Change directories into OSPREY/pos-design/ksConfs

   This directory contains all of the PDB files output for the positive design (Subheading 3.2, **step 2**).
2. Open and view the PDB files using PyMOL (Subheading 2.4, **item 1**). Several files can be opened and viewed simultaneously.
3. Change directories into OSPREY/neg-design/ksConfs

   This directory contains all of the PDB files output for the positive design (Subheading 3.2, **step 3**).
4. Open and view the PDB files using PyMOL (Subheading 2.4, **item 1**). Several files can be opened and viewed simultaneously.

4 Notes

1. In this example, we modeled the inputs for both the positive and negative design steps from structures of related ligands bound to SaDHFR. Other 3D protein structures (i.e., determined by NMR and X-ray crystallography) are also viable input structures for OSPREY.
2. A structure of dihydrofolate (DHF) or compound 1 bound to SaDHFR was not available when the original predictions were made. As a result, the bound complex of SaDHFR:DHF:NADPH (positive design) was modeled on the coordinates of a single mutant Sa(F98Y)DHFR bound to folate and NADPH [1]. (The structure upon which the model is based was not deposited in the Protein Data Bank.) The structure for SaDHFR:compound 1:NADPH (negative design) was

modeled using the bound structure of a related SaDHFR inhibitor (PDB ID 3FQC, [27]).

3. It is often necessary to alleviate steric clashes in the input structures prior to running OSPREY. This is achieved by performing an energy minimization step using AmberTools (*See* Subheading 2.4, **item 2**). This process is detailed in the Antechamber tutorial:

   http://ambermd.org/tutorials/basic/tutorial4b/

4. To parameterize a non-protein compound in the input structure into an OSPREY-compatible format, replace the *antechamber* command in the Antechamber Tutorial with the following command:

   ```
   # antechamber -i x.pdb -fi pdb -o x.prepi
   -fo prepi -c bcc -s 2
   ```

   and append the contents of output file *x.prepi* (where *x* is the base name of the.pdb file containing only the coordinates of the non-protein compound), starting from `This is a remark line`, to the file `all_nuc94_and_gr.in`, which is part of OSPREY's input model. Next, create a file named

   ```
   GenericRotamers.dat
   ```

   to store rotamers for the compound. To determine rotamers for the compound, open the structure in Pymol and use the `Wizard > Measurement` tool in PyMOL (*see* Subheading 2.4, **item 1**). Add rotamers in the format specified in the OSPREY manual (*see* **Note 8**). Reference this file in `System.cfg` using the `grotFile`*i* keyword.

5. To create an 8 Å shell of a protein for your own designs, use PyMOL (*see* Subheading 2.4, **item 1**).

6. Resistance (i.e., positive to negative design ratio) rankings in which either the positive or negative design K^* score is 0 are handled specially. Mutations for which only the negative design score is 0 receive a score ratio of infinity. Mutations for which either only the positive design score is 0 or both positive and negative design scores are 0 receive a design ratio of 0.

7. A candidate resistant mutant has both a high positive design score (indicating of high predicted binding affinity for dihydrofolate) and a low negative design score (denoting low predicted binding affinity for compound 1). Mutants with a high positive to negative design score ratio but a low positive design score (such as L20F) relative to the wild type are not considered viable, as they are predicted to bind dihydrofolate poorly.

8. To perform your own protein designs using OSPREY, please refer to the user manual found in the OSPREY software download from Subheadings 2.3 and 3.1.

9. The results presented in [11] were performed using OSPREY 1.1a. This can lead to slightly different results than those in the newer version of OSPREY. To reproduce the results in [11] exactly, please download the code from:

   http://www.cs.duke.edu/donaldlab/Supplementary/mimb2015/OSPREY-V2010-MIMB2015.zip

10. Nevertheless, for future predictions, we recommend using the latest version of OSPREY for improved accuracy and speed.

Acknowledgements

The authors would like to thank Drs. Mark Hallen and Kyle Roberts for thoughtful suggestions and technical assistance. This work was supported by NIH grant R01 GM-78031 to B.R.D., R01 AI-111957 to A.C.A., and A.O. was supported in part by NSF Graduate Research Fellowships Program Award 1106401.

References

1. Dale GE, Broger C, D'Arcy A, Hartman PG, DeHoogt R, Jolidon S, Kompis I, Labhardt AM, Langen H, Locher H, Page MG, Stüber D, Then RL, Wipf B, Oefner C (1997) A single amino acid substitution in Staphylococcus aureus dihydrofolate reductase determines trimethoprim resistance. J Mol Biol 266 (1):23–30
2. Georgiev I, Lilien RH, Donald BR (2008) The minimized dead-end elimination criterion and its application to protein redesign in a hybrid scoring and search algorithm for computing partition functions over molecular ensembles. J Comput Chem 29(10):1527–1542
3. Donald BR (2011) Algorithms in structural molecular biology. MIT Press, Cambridge
4. Gainza P, Roberts KE, Donald BR (2012) Protein design using continuous rotamers. PLoS Comput Biol 8(1):e1002335
5. Hallen MA, Keedy DA, Donald BR (2013) Dead-end elimination with perturbations (DEEPer): a provable protein design algorithm with continuous sidechain and backbone flexibility. Proteins 81(1):18–39
6. Gainza P, Roberts KE, Georgiev I, Lilien RH, Keedy DA, Chen C-Y, Reza F, Anderson AC, Richardson DC, Richardson JS, Donald BR (2013) OSPREY: protein design with ensembles, flexibility, and provable algorithms. Methods Enzymol 523:87–107
7. Hallen MA, Donald BR (2015) COMETS (constrained optimization of multistate energies by tree search): a provable and efficient algorithm to optimize binding affinity and specificity with respect to sequence. Research in computational molecular biology (RECOMB), vol 9029. Springer, Cham, pp 122–135
8. Jou J, Jain S, Georgiev I, Donald BR (2015) BWM*: a novel, provable, ensemble-based dynamic programming algorithm for sparse approximations of computational protein design. Research in computational molecular biology (RECOMB), vol 9029. Springer, Cham, pp 154–166
9. Hallen MA, Gainza P, Donald BR (2015) Compact representation of continuous energy surfaces for more efficient protein design. J Chem Theory Comput 11(5):2292–2306
10. Frey KM, Georgiev I, Donald BR, Anderson AC (2010) Predicting resistance mutations using protein design algorithms. Proc Natl Acad Sci U S A 107(31):13707–13712
11. Reeve SM, Gainza P, Frey KM, Georgiev I, Donald BR, Anderson AC (2015) Protein design algorithms predict viable resistance to an experimental antifolate. Proc Natl Acad Sci U S A 112(3):749–754
12. Stevens BW, Lilien RH, Georgiev I, Donald BR, Anderson AC (2006) Redesigning the PheA domain of gramicidin synthetase leads to a new understanding of the enzyme's mechanism and selectivity. Biochemistry 45 (51):15495–15504

13. Chen C-Y, Georgiev I, Anderson AC, Donald BR (2009) Computational structure-based redesign of enzyme activity. Proc Natl Acad Sci U S A 106(10):3764–3769
14. Georgiev I, Schmidt S, Li Y, Wycuff D, Ofek G, Doria-Rose N, Luongo T, Yang Y, Zhou T, Donald BR, Mascola J, Kwong P (2012) Design of epitope-specific probes for sera analysis and antibody isolation. *Retrovirology* 9:50
15. Roberts KE, Cushing PR, Boisguerin P, Madden DR, Donald BR (2012) Computational design of a PDZ domain peptide inhibitor that rescues CFTR activity. PLoS Comput Biol 8(4):e1002477
16. Gorczynski MJ, Grembecka J, Zhou Y, Kong Y, Roudaia L, Douvas MG, Newman M, Bielnicka I, Baber G, Corpora T, Shi J, Sridharan M, Lilien R, Donald BR, Speck NA, Brown ML, Bushweller JH (2007) Allosteric inhibition of the protein-protein interaction between the leukemia-associated proteins Runx1 and CBFbeta. Chem Biol 14(10):1186–1197
17. Georgiev IS, Rudicell RS, Saunders KO, Shi W, Kirys T, McKee K, O'Dell S, Chuang G-Y, Yang Z-Y, Ofek G, Connors M, Mascola JR, Nabel GJ, Kwong PD (2014) Antibodies VRC01 and 10E8 neutralize HIV-1 with high breadth and potency even with IG-framework regions substantially reverted to germline. J Immunol 192(3):1100–1106
18. Rudicell RS, Kwon YD, Ko S-Y, Pegu A, Louder MK, Georgiev IS, Wu X, Zhu J, Boyington JC, Chen X, Shi W, Yang Z-Y, Doria-Rose NA, McKee K, O'Dell S, Schmidt SD, Chuang G-Y, Druz A, Soto C, Yang Y, Zhang B, Zhou T, Todd J-P, Lloyd KE, Eudailey J, Roberts KE, Donald BR, Bailer RT, Ledgerwood J, NISC Comparative Sequencing Program, Mullikin JC, Shapiro L, Koup RA, Graham BS, Nason MC, Connors M, Haynes BF, Rao SS, Roederer M, Kwong PD, Mascola JR, Nabel GJ (2014) Enhanced potency of a broadly neutralizing HIV-1 antibody in vitro improves protection against lentiviral infection in vivo. J Virol 88(21):12669–12682
19. Parker AS, Choi Y, Griswold KE, Bailey-Kellogg C (2013) Structure-guided deimmunization of therapeutic proteins. J Comput Biol 20(2):152–165
20. Salvat RS, Choi Y, Bishop A, Bailey-Kellogg C, Griswold KE (2015) Protein deimmunization via structure-based design enables efficient epitope deletion at high mutational loads. Biotechnol Bioeng 112(7):1306–1318
21. Zhao H, Verma D, Li W, Choi Y, Ndong C, Fiering SN, Bailey-Kellogg C, Griswold KE (2015) Depletion of T cell epitopes in lysostaphin mitigates anti-drug antibody response and enhances antibacterial efficacy in vivo. Chem Biol 22(5):629–639
22. Gilson MK, Given JA, Bush BL, McCammon JA (1997) The statistical-thermodynamic basis for computation of binding affinities: a critical review. Biophys J 72(3):1047–1069
23. Pierce NA, Winfree E (2002) Protein design is np-hard. Protein Eng 15(10):779–782
24. Jiang X, Farid H, Pistor E, Farid RS (2000) A new approach to the design of uniquely folded thermally stable proteins. Protein Sci 9 (2):403–416
25. Kuhlman B, Baker D (2000) Native protein sequences are close to optimal for their structures. Proc Natl Acad Sci U S A 97 (19):10383–10388
26. Frey KM, Lombardo MN, Wright DL, Anderson AC (2010) Towards the understanding of resistance mechanisms in clinically isolated trimethoprim-resistant, methicillin-resistant Staphylococcus aureus dihydrofolate reductase aureus dihydrofolate reductase. J Struct Biol 170(1):93–97
27. Frey KM, Liu J, Lombardo MN, Bolstad DB, Wright DL, Anderson AC (2009) Crystal structures of wild-type and mutant methicillin-resistant Staphylococcus aureus dihydrofolate reductase reveal an alternate conformation of NADPH that may be linked to trimethoprim resistance. J Mol Biol 387 (5):1298–1308
28. Frey KM, Viswanathan K, Wright DL, Anderson AC (2012) Prospective screening of novel antibacterial inhibitors of dihydrofolate reductase for mutational resistance. Antimicrob Agents Chemother 56(7):3556–3562
29. Lovell SC, Word JM, Richardson JS, Richardson DC (2000) The penultimate rotamer library. Proteins 40(3):389–408
30. Pearlman DA, Case DA, Caldwell JW, Ross WS, Cheatham TE, DeBolt S, Ferguson D, Seibel G, Kollman P (1995) Amber: a package of computer programs for applying molecular mechanics, normal mode analysis, molecular dynamics and free energy calculations to simulate the structural and energetic properties of molecules. Comput Phys Commun 91(42):1–41
31. Lazaridis T, Karplus M (1999) Discrimination of the native from misfolded protein models with an energy function including implicit solvation. J Mol Biol 288(3):477–487
32. Hart PE, Nilsson NJ, Raphael B (1968) A formal basis for the heuristic determination of

minimum cost paths. IEEE Trans Syst Sci Cybern 4:100–114

33. Desmet J, De Maeyer M, Hazes B, Lasters I (1992) The dead-end elimination theorem and its use in protein side-chain positioning. Nature 356(6369):539–542
34. Goldstein RF (1994) Efficient rotamer elimination applied to protein side-chains and related spin glasses. Biophys J 66(5): 1335–1340
35. Roberts KE, Donald BR (2015) Improved energy bound accuracy enhances the efficiency of continuous protein design. Proteins 83(6): 1151–1164

Part III

Computational Protein Design of Specific Targets

Chapter 16

Evolution-Inspired Computational Design of Symmetric Proteins

Arnout R.D. Voet, David Simoncini, Jeremy R.H. Tame, and Kam Y.J. Zhang

Abstract

Monomeric proteins with a number of identical repeats creating symmetrical structures are potentially very valuable building blocks with a variety of bionanotechnological applications. As such proteins do not occur naturally, the emerging field of computational protein design serves as an excellent tool to create them from nonsymmetrical templates. Existing pseudo-symmetrical proteins are believed to have evolved from oligomeric precursors by duplication and fusion of identical repeats. Here we describe a computational workflow to reverse-engineer this evolutionary process in order to create stable proteins consisting of identical sequence repeats.

Key words Symmetrical proteins, Repeat proteins, Rosetta, Evolution, Ancestral reconstruction, Computational protein design

1 Introduction

During the last few years, the field of bionanotechnology has emerged as an important contributor to nanotechnological research by constructing nanodevices from biological components. The target areas of this emerging research discipline range from the development of biopharmaceutical nanodevices for dedicated drug delivery to the construction of microelectronics through the combination of metallo-chemistry with bio-macromolecular design.

Symmetrical protein assemblies are an interesting class of proteins that lend themselves to this type of application. However there are no examples of monomeric perfectly symmetrical naturally occurring proteins to be used as structural frameworks for such design [1].

Structural analysis of many thousands of proteins has indicated that the majority share common molecular architectures at different levels [2]. Moreover, many proteins appear to consist of tandem repeats of domains or subdomains [3]. Proteins consisting of such

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_16,

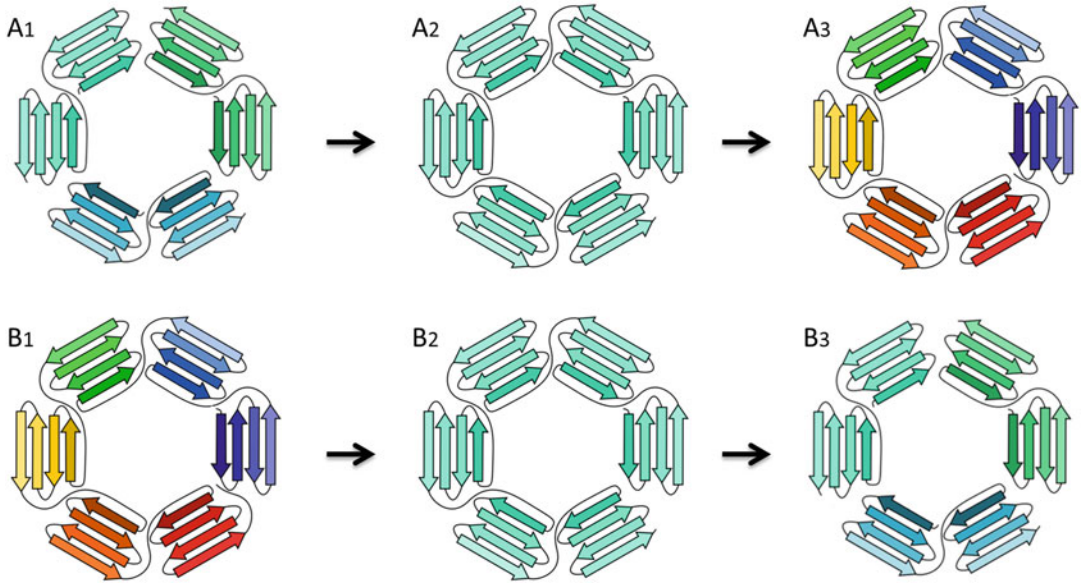


Fig. 1 Reverse engineering evolution to create symmetrical proteins. The protein evolutionary theory of duplication and fusion (shown on the *top row*) suggests that genes for pseudo-symmetrical proteins originate from a smaller ancestral coding fragment. This fragment encodes a polypeptide that can assemble as a multimer (A_1). After duplication and fusion of the ancestral gene, a monomeric symmetrical protein is produced with the same topology (A_2). Subsequent genetic drift and natural selection diversify the amino acid sequence (A_3). Computationally reverse-engineering this evolutionary process is shown on the *bottom row*. Starting from a naturally occurring protein (B_1) with multiple domains of similar fold but different sequence, a monomeric perfectly symmetrical protein (B_2) is designed that may, if desired, be cleaved into a self-assembling fragment (B_3)

repeated motifs exhibit repeated secondary and tertiary structure, creating symmetrical configurations.

This general principle of motif reuse arises from the fact that evolution is driven in part by the duplication of genetic material, and this duplication is sometimes followed by fusion to create tandemly repeated coding sequences [4, 5]. Such genes (and the encoded proteins) will initially show perfectly repeating sequences, but, under the influence of genetic drift and evolutionary pressure, they evolve more diversified repeats as the new protein optimizes its functionality (*see* Fig. 1). On a three-dimensional level, however, the overall tertiary structure retains a highly conserved secondary structure with repeated motifs, reflecting the ancestral genetic element.

Classical examples of symmetrical proteins include the Ankyrin, Armadillo, HEAT, TPR, and LRR repeats that create overall a linear or toroidal shape depending on the number of tandem repeats. However, repeats can also be observed in globular proteins, such as trefoils and TIM-barrels, which exhibit pseudo-rotational symmetry with a fixed number of repeats (three- and eightfold respectively) [6]. Given the symmetrical nature of repeat proteins, they

are ideal starting points for engineering protein assemblies that can be designed with a module-based approach [7, 8]. Thus, not surprisingly, symmetrical proteins have been targeted by several groups for protein design. For example DARPINs are designer proteins based on the Ankyrin repeat with a variety of functions [9]. Recently, Baker and coworkers made great process in the computational design of repeat proteins with toroidal symmetry [10, 11].

Other successes in symmetric protein design include work by Lanci et al. to build protein crystal lattices from symmetrical assemblies of coiled-coil motifs, and the creation of an alpha-barrel protein by Woolfson and coworkers [12, 13]. Furthermore, two groups independently produced symmetrical trefoil proteins named Symfoil and ThreeFoil [14, 15].

However applications can be envisaged for repeat proteins with different rotational symmetry, and the beta-propeller family is of interest since it provides templates with 4–10 repeats arranged in a pseudo-symmetrical fashion about a central point. Beta-propeller proteins are named because of their roughly circular, propeller-like architecture, with each domain or "blade" forming a fan-like structure [16]. These proteins are widely distributed and play numerous roles in cells from each Kingdom of Life; propeller proteins include enzymes and membrane channels, but a very large number have ligand binding functions, either carrying simple ligands such as metal ions or as scaffolds in protein networks, holding different partner proteins together. Nature has clearly selected propeller proteins to carry out these different roles due to the versatility of the fold and its modular structure. This endorsement by natural selection makes propellers interesting building blocks for the artificial engineering of proteins with chosen properties. Recently a symmetrical self-assembling protein called Pizza6 was computationally designed using a natural nonsymmetric six-bladed propeller protein as the template [17]. Pizza6 contains six identical copies of a 42-residue "blade". It was designed by reversing the evolutionary mechanism of duplication and fusion, turning back time by calculating the most likely ancestral sequences. Working on the principle that a stable, symmetrical ancestor must have existed in the past, such sequences provide an excellent starting point for finding perfectly symmetrical variants of existing pseudo-symmetrical proteins. While the propeller protein Pizza6 was designed as a test of this principle, the same method can be used to derive repeat sequences from any template protein built from nonidentical but related domains or subdomains of the same fold. Recently, we have reported the biomineralization of a $CdCl_2$ nanocrystal by a Pizza variant with an engineered metal binding site [18]. We anticipate that in the future new symmetrical proteins may be designed, using the method proposed here, to biomineralize different nanocrystals for specific applications.

2 Materials

Due to the computational nature of this approach, up-to-date hardware is required which can run current, supported versions of the correct software.

2.1 Hardware Dependencies

Recommended hardware specifications are a workstation with at least 8GB of RAM and one of the following operating systems: GNU/Linux (any 64-bit distribution should be suitable, Red Hat and Ubuntu are popular and recommended), Mac OS X v10.6 or later, Windows 7 or later.

2.2 Software Dependencies

As protein structures will be analyzed and designed a protein visualization tool, preferably with stereographic rendering options, will need to be installed. Examples of such software are Chimera or Pymol [19, 20].

1. A structural alignment program to align the amino acid sequences by superpositioning protein backbones in 3D space is preferable to simple amino acid sequence alignments. A good example is STRAP [21].
2. The "evolutionary" relationship and distance between the repeating motifs can be represented by a phylogenetic tree. These can be created from the sequence alignments using locally installed software or webservers. An example is PHYLIP [22].
3. The creation of the "putative ancestral" consensus sequences requires specialized bioinformatics algorithms. Several choices are available, either installed locally or available from a webserver such as the FastML server [23].
4. The bio-macromolecular modeling suite Rosetta is required for the modeling of the protein structures and the scoring of the designed sequences [24]. Rosetta bundles a variety of modeling methods including structure prediction, protein and small molecule docking, backbone modeling and protein and enzyme design. Additionally, PyRosetta, the python interface to Rosetta is required by our sequence mapping tool [25]. It requires Python version 2.6 or later to be installed with shared libraries enabled.
5. Finally the "Sequence Mapping Tool" itself is required for computational scoring of the designed sequences against the designed backbone.

   This tool is written in Python and aims at mapping an amino acid sequence (the putative ancestral sequences) onto a protein backbone (the desired symmetrical backbone conformation). It takes an input protein structure in PDB format and a sequence

in FASTA format (or a list of sequences in multi-FASTA format). It builds a model PDB file with each sequence, and outputs the models together with the corresponding Rosetta scores. Optionally the protein may be relaxed after the mapping, and the desired number of relaxed structures may be set. The relaxation is handled by the default Rosetta FastRelax protocol, with the latest "Talaris2014" scoring function. The full python script can be found in Subheading 4 (*see* **Note 1**).

3 Methods

The design protocol can be divided into the following consecutive computational stages (*see* Fig. 2). The output sequences should be tested experimentally by standard protein expression and purification approaches, which will not be covered here.

3.1 Selection of the Nonsymmetrical Parent Template

1. First the 3D structures of tandem repeat proteins exhibiting the pseudo symmetry of choice should be retrieved from the Protein Databank (PDB) (*see* Fig. 2a and **Note 2**). The most straightforward method to find templates of the desired fold and symmetry is to use databases, such as the Structure Classification of Protein (SCOP), which assign known structures to different classes [26].
2. The sequences of potential templates should be carefully analyzed to identify a protein with optimal sequence and structural features. Ideally the different repeats are closely related, with obvious sequence similarity and without significant insertions or deletions per sequence. The most straightforward method is by visualization of the protein structures and utilizing structure-based alignment algorithms to check the level of structural and sequential conservation (*see* Fig. 2b and **Note 3**).

3.2 Generation of "Ancestral" Consensus Sequences

1. Once the pseudo-symmetrical parent template has been selected, the individual repeats should be isolated into separate structural files (PDB format) and corresponding sequence files. This can be easily performed manually using a simple molecular visualizer (*see* **Note 4**).
2. In order to create a consensus sequence, the sequences of the individual repeats must be aligned (*see* **Note 5**). As these sequences will be later mapped in 3D on the backbone, the alignment should preferably be performed using a 3D structural alignment algorithm. The alignment can be manually edited to remove insertions or deletions (*see* Fig. 2c, d). The alignment should be saved in the appropriate format (ALN) to

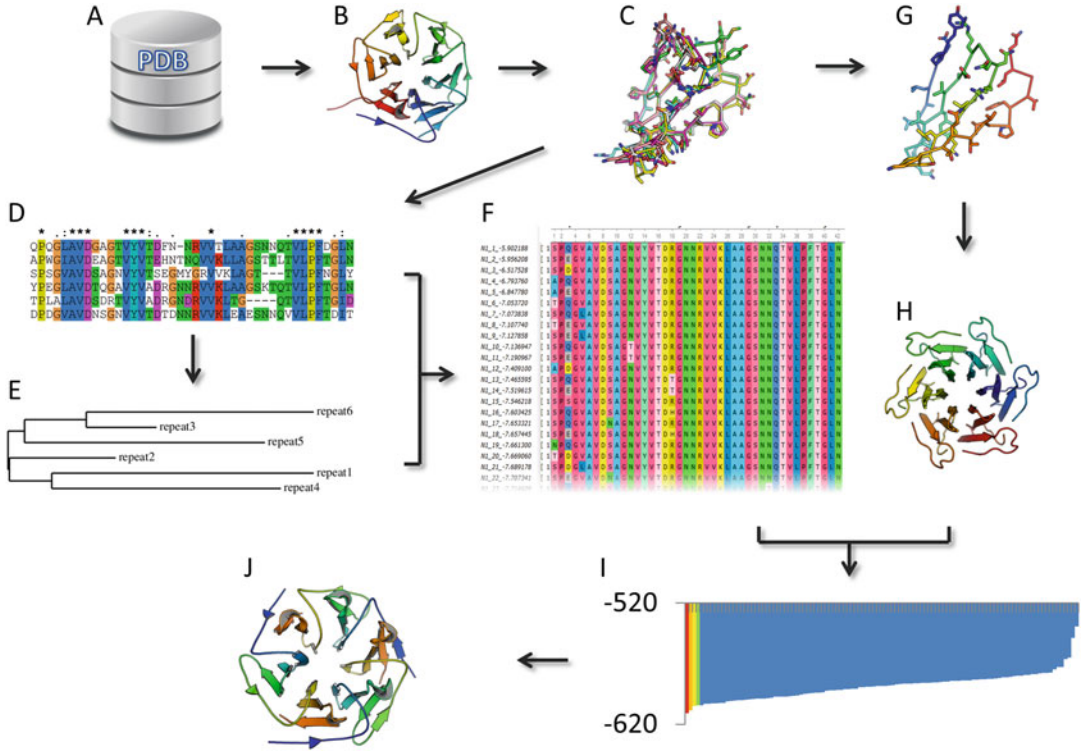


Fig. 2 Flowchart of the evolution-inspired computational procedure to design symmetrical proteins. Starting from the protein structure database (**a**), a suitable parent repeat protein with the correct symmetry is identified (**b**). The individual repeats are isolated and aligned using a structure-based alignment protocol (**c**). This serves as the input for the amino acid sequence alignment (**d**), which will also produce a phylogenetic tree of the different repeats (**e**). The alignment and the phylogenetic tree are used together to create putative ancestral consensus sequences (**f**). From the structurally aligned repeat structures a backbone motif is selected (**g**) to create the desired symmetrical protein backbone (**h**). This backbone serves as the template to score and rank the putative ancestral sequences (**f**) using a 3D computational protein design tool based on Rosetta (**i**). The *top* sequences can then be experimentally validated (**j**)

be used as input for the phylogenetic tree and ancestral sequence reconstruction steps.

3. A phylogenetic tree must be created from the aligned sequences (*see* Fig. 2e). Each branch point or "node" of the tree represents a divergence in sequence between the different domains, and an "ancestral" consensus sequence must be determined for each node. While for studies of evolution it is important to derive the most accurate tree possible, for the purpose of protein design the correct rooting of the tree is less important since a number of possible ancestral sequences are used (*see* **Note 6**).
4. Finally the alignment of the sequence repeats and the derived phylogenetic tree are used to predict the so-called ancestral sequences at the different nodes of the tree (*see* Fig. 2f).

While the algorithm will produce a top-scoring result, it is better to collect many possible ancestral sequences as proceeding from a single result is unlikely to yield the desired symmetrical and stably folding sequence. The derived consensus sequences are stored in a multi FASTA format file.

3.3 Generation of 3D Templates

1. The ideal fully symmetrical template backbone has to be chosen from the models of individual repeat motifs (that were aligned in Subheading 3.2, **step 2**). First, any models with insertions or deletions compared to the reconstructed ancestral sequences should be removed. Secondly, models with any distinctly unique backbone conformations compared to the other repeats should be discarded (*see* **Note 7**). The final repeat structure is identified as the one with the highest homology to the sequence of the ancestral sequences (*see* Fig. 2g).
2. To avoid steric clashes of the polypeptide termini during assembly of the complete symmetrical backbone template by docking, the first or last amino acid residue of the selected repeat model should be removed. Preferably this residue should not contribute to the inter-domain packing, and lie in a loop region. Due to its small size and flexibility, glycine is a good choice for re-connecting the separate copies of the isolated repeat if the deleted side-chain is not conserved and makes no inter-domain interactions (*see* **Note 8**).
3. To create a fully symmetrical backbone model from the single repeat structures, Rosetta's SymmetryDocking function can be used with the default parameters, and the symmetry operator of the protein being designed (*see* Fig. 2g). The 10,000 individual structures should be ranked and the top solutions should be visually investigated. The top solution should have a perfectly symmetrical backbone with the desired topology (*see* **Note 9**).
4. The output structures of the docking algorithm however consist of multiple chains. Clipping the fitted repeat model by one residue may result in a salt bridge interaction between the N- and C-termini of adjacent chains. This gap can be easily repaired by re-introducing one amino acid residue and linking the chains together, resulting in a perfectly symmetrical model with the chosen number of identical repeats. (If the residue omitted prior to the SymmetryDocking step was a conserved one, the same side-chain will be re-introduced in the next step.)
5. Finally, the ancestral consensus sequences are mapped onto the symmetrical template backbone utilizing the python script for PyRosetta, ranking each sequence according to the predicted energy (*see* Fig. 2h).

6. The top scoring sequence and model should be visually inspected. Ideally there should be no obvious deviation from the template backbone, or internal voids. Numerous validation tools have been developed by structural biologists to assess the stability of a given protein model, and identify significant deviations from usual geometrical features. Any output model should be subjected to an array of these tests in order to identify possible high-energy distortions (*see* **Note 10**).
7. The sequences of the top-ranked models chosen for experimental testing are now back-translated into a suitable DNA sequence, codon optimized for expression in the chosen expression system. Numerous companies provide gene synthesis services and can assist with back-translation from a desired protein sequence. It may be helpful to introduce silent restriction sites to allow for simple deletion or introduction of one or more repeats from the coding sequence. These sites also allow simple cassette mutagenesis (*see* **Note 11**).

4 Notes

1. The following script is required for mapping the sequence on the protein backbone followed by scoring using PyRosetta. It can be downloaded from "http://www.riken.jp/zhangiru/software.html"

   To see a description of the available options:

   ```
   > python sequence_mapping.py –help
   ```

   To run with default options (single relaxation per sequence):

   ```
   > python sequence_mapping.py --backbone input_backbone.pdb --sequences input_sequences.fasta
   ```

   To run with multiple relaxations per sequence:

   ```
   > python sequence_mapping.py --backbone input_backbone.pdb --sequences input_sequences.fasta --nstruct n
   ```

   To run without any relaxation :

```
> python sequence_mapping.py --backbone input_backbone.pdb --sequences input_sequences.fasta --no_relax

The sequence_mapping.py source :
#!/usr/bin/env python

import os
import optparse
import re
from rosetta import *
from toolbox import *
```

```
one_to_three = {'A': 'ALA',
'R': 'ARG',
'N': 'ASN',
'D': 'ASP',
'C': 'CYS',
'E': 'GLU',
'Q': 'GLN',
'G': 'GLY',
'H': 'HIS',
'I': 'ILE',
'L': 'LEU',
'K': 'LYS',
'M': 'MET',
'F': 'PHE',
'P': 'PRO',
'S': 'SER',
'T': 'THR',
'W': 'TRP',
'Y': 'TYR',
'V': 'VAL',
}
def sequence_mapping(pdb_file, sequence_file, score_file, relax, jobs):
 if os.path.exists( os.getcwd() + '/' + pdb_file ) and pdb_file:
 init()
 pose = Pose()
 score_fxn = create_score_function('talaris2014')
 if (relax):
 refinement = FastRelax(score_fxn)
 pose_from_pdb(pose, pdb_file)
 if os.path.exists( os.getcwd() + '/' + sequence_file ) and sequence_file:
 fid = open(sequence_file,'r')
fod = open(score_file,'w')
data = fid.readlines()
fid.close()
sequences = []
read_seq = False
for i in data:
 if not len(i):
continue
elif i[0] == '>':
read_seq = True
fasta_line = re.split(':|\s+|\||\crn',i[1:])
name_cpt=0
            while (name_cpt<len(fasta_line) and not fasta_line[name_cpt]):
 name_cpt+=1
if name_cpt<len(fasta_line):
job_output = fasta_line[name_cpt]
else:
                print 'Error: Please enter an identifier for sequences in your fasta file'
```

```
 exit(1)
elif read_seq:
seq=list(i)
resn=1
for j in i:
if j!='\n' and resn<=pose.total_residue():
                    mutator = MutateResidue( resn , one_to_three[j] )
 mutator.apply( pose )
 resn+=1
 elif resn>pose.total_residue():
          print 'WARNING: couldn\'t mutate residue number
'+str(resn)+', sequence too long for backbone...'
 resn+=1
if (relax):
    jd = PyJobDistributor(job_output, jobs, score_fxn)
 jd.native_pose = pose
 scores = [0]*(jobs)
 counter = 0
 decoy=Pose()
 while not jd.job_complete:
decoy.assign(pose)
resn=1
refinement.apply(decoy)
jd.output_decoy(decoy)
scores[counter]=score_fxn(decoy)
counter+=1
 for i in range(0, len(scores)):
   fod.writelines(job_output + '_' + str(i+1) + ' :
'+str(scores[i])+'\n')
else:
        pose_packer = standard_packer_task(pose)
pose_packer.restrict_to_repacking()
        packmover = PackRotamersMover(score_fxn, pose_packer)
packmover.apply(pose)
        fod.writelines(job_output+' : '+str(score_fxn(pose))+'\n')
        pose.dump_pdb(job_output+'_1.pdb')
else:
print 'Bad fasta format'
exit(1)
 fod.close()
 else:
 print 'Please provide a valid sequence file, '+sequence_file+' doesn\'t exist'
 else:
 print 'Please provide a valid backbone file, '+pdb_file+' doesn\'t exist'

parser=optparse.OptionParser()
parser.add_option('--backbone', dest = 'pdb_file',
 default = '',
 help = 'the backbone in PDB format' )
```

```
parser.add_option('--sequences', dest = 'seq_file',
 default = '',
 help = 'the sequences to map' )
parser.add_option('--out', dest = 'score_out',
 default = 'scores.sc',
 help = 'the score file to output' )
parser.add_option('--clean', action="store_true", dest = 'clean_pdb',
 default = False,
 help = 'makes the pdb Rosetta friendly' )
parser.add_option('--no_relax',action="store_false", dest = 'relax',
 default = True,
 help = 'no relaxation after sequence mapping' )
parser.add_option('--nstruct', dest = 'jobs',
 default = '1',
 help = 'number of relaxations per sequence' )
(options,args) = parser.parse_args()
pdb_file=options.pdb_file
sequence_file = options.seq_file
score_file=options.score_out
clean_pdb=options.clean_pdb
relax=options.relax
jobs=int(options.jobs)
if clean_pdb:
  cleanATOM( pdb_file )
 sequence_mapping(pdb_file[:-4]+'.clean.pdb', sequence_file, score_file, relax, jobs)
else:
 sequence_mapping(pdb_file, sequence_file, score_file, relax, jobs)
```

2. This method is suitable for different symmetrical assemblies including globular symmetrical proteins such as the beta-propeller family, the TIM barrel family, trefoil family, and beta-plaits. However toroidal repeating proteins which do not have point symmetry, such as the ARM, ANK, and LRR proteins, can also be designed according this approach. For such proteins special attention should be paid to the backbone creation, and whether capping ends are required (*see* **Note** 3). In the case of nonrotational symmetry, the end domains will be more exposed to solvent than the central domains and this may require the introduction of more hydrophilic domains, or even a different fold, in order to produce a stable soluble protein. Such capping ends may therefore need to be excluded from the design of the central repeating motif.
3. While the protocol described above only utilizes a single protein for the generation of both the putative ancestral consensus sequences and the backbone template onto which they are mapped, it is perfectly suited to work with multiple parent

structures, as the domain structures and sequences are treated as individual input components.

4. At the alignment stage it may be a good idea to investigate the alignment of double repeats as well. From a structural point of view it may appear that every repeat is related to each other ($A_1A_2A_3\ldots$). However it is possible that in fact the structure is more properly considered as being constructed from tandem repeats of related motifs and can be described as $A_1B_1A_2B_2A_3B_3\ldots$, where A and B are similar but nevertheless distinct. An example of such a case is the ribonuclease inhibitor belonging to the Leucine Rich Repeat family. This can easily be identified from the phylogenetic tree where all the repeats are classified as belonging to one motif type or the other. Within the phylogenetic tree, capping domains at the N and C termini are also found to be more distant from the central repeats.
5. During the alignment step, small insertions or deletions will often be observed for a number of repeats. Care must be taken to prevent any register shift during the 3D sequence mapping and scoring procedure, which essentially requires that the sequences all match in a one-to-one fashion. Whether to remove or maintain any observed insertion or deletion is a fundamental decision in the backbone design, and all ancestral sequences must have an unambiguous mapping onto the backbone.
6. While correctly rooting a phylogenetic tree is important when analyzing genetic sequences from an evolutionary perspective, here the method was only utilized to create a variety of putative ancestral consensus sequences, which are later scored against the desired backbone. Variety is essential to cover a sufficient region of sequence-space to give a reasonable chance of finding a high scoring sequence. Therefore it is not necessary to obtain a rigorously accurate tree. Variety may be increased by utilizing homologous motif sequences from one or more repeats found in other proteins. In this case, all the input sequences should preferably be equally homologous.
7. Rather than working with only one repeat structure, multiple repeats can be created as well to create multiple backbone templates, each with minor differences, which can all be used for the mapping. In the final stage the best combination of backbone and sequence can be selected.
8. In a case where the selected template has an unusual residue (such as a much larger side-chain) compared to the consensus sequences, then this residue should be introduced by manual mutation before creating the perfectly symmetric backbone structure.

9. Another option to create a symmetric template backbone is to employ Rosetta Remodel. This is also a more suitable method if the fold does not exhibit point symmetry. For more information on this method, and how it has been used in the successful design of repeat proteins, we refer the reader to Parmeggiani et al. 2015 [9].
10. While the Rosetta scoring function is a good method for scoring the sequences on the 3D templates, other methods are available as well. The final structures must be inspected for internal voids (which do not contain conserved water molecules) either visually or utilizing RosettaHoles. The structures can be subjected to molecular dynamics simulations to test their structural integrity. Less time-consuming assessment of model quality include secondary and tertiary structure prediction from the sequence. Large deviations from the original experimental structure taken from PDB would require a restart with an adaptation of the protocol such as a different treatment of insertions/deletions at the alignment stage or the selection of a different template structure for the creation of the symmetric backbone.
11. During the back-translation step it is a good idea to diversify the coding sequence of each protein repeat as much as possible while avoiding codons likely to be highly detrimental to translation. This will assist the introduction of mutations by site-directed mutagenesis, and the sequencing of longer repeat genes. By introducing silent restriction sites at equivalent positions within the repeats, genes with a different number of repeats can be easily created by restriction digest and religation.

Acknowledgements

AV acknowledges RIKEN's program for Junior Scientists for the FPR fellowship and funding.

References

1. Goodsell DS, Olson AJ (2000) Structural symmetry and protein function. Annu Rev Biophys Biomol Struct 29:105–153. doi:10.1146/annurev.biophys.29.1.105
2. Caetano-Anolles G, Wang M, Caetano-Anolles D, Mittenthal JE (2009) The origin, evolution and structure of the protein world. Biochem J 417(3):621–637. doi:10.1042/BJ20082063
3. Jorda J, Xue B, Uversky VN, Kajava AV (2010) Protein tandem repeats—the more perfect, the less structured. FEBS J 277(12):2673–2682. doi:10.1111/j.1742-464X.2010.07684.x
4. Jorda J, Kajava AV (2010) Protein homorepeats sequences, structures, evolution, and functions. Adv Protein Chem Struct Biol 79:59–88. doi:10.1016/S1876-1623(10)79002-7
5. Kinch LN, Grishin NV (2002) Evolution of protein structures and functions. Curr Opin Struct Biol 12(3):400–408
6. Brych SR, Dubey VK, Bienkiewicz E, Lee J, Logan TM, Blaber M (2004) Symmetric primary and tertiary structure mutations within a symmetric superfold: a solution, not a

constraint, to achieve a foldable polypeptide. J Mol Biol 344(3):769–780. doi:10.1016/j.jmb.2004.09.060

7. Zhang J, Zheng F, Grigoryan G (2014) Design and designability of protein-based assemblies. Curr Opin Struct Biol 27:79–86. doi:10.1016/j.sbi.2014.05.009
8. Sawyer N, Chen J, Regan L (2013) All repeats are not equal: a module-based approach to guide repeat protein design. J Mol Biol 425 (10):1826–1838. doi:10.1016/j.jmb.2013.02.013
9. Pluckthun A (2015) Designed ankyrin repeat proteins (DARPins): binding proteins for research, diagnostics, and therapy. Annu Rev Pharmacol Toxicol 55:489–511. doi:10.1146/annurev-pharmtox-010611-134654
10. Park K, Shen BW, Parmeggiani F, Huang PS, Stoddard BL, Baker D (2015) Control of repeat-protein curvature by computational protein design. Nat Struct Mol Biol 22 (2):167–174. doi:10.1038/nsmb.2938
11. Parmeggiani F, Huang PS, Vorobiev S, Xiao R, Park K, Caprari S, Su M, Seetharaman J, Mao L, Janjua H, Montelione GT, Hunt J, Baker D (2015) A general computational approach for repeat protein design. J Mol Biol 427 (2):563–575. doi:10.1016/j.jmb.2014.11.005
12. Thomson AR, Wood CW, Burton AJ, Bartlett GJ, Sessions RB, Brady RL, Woolfson DN (2014) Computational design of water-soluble alpha-helical barrels. Science 346 (6208):485–488. doi:10.1126/science.1257452
13. Lanci CJ, MacDermaid CM, Kang SG, Acharya R, North B, Yang X, Qiu XJ, DeGrado WF, Saven JG (2012) Computational design of a protein crystal. Proc Natl Acad Sci U S A 109 (19):7304–7309. doi:10.1073/pnas.1112595109
14. Broom A, Doxey AC, Lobsanov YD, Berthin LG, Rose DR, Howell PL, McConkey BJ, Meiering EM (2012) Modular evolution and the origins of symmetry: reconstruction of a three-fold symmetric globular protein. Structure 20(1):161–171. doi:10.1016/j.str.2011.10.021
15. Lee J, Blaber SI, Dubey VK, Blaber M (2011) A polypeptide "building block" for the beta-trefoil fold identified by "top-down symmetric deconstruction". J Mol Biol 407(5):744–763. doi:10.1016/j.jmb.2011.02.002
16. Paoli M (2001) Protein folds propelled by diversity. Prog Biophys Mol Biol 76 (1–2):103–130
17. Voet AR, Noguchi H, Addy C, Simoncini D, Terada D, Unzai S, Park SY, Zhang KY, Tame JR (2014) Computational design of a self-assembling symmetrical beta-propeller protein. Proc Natl Acad Sci U S A 111 (42):15102–15107. doi:10.1073/pnas.1412768111
18. Voet AR, Noguchi H, Addy C, Zhang KY, Tame JR (2015) Biomineralization of a cadmium chloride nano-crystal by a designed symmetrical protein. Angew Chem Int Ed Engl. doi:10.1002/anie.201503575R1
19. Delano WL (2010) The PyMOL molecular graphics system, version 1.3. Schrödinger, LLC
20. Pettersen EF, Goddard TD, Huang CC, Couch GS, Greenblatt DM, Meng EC, Ferrin TE (2004) UCSF Chimera—a visualization system for exploratory research and analysis. J Comput Chem 25(13):1605–1612. doi:10.1002/jcc.20084
21. Gille C (2005) STRAP. http://www.bioinformatics.com/STRAP
22. Retief JD (2000) Phylogenetic analysis using PHYLIP. Methods Mol Biol 132:243–258
23. Ashkenazy H, Penn O, Doron-Faigenboim A, Cohen O, Cannarozzi G, Zomer O, Pupko T (2012) FastML: a web server for probabilistic reconstruction of ancestral sequences. Nucleic Acids Res 40(web server issue):W580–W584. doi:10.1093/nar/gks498
24. Kaufmann KW, Lemmon GH, Deluca SL, Sheehan JH, Meiler J (2010) Practically useful: what the Rosetta protein modeling suite can do for you. Biochemistry 49(14):2987–2998. doi:10.1021/bi902153g
25. Chaudhury S, Lyskov S, Gray JJ (2010) PyRosetta: a script-based interface for implementing molecular modeling algorithms using Rosetta. Bioinformatics 26(5):689–691. doi:10.1093/bioinformatics/btq007
26. Andreeva A, Howorth D, Chothia C, Kulesha E, Murzin AG (2015) Investigating protein structure and evolution with SCOP2. Curr Protoc Bioinformatics 49:1.26.1–1.26.21. doi:10.1002/0471250953.bi0126s49

Chapter 17

A Protocol for the Design of Protein and Peptide Nanostructure Self-Assemblies Exploiting Synthetic Amino Acids

Nurit Haspel, Jie Zheng, Carlos Aleman, David Zanuy, and Ruth Nussinov

Abstract

In recent years there has been increasing interest in nanostructure design based on the self-assembly properties of proteins and polymers. Nanodesign requires the ability to predictably manipulate the properties of the self-assembly of autonomous building blocks, which can fold or aggregate into preferred conformational states. The design includes functional synthetic materials and biological macromolecules. Autonomous biological building blocks with available 3D structures provide an extremely rich and useful resource. Structural databases contain large libraries of protein molecules and their building blocks with a range of sizes, shapes, surfaces, and chemical properties. The introduction of engineered synthetic residues or short peptides into these building blocks can greatly expand the available chemical space and enhance the desired properties. Herein, we summarize a protocol for designing nanostructures consisting of self-assembling building blocks, based on our recent works. We focus on the principles of nanostructure design with naturally occurring proteins and synthetic amino acids, as well as hybrid materials made of amyloids and synthetic polymers.

Key words Nanostructures, Self-assembly, Peptide-based nanodesign, Synthetic amino acids, Beta-helical proteins, Computational nanodesign, Amyloid peptides, Hybrid materials

1 Introduction

Nanotechnology aims to design novel materials and molecular devices, often by exploiting the natural ability of molecules to self-assemble into larger, ordered structures at the nanoscale. Nanotechnology applications include targeted drug delivery systems, computational devices, and scaffolding tissues [1–3]. In nature, protein domains often self-assemble, spontaneously organizing in stable higher-order structures through noncovalent interactions. These molecules may create large complexes of well-defined structures and functions. The shapes, sizes, and functions of these structures are determined by the amino acid sequence of these

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_17,

proteins [4, 5]. In recent years there has been much focus on the experimental and computational design of self-assembled nanomaterials based on the self-assembly properties of proteins. Exploiting the natural ability of macromolecules to self-assemble can be a very useful approach in the design and construction of novel molecular structures [5–8]. Much work has been done in recent years in the design and construction of nanostructures using DNA, RNA, and protein segments [9–14]. Advances in peptide synthesis and molecular engineering techniques have made self-assembly of peptide segments a favorable route by which to obtain nanostructures [5, 15–17], particularly those consisting of single or associated tubes, fibers, and vesicles.

Computational methods have become a powerful tool in nanobiology and nanostructure design. The use of advanced simulation methods and efficient modeling algorithms, in addition to the rapidly increasing amount of data in DNA, RNA, and protein databases, can considerably accelerate the design process via fast probing of many possible models in a high-throughput cost-effective way, aiming to experimentally test only feasible models. In this chapter we describe a computational and experimental protocol, based on our previous and current work. We first introduce a protocol for designing self-assembled nanostructures from naturally occurring protein motifs, followed by structural enhancement by synthetic amino acids. Next, we introduce a related method to construct nanostructures based on amyloid peptides. Finally, we introduce a protocol to design hybrid materials based on the conjunction of functional amyloids and synthetic polymers.

2 Computational Nanodesign

Construction of stable nanostructures using natural building blocks is a reasonable and promising strategy toward precisely and quantitatively controlling the supramolecular assemblies. A building block is a well-defined secondary structural unit which, if cut from the protein chain and placed in solution, is still likely to have a conformation similar to the one it has when embedded in the native protein structure. The Protein Data Bank (PDB) is populated by an extensive repertoire of building blocks, with different shapes, sizes, and chemical properties which can be used in rational design of protein-based nanostructures [18]. Some naturally occurring proteins contain a tubular or fibrillar motif in their folds. A good example of tubular proteins is the β-helix protein fold. The fold of β-helical proteins contains a repetitive helical strand-loop motif [19], where each repeat contributes a strand to one or more parallel β-sheet(s). The left-handed β-helical fold is especially suitable: the tubular structure is regular and symmetrical and is often stabilized by a network of interactions between similar residues in consecutive coils [20] (*see* Fig. 1).

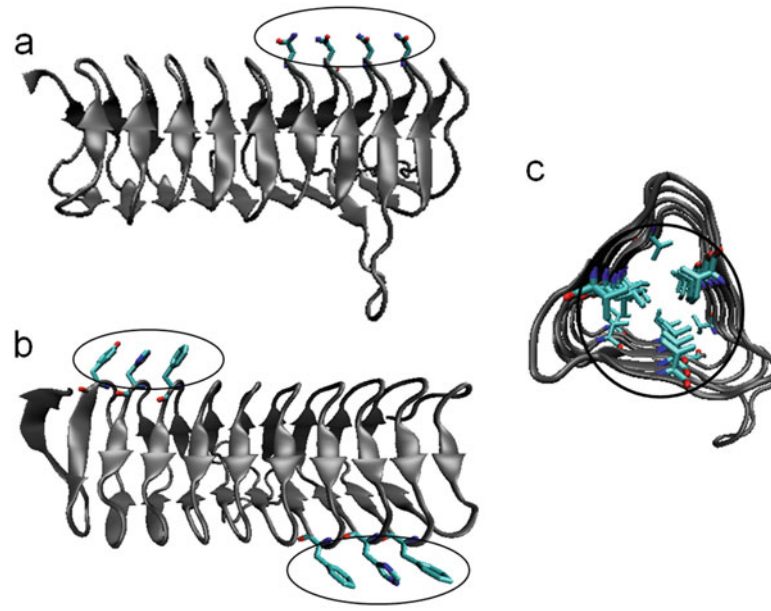


Fig. 1 An example of β-helical proteins and their typical interactions. (**a**) An example of an asparagine ladder. (**b**) An example of aromatic ring stacking. (**c**) An example of hydrophobic residue interactions

The common types of interactions in β-helices include:

1. Asparagine (or glutamine) ladders that stabilize the helical structure through hydrogen bonds between residues in consecutive rungs (Fig. 1a).
2. Stacking of aromatic (Phe, Tyr, His) and aliphatic (Pro) rings (Fig. 1b).
3. Hydrophobic interactions (especially Val, Ile, Leu) (Fig. 1c).

The tubular nature of left-handed β-helical proteins makes them excellent candidates to be used as building blocks to construct fibrillar or tubular nanostructures without the need to perform many structural manipulations. In addition, their helical and symmetric structure makes them good candidates to be excised and tested as modules.

In our work [21] we presented a general approach to the design of nanostructures based on the potential assembly property of protein segments, in which the segments are taken from naturally occurring proteins and have preferred conformational tendencies. We designed nanoconstructs based on left-handed β-helical proteins by selecting short (two turns), repetitive motifs and extracting the corresponding coordinates from the PDB [22]. We assembled copies of the motifs on top of one another so that the assembled nanotube had an almost perfect equilateral triangular shape, with each side being ~18 Å. We simulated the nanostructures using

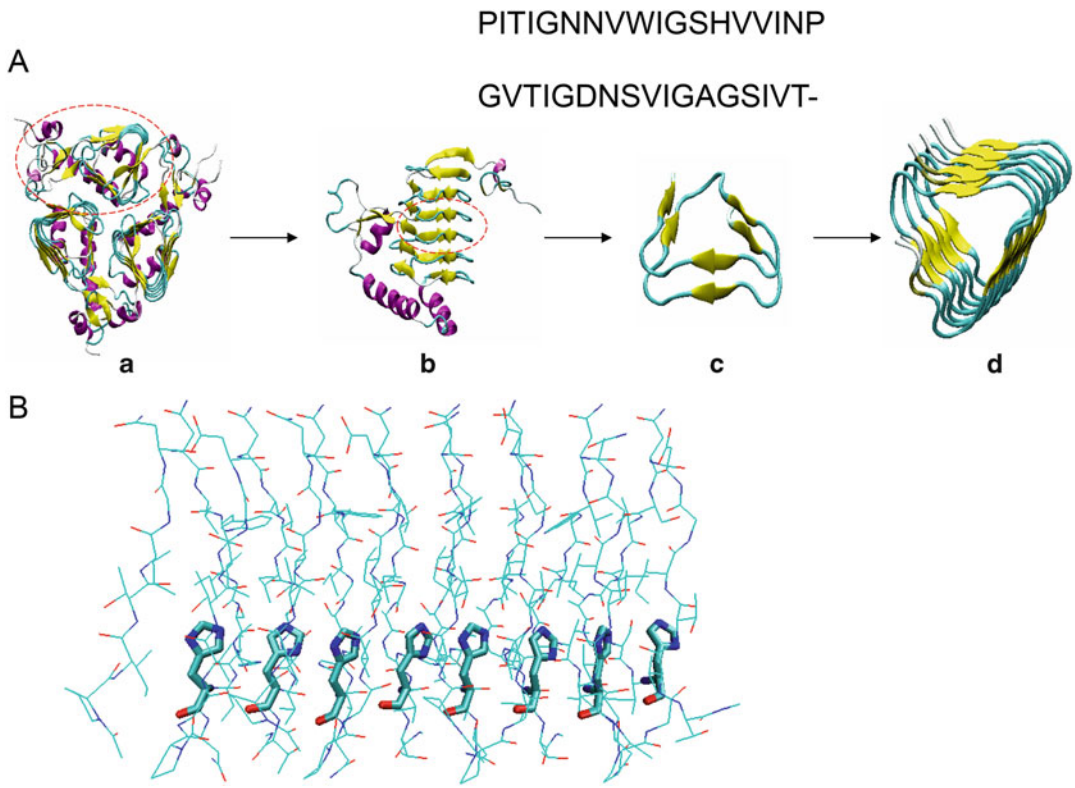


Fig. 2 A schematic procedure of the construction of a nanotube using the naturally occurring protein building block from a β-helix (taken from galactoside acetyltransferase, PDB code: 1KRR). *Top*: (**a**) The trimeric crystal structure of galactoside acetyltransferase (GAT) from *E. coli*, with three left-handed parallel β-helix domains. (**b**) The monomeric structure of GAT (*circled*) taken from the trimeric GAT structure. (**c**) A single building motif (*circled*) taken from the monomeric 1KRR structure with selected residues 131–165. (**d**) A nanotubular structure obtained by stacking four repetitive building motifs on top of each other. *Bottom*: An example of one of the Histidine mutants

molecular dynamics (MD) to test their structural stability over time. Figure 2 illustrates a schematic flowchart of the process. *Our design principle* is that if the nanostructures can preserve their tube organization and motif association during the simulations, they are promising candidates for experiment. Otherwise, if the nanostructures cannot preserve their original organization in the simulations, they are unlikely to preserve their organization in experiment as well. For those unstable nanostructures, in a subsequent stage (*see* next sections) we reduced the conformational freedom by introducing restricted synthetic residues in strategic positions to improve the structural stability of the designed nanostructures. For a successful design, it is desirable that the substitutions retain both favorable packing interactions and hydrogen bonding with the neighboring residues. Of the 17 systems that we tested, the construct based on the assembly of copies of residues

131–165 of galactoside acetyltransferase from *E. coli* (PDB code: 1krr, chain A) was very stable over the simulation time under all of the tested temperature and ionic strength conditions. Figure 2 shows the sequence and structure of the 1krr system. To assess the structural stability of our tested models, we looked at the retention of the structural organization over time. We largely focused on the organization of the loop regions since these are typically the least stable. We also studied the effect of specific amino acids and chemical interactions on the conformational stability of the structure, focusing again on loop regions. Through mutational study, we found that apart from the characteristic inter-strand interactions of β-helical proteins, the presence of proline residues around the loop areas greatly contributes to the retention of the loop structure and hence to the stability of the overall conformation. In addition, in many cases we found a relatively large number of glycines in loop regions. These glycines were involved in hydrogen bonds with the side chains of other residues in their vicinity and hence contributed to maintaining the conformation of the loops. We next aimed to further enhance the stability of the system by inserting specific point mutations using noncoding amino acids whose structures are available. The choice of such conformationally restricted residues and the positions of insertion were guided by our mutational observations on the stabilizing effects of naturally occurring residues on the entire system. This work is described in more details below.

We next aimed to modify the 1krr-based nanostructure [23] toward useful biological functions. The original construct is characterized by an internal hydrophobic core, containing mainly valine and isoleucine residues, rendering it inappropriate for the transfer of matter or charge. Since the hollow space inside the structure was narrow and unsuitable for the transport of large molecules, charge transfer seemed to be a feasible application. However, to allow charge transfer we had to modify the chemistry of the internal core of the structure. Charge can be transferred through π-electron stacking or through H1 transfer. Although charge transfer cannot be modeled through classical mechanics, this methodology was appropriate to assess its feasibility. We considered two different scenarios for charge transfer: (1) formation of ladders of π-electron-rich functional groups by substituting some of the original residues in the interior of the construct by other residues capable of π-stacking; and (2) the generation of proton transfer environment through a network of salt bridges, reminiscent of the serine protease catalytic triad. The two necessary conditions to achieve this goal are:

1. The mutated structures must still retain their tubular structures in the simulation.
2. There must be a side-chain distribution that allows these chemical processes.

To create a ladder of π-stacking residues and to test whether this would affect the structural organization, we inserted a row of histidine residues in each of the three beta-sheets, one sheet at a time (*see* Fig. 2b for an illustration of an example, the histidine mutant) and simulated the mutated structures. Histidine is an aromatic amino acid capable of π-stacking. Its side chain is fairly similar to the size of valine and isoleucine so no drastic steric hindrance or structural changes were expected to occur. The pKa of histidine is six, so in a physiological pH it can assume both a neutral and a charged form with a relatively high probability. Moreover, its neutral form corresponds to equilibrium between two states: one with d hydrogen (ND) protonated and the other with e hydrogen (NE) protonated. This allowed us to test different possible combinations of ionization states. Naturally, we could not sample all possible ionization states due to computational time limitations, but we tried to sample as many as possible as well as different combinations of ionization states. We identified a position where, despite the insertion of histidine, the simulated nanostructures retained their initial organization to a reasonable extent and created a configuration made of networks of neutral histidine, charged histidine, and aspartate, imitating the serine protease catalytic triad. We suggested a structure with a π-stacked row of alternatively neutral and charged histidine residues that interact with a row of aspartate residues through salt bridges, and thus provided the conditions for creating a nanosystem potentially capable of charge transfer.

2.1 Nonstandard Amino Acids

The catalog of amino acids available nowadays for materials sciences applications has rapidly expanded. Only in natural structures, there can be found more than 700 different amino acids [24, 25] (most of them, also L-amino acids) beyond the 20 genetically coded L-amino acids that are contained in proteins. Furthermore, many others have been imagined and synthesized by organic chemists [24–28]. All those compounds are named under a common designation of nonproteinogenic or noncoded amino acids (nc-aa). Although they are not involved in ribosomal synthesis of native peptides and proteins, several naturally occurring peptides and proteins contain nc-aa [29, 30]. The majority are the results of post-translational modifications that active proteins undergo upon release from the ribosome. Most of these chemical modifications have been found to play crucial roles in both the regulation of metabolic routes and genomic expression. The part played by citrullination (conversion of arginine residue to citrulline) in the relaxation of chromatin and the modulation of the pluripotency of stem cells is especially noteworthy [26].

Currently the use of such molecules stretches over almost all fields of applied natural sciences. They are applied to improve the pharmacological profile of natural peptides endowed with

biological activity [31, 32] (to confer resistance against enzymatic degradation, enhance membrane permeability, or increase selectivity and affinity for a particular receptor), and are responsible for the development of nonpeptidic drugs [33]. On the other hand, nc-aa have recently been used in biotechnology for protein reengineering [34, 35]. Thus, proteins containing such residues can acquire new chemical features such as fluorescence [36, 37], redox-activity [38], photosensitivity [37, 39], and specific chemical reactivity [40]. Those new spectroscopic properties [41] can be used as biosensors, spectroscopic or biophysical probes, or even for building new nanosystems for drug delivery and diagnosis through imaging to be used in medicine [34, 40, 42, 43]. Other applications of Nc-aa are their use in nanobiology to promote the self-assembly of nanostructures [44, 45] or for developing bioinspired synthetic organic polymers that emulate the shape and properties of natural peptides and proteins [46, 47].

2.2 Structurally Restricted Amino Acids

Practical use of nc-aa is frequently hampered by the high degree of dispersion that their relevant conformational data present. The most accurate information is extracted from first principle calculations, which are typically reported in physical chemistry journals, whereas their synthetic details are dispersed among specialized journals of organic chemistry. Moreover, spectroscopic and structural studies of small peptide containing nc-aa are generally performed by organic and peptide chemists and their findings are away from biology specialized journals. However, most applications of nc-aa are developed and tried by researcher working on fields related with medicine, protein science, or materials engineering. The lag of systematically correlated information and the great potential applicability of nc-aa led us to integrate these diverse existing contributions into a unified and simple informatics tool that should facilitate the universal use of nc-aa in practical applications. This new database contains the conformational descriptors of any nc-aa ever studied and any relevant bibliographic information about already reported practical uses. The NCAD (Non-Coded Amino acids Database) [48] is a database designed to identify the most suitable nc-aa for any given structural motif, compatibility required for any use in both life and materials sciences. Our tool integrates all structural and energetic descriptors previously reported using quantum mechanics calculations for each nc-aa. A summary of the information integration in the database is presented in Fig. 3. Per each amino acid NCAD contains its complete structural profile, which includes a detailed description of each minimum energy conformation (dihedral angles, three-dimensional structure, relative energy, etc.) and, if available, all the bibliographic information related to experimental data, included all reported applications (both in bioscience and materials science).

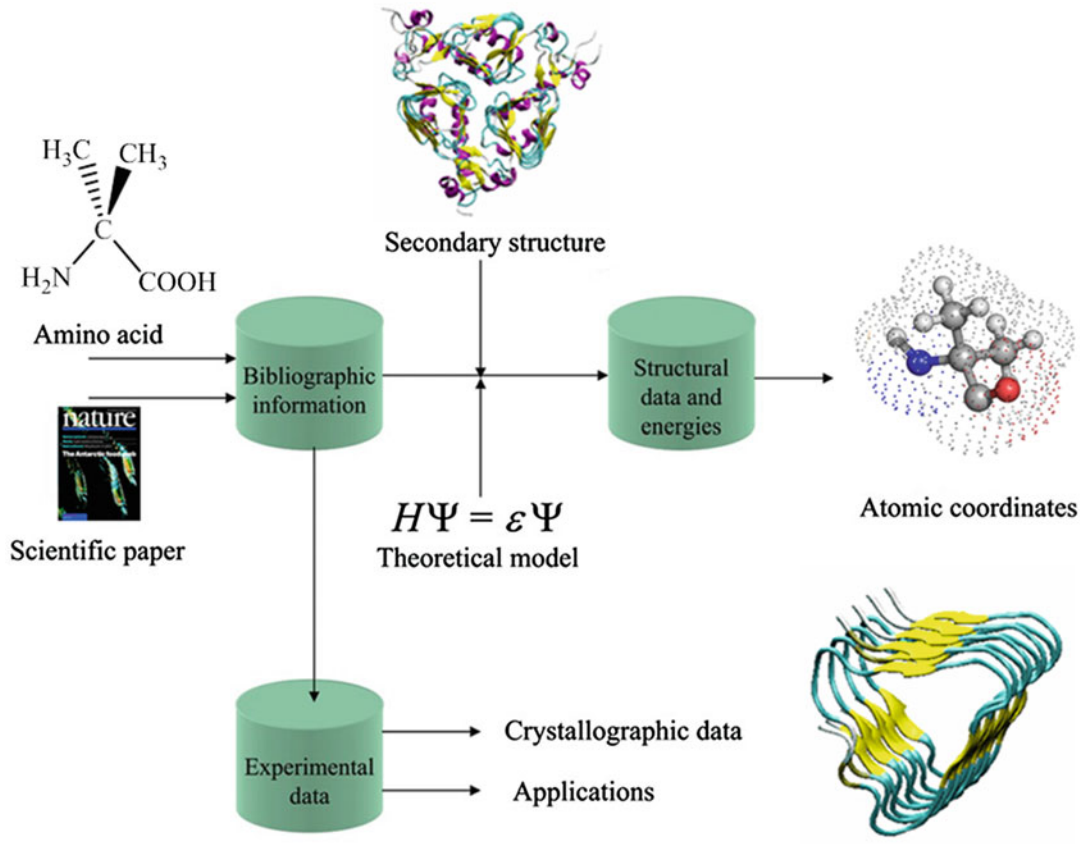


Fig. 3 Schematic representation of the information flown in the NCAD database

Since simulations can quickly probe many models and provide potentially good candidate nanostructures in terms of structural stability and minimum free energy for experimental test, they could accelerate the design process.

2.2.1 Application to β-Helices Motifs

To enhance the thermodynamic stability of a given β-helical repeat sequence, we engineered chemically constrained residues with backbone conformational tendencies similar to those of natural amino acids in the most mobile (loop) regions. Among the synthetic residues that our group has prepared and studied, here we focused on 1-aminocyclopropanecarboxylic acid (Ac_3c), a simple cyclic *R*,*R*-dialkylated amino acid with strong stereochemical constraints induced by the highly strained cyclopropane ring. We also tested its double-phenyl derivative, 1-amino-2,2-diphenylcyclopropanecarboxylic acid (c_3Dip), a cyclopropane analogue of phenylalanine bearing two germinal phenyl rings. However, this substitution was unsuccessful, due to the steric effects induced by the residue side chain size.

2.2.2 Survey of Ac_3c Derivatives

Ac_3c is the simplest achiral C-α-tetra-substituted α-amino acid with Cα↔Cα cyclization (Fig. 4). The stereochemical constraints of this amino acid are produced by the unfavorable steric interaction of the two β-methylene groups and by the three-membered ring rigidity. The conformational preferences of Ac_3c were characterized by energy computations of the monopeptide [49–51] and X-ray diffraction analyses [52–55] of a variety of peptides of this residue up to the tetramer level. These studies illustrated that the Ac_3c amino

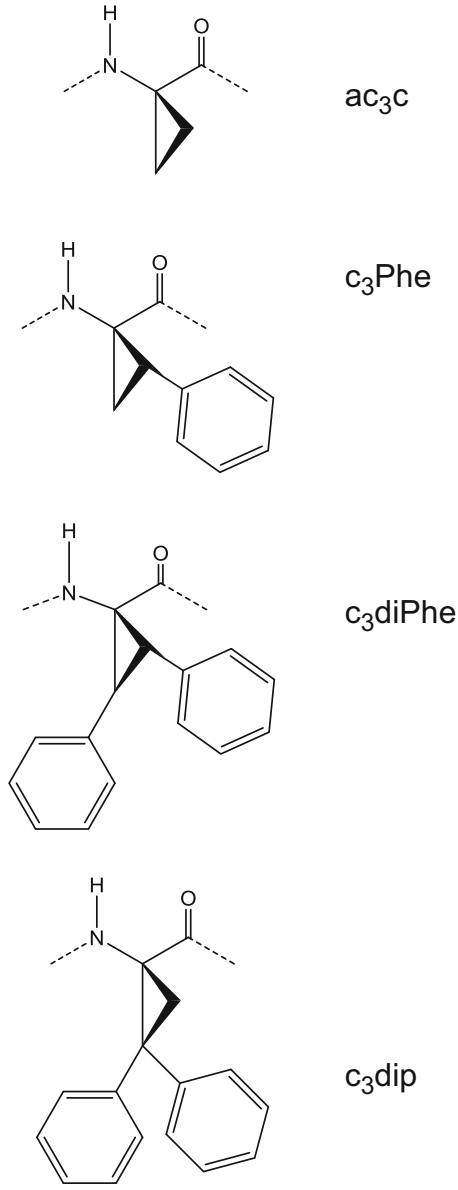


Fig. 4 Illustration of the noncoding amino acids used in this study

acid prefers the "bridge" region of the Ramachandran map, i.e., φ, $\psi \approx \pm 80°, 0°$, which corresponds to position $i + 2$ of type I/I′ and type II/II′ β-turns. Theoretical studies indicated that the tendency of Ac_3c to adopt a small value of ψ is due to the hyper-conjugation between the lone pairs of the carbonyl oxygen of the residue and some adjacent molecular orbitals associated with the C-β-C-β′ bond [56]. This conjugative ability of the Ac_3c cyclopropyl moiety was demonstrated by X-ray crystallography. The N-Cα and Cα-C bond lengths are significantly shortened compared to Cα-tri-substituted and Cα-tetra-substituted α-amino acids [57], and the mean exocyclic N-Cα-C bond angle is significantly larger (116–118°) than the tetrahedral angle (109.5°). Thus, the strong tendency of Ac_3c to adopt β-turn conformations is enhanced by specific intra-residue electronic interactions.

Incorporation of selectively oriented side-chain substituents into conformationally restricted amino acids allows increased control of the backbone fold [58]. Cyclopropane analogues of phenylalaline are particularly attractively because the rigidly oriented phenyl side groups may interact with the backbone sterically and electronically through the aromatic π-orbitals [53, 55, 59]. The side chain orientation of 1-amino-2-phenylcyclopropanecarboxylic acid (c_3Phe) stereoisomers drastically affects the backbone conformational preferences, with a tendency to adopt folded conformations [56, 60, 61]. This tendency was observed in the stereoisomers of 1-amino-2,3-diphenylcyclopropanecarboxylic acid with the phenyl substituents in a *trans* relative disposition (c_3DiPhe) [59, 62] in both solid state and solution. A cyclopropane analogue of phenylalanine bearing two geminal phenyl side substituents was recently incorporated into Pro-c_3Dip. X-ray diffraction analysis showed that the (*S*)-Pro-(*R*)-c_3Dip stereoisomer adopts two consecutive γ-turns stabilized by intramolecular hydrogen bonds [63]. The ability of c_3Dip to adopt a γ-turn and to induce this structural motif in neighboring amino acids was explained by calculations [64]. The dihedral angle ψ values for all cyclopropane analogues of phenylalanine are close to 0° due to the presence of hyperconjugative effects [56, 62, 64]. It is worth noting that interesting supramolecular structures have been characterized for peptides rich in c_3Dip [65, 66]. Here we focused on Ac_3c, the simplest Cα-tetra-substituted cyclic *R*-amino acid promoting β-turn-type conformations, and c_3Dip (Fig. 4), in which the Ac_3c conformational preferences are guided toward the γ-turn. Force field parameters for Ac_3c and its derivatives were explicitly developed [50, 51].

To test our design principle, we built two nanotubes using two different motifs of 1krr and 1hv9, both of which adopt similar left-handed β-helical conformation. When submitting the two nanotubes to MD simulations, it can be seen clearly in Fig. 5 that the

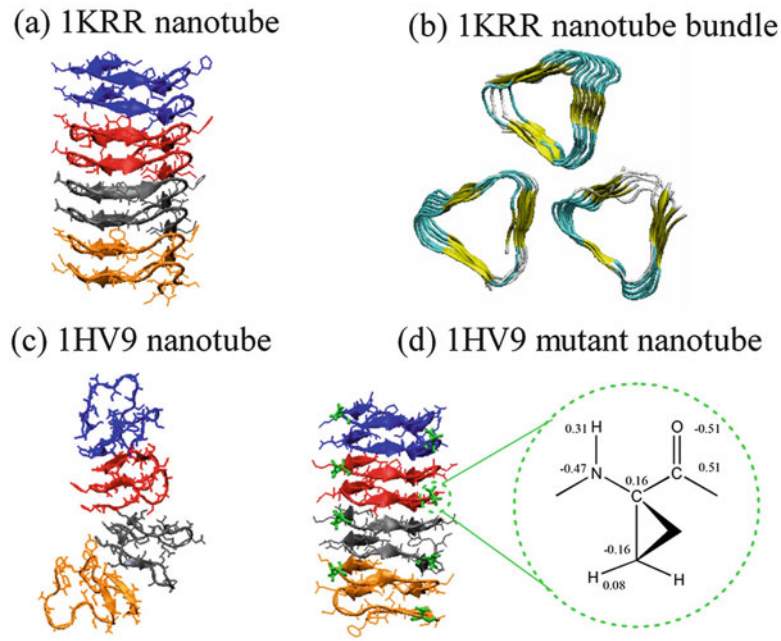


Fig. 5 MD simulations of different nanotubes constructed by left-handed β-helical motifs. (**a**) 1krr nanotube and (**b**) 1krr nanotube bundle display high structural stability. (**c**) 1hv9 nanotube is not structurally stable. (**d**) Introduction of the conformationally restricted Ac3c residue in loop regions greatly enhances the stability of 1hv9 nanotube. Ac3c is displayed as a *green stick*

1krr nanotube with four repeat β-helical units can well preserve its original tubular structure and display high structural stability (Fig. 5a, b). Moreover, 1krr nanotube bundle—containing three β-helical segments forming a trimeric structure along a threefold screw axis—can retain both the individual tubular structure and the overall trimeric structure. Conversely, the 1HV9 nanotube completely lost the initial, compact nanotubular structure, and all β-helical units started to separate from each other (Fig. 5c). Further structural analysis has determined the most unstable residues at the turn region. Based on our design principle, we replaced two turn residues of Asn5 and Ala27 with the conformationally restricted 1-aminocyclopropanecarboxylic acid (Ac_3c) residue, and the mutated 1hv9 nanotube were able to retain its original tubular structures and displayed very high structural stability (Fig. 5d). Compared to the unstable wild-type 1hv9 nanotube, the enhanced stability originates not only from the increasing number of hydrogen bonds and hydrophobic contacts between each building subunit, but also from the reduced flexibility in the loop regions induced by Ac_3c within each building subunit. Thus, the Ac_3c geometrical confinement effect is sequence-specific and position-specific.

2.2.3 Simulation Protocol

Calculations were performed by using the NAMD package [67]. All of the atoms of the system were considered explicitly, and the energy was calculated by using the CHARMM22 force field [68]. Water molecules were represented explicitly, by using the TIP3 model [69]. The simulations were performed by using the NVT ensemble in an orthorhombic simulation box. We chose constant volume simulations because all of the trajectories were obtained at high temperature. By these means, we could ensure that proper density distribution would not be lost due to thermal effects. Periodic boundary conditions were applied by using the nearest image convention. The box size was adjusted to fit the complex size, so that infinite dilution conditions would be maintained. The box dimensions were adjusted to ($50 \times 50 \times 70$ Å^3) to ensure infinite dilution. Each system contained approximately 15,000–20,000 atoms, including the solvent. The starting molecular structures were built by using the INSIGHTII molecular package (2000, Accelrys, San Diego, CA). For any given arrangement, we fixed the inter-turn distance of adjacent repetitive units to match the inter-strand distance within each unit, which was approximately 4.5 Å. The charge of all potential titratable groups was fixed to those values corresponding to neutral pH, such that all aspartic acid side chains were represented in their anionic form and all lysine side chains in their acidic positively charged form. Both peptide edges were capped to avoid interactions between adjacent termini.

2.2.4 The Simulation Conditions

We performed the simulations under the following conditions:

1. No ions in the solution, 300 K.
2. No ions in the solution, 360 K.
3. Ionic strength of 0.23 % w/w, 300 K (approx. 8 ions).
4. Ionic strength of 0.5 % w/w, 300 K (approx. 16 ions).
5. Ionic strength of 0.8 % w/w, 300 K (approx. 24 ions).

In the case of ionized solution, we kept the overall charge of the system neutral for the use of EWALD particle mesh summation [70] to calculate the electrostatic charges. The ions were chloride and sodium.

Before running each molecular dynamics simulation, the potential energy of each system was minimized by using 5000 conjugate gradient steps. The heating protocol included 15 ps of increasing the temperature of the system from 0 K to the final temperature of 300 K (or 18 ps of increasing the temperature from 0 to 360 K) plus 100 ps of an equilibration period. We perform the simulations at 300 and 360 K in order to enhance the stability differences between the models by means of thermal stress. Furthermore, using high temperature allowed us to infer some kinetic tendencies. Residue-based cutoff was applied at 14 Å,

i.e., if any two molecules have any atoms within 14 Å, the interaction between them is evaluated. A numerical integration time step of 1 fs was used for all of the simulations. The nonbonded pair list was updated every 20 steps, and the trajectories were saved every 1000 steps (1 ps) for subsequent analysis. Each simulation was run for a period of 20 ns. Potentially stable systems were run for an additional 20 ns.

We have used this protocol for a few years [21, 23, 71–76] and observed that its results correspond to experimental observations [77–80].

2.2.5 Structural Analysis

We calculated the structural conservation in the following ways:

- Conservation of the size of the structure with respect to the minimized structure: the trajectories were aligned with the initial structure, and the RMSD was calculated with respect to C-α atoms.
- Conservation of the loops was defined as the RMSD of the Cα of each residue of the loop with respect to the initial minimized structure. In addition, the distribution of the backbone dihedral angles was plotted.
- Sequence alignment and analysis were performed with the CLUSTALX software [81].

3 Introduction to Hybrid Materials Based on Amyloid-PLA Conjugates

Hybrid materials are one of the most active areas in biomaterials science. This is because by combining different types of molecules it is possible to merge their properties into new useful chimeric compounds. In the particular case of peptide-polymer conjugates, which result from the covalent integration of a peptide with a synthetic polymer block, they are especially attractive because this kind of hybrid macromolecules combines unique properties that come from the precise chemical structure and functionality of peptides and the stability, functions, and processability of synthetic polymers [82, 83]. The conformational profile of peptide-polymer hybrid compounds has a crucial importance due to its influence on many other parameters such as binding affinity and bioactivity. The conformational landscape of new conjugated macromolecules cannot be understood only in terms of a simple addition of their parts; rather, the dynamic interactions between them should also be considered. Thus, conformational exploration needs to be carried out for the whole system and for its separate components, and the results have to be compared. The huge number of feasible combinations of the conformational states of each of the molecular

components dramatically increases the complexity of the problem. Theoretical chemistry tools provide a feasible approach for conformational exploration, since they allow performing the search in a faster, more efficient manner. Diblock copolymers that covalently link proteins and synthetic polymers are among the most promising chimeras, being the subject of intense research using both synthetic and theoretical approaches.

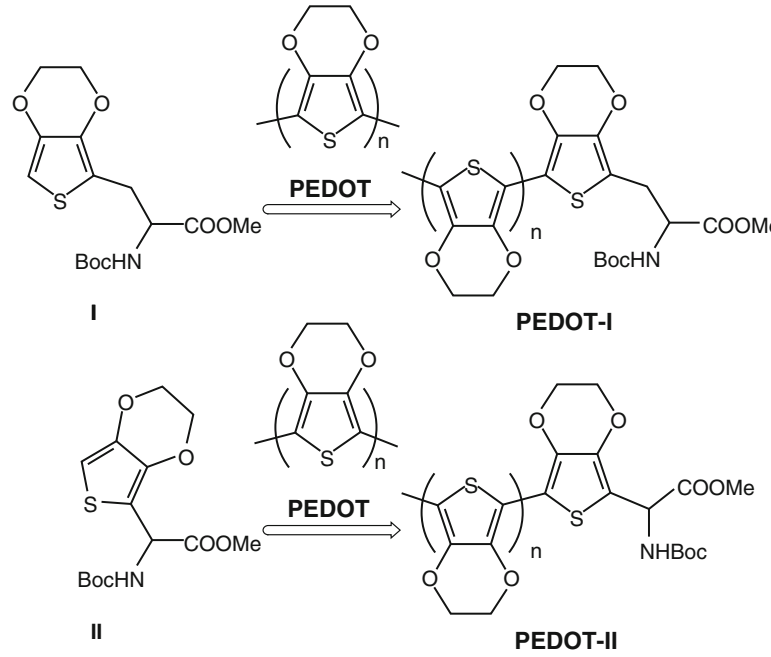
Self-aggregating proteins are found in several pathological processes and are also a target in material science due to their ability to spontaneously form ordered materials with useful physicochemical and mechanical properties [84]. Arginine-Vasopressin (hereafter Vas) and Neuromedin-K (also known as Neurokinin B, and hereafter abbreviated Neuro) are among those peptides that are known to self-aggregate. Vas is a peptidic human hormone involved in the pathogenesis of neurohypophyseal diabetes insipidus (NDI) by aggregating into amyloid-like microfibrils; Neuro is a member of the tachykinins protein family that plays an important role as a neurotransmitter and neuroregulator with the ability to form fibrils resembling amyloids [84]. Neuro has been shown to decrease neuronal damage caused by beta-amyloid protein aggregation by interfering in this molecular process. These two peptides have an intrinsic ability to form self-aggregating self-structured biomaterials both in vivo and in vitro; however, immunological problems may arise due to their proteinogenic nature. On the other hand, poly(*R*-lactic acid) (*R*-PLA, also known as PDLA) is a semicrystalline biodegradable and biocompatible polyester that has physicochemical properties suitable for making release-controlled systems and tissue engineering scaffolds. These features make *R*-PLA a suitable candidate for introducing biocompatible components by conjugating it to these other molecules. Formation of hybrid conjugates by combining peptides (and proteins) with synthetic polymers result in chimeras (i.e., artificial biomolecules) with a useful set of features, where each component has different sets of properties. The capability of peptides and proteins to self-organize into supra-molecular arrangements complements the inherent tendency of *R*-PLA to similarly self-organize at the supra-molecular level. This polyester has a crystallinity of around 37 %, a glass transition temperature between 60 and 65 °C and a melting temperature between 173 and 178 °C. The fusion of such properties may lead to novel macromolecules capable of self-aggregation and self-organizing while preserving the key properties of biodegradability and biocompatibility [85].

In a recent work [86] we used computational methods to characterize the conformational preferences of two new hybrid materials derived from the conjugation of Vas and Neuro to a 150 residues-long *R*-PLA chain. Determination of the influence of the polymer component on the conformational preferences of the peptide component is a key question for peptide-mediated

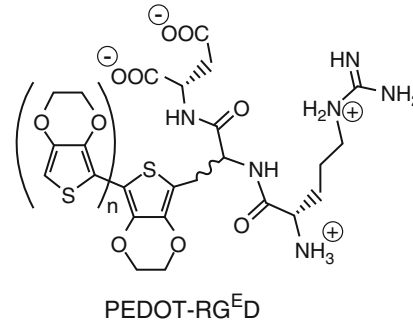
self-aggregation, since the conformation of the peptide has a strong impact on such processes. Our study focused on the hypothesis that noncovalent self-aggregation involving tight binding to a hairpin-like backbone conformation enables amyloid formation. Thus, the conformational profiles of the free peptides were first assessed so it can be compared with the conformational profile of the polymer-linked peptide. The comparison between the free peptides and the peptides linked to a model polymer provided an initial benchmark for studying novel potentially self-aggregating materials. Our approach relied on the premise that similar conformational behavior of the free and the polymer-linked peptides, is expected to lead to similar properties. Next we investigated the properties of the polymer when isolated and when conjugated to the peptides to ascertain that it also retains its global properties. Here, we briefly describe the conformational characterization of two amyloidogenic peptides and two new chimeric molecules combining the properties of amyloidgenic peptides and polymers. The study of these specific cases allowed us to model new peptide-polymer chimeras based on general trends observed in studies such as the one presented here. This work set the path for further theoretical and experimental work not only to address the peptide and polymer self-aggregation but also to develop new biomaterials with advanced properties.

Details of the preparation and characterization at the molecular level of the peptide-polymer conjugates resulting from the combination of FF and poly(L-lactide) (PLA), hereafter denoted FF-PLA were described elsewhere [87]. Conjugates based on biodegradable PLA, which is obtained from renewable resources, are expected to present important medical and biotechnological applications. PLAs are produced by ring-opening polymerization (ROP) of lactides and the lactic acid monomers used are obtained from the fermentation of sugar feed stocks. The different stereoisomeric PLA grades, which are produced from L-, D-, and D,L-lactides, can be used in biomedical devices (e.g., scaffolds and drug delivery systems) in which they slowly hydrolyze back to lactic acid and reenter the Krebs cycle. Fan et al. [88] described the synthesis of L-phenylalanine-terminated PLA, F-PLA, using a three-step process: (1) hydroxyl-terminated PLA was synthesized through the ROP of L-lactide; (2) the hydroxyl end group of PLA was blocked with Boc-L-phenylalanine; and (3) the free amino end group was obtained by removal of the *t*-butoxycarbonyl group. The resulting F-PLA conjugate was employed as macroinitiator for the synthesis of poly(L-lactide)-*b*-poly(L-lysine) block copolymers.

We prepared and characterized F-PLA and FF-PLA using an alternative process [87] to the one discussed above [88]. Accordingly, the polymer was grown from the peptide segment, which was used as initiator of the polymerization reaction (Fig. 6a). This conjugate exhibited relatively high molecular weights (i.e., 49,000 and 66,000 g/mol and polydispersity of 1.41 and 1.48



Scheme 1



Scheme 2

Fig. 6 (**a**) Synthesis of F–PLA and FF–PLA conjugates initiated by L-phenylalanine (H–Phe–OH) and L,L-diphenylalanine (H–Phe–Phe–OH). Taken with permission from ref. 87 (RSC Advances, 2014. 4(44): pp. 23,231–23,241); (**b**) Chemical structure of PEDOT-RG^{E}D

for F-PLA and FF-PLA, respectively) and a yield of ~70 % for the PLA component. A suitable choice of the reaction time and temperature avoided thermal degradation of the biomaterial. The degree of crystallinity was around 30–33 % for the two hybrids, which is consistent with the relatively long segments arranged in a 10_7 helical conformation identified by FTIR spectroscopy. Circular dichroism (CD) spectroscopy was used to

examine the possible interactions between the peptide and polymer fragments in the conjugates, with the results indicating the absence of interaction between the two fragments for F-PLA and very weak for FF-PLA.

To gain microscopic information about the level organization of the fragments and the level of interaction among them, MD simulations were performed on a model conjugate formed by a 40 residues-long tail of PLA linked to the C-terminus of a diphenylalanine peptide. Simulations, which were performed in 1,1,1,3,3,3-hexafluuoroisopropanol to facilitate the comparison with available experimental data, evidenced that the peptide fragment retains the intrinsic conformational preferences of diphenylalanine. This conclusion was in agreement with the relatively scarce interactions found between the FF and PLA blocks by CD spectroscopy. Indeed, the existent interactions were restricted to hydrogen bonds between the nonterminal phenylalanine residue and the L-lactide unit immediately after it. Thus, PLA tends to organize independently, which is essential for the construction of peptide guided assemblies.

Similar conclusions were reached in a previous study devoted to the hybrid amphiphile formed by the conjugation of a hydrophobic peptide with four phenylalanine (Phe) residues and hydrophilic poly(ethylene glycol) (PEG), hereafter denoted FFFF-PEG. This polymer is widely used in biomedicine because its properties as steric stabilizer, which help to encapsulate insoluble small molecules such as drugs, prevent or hinder their uptake, and facilitate their slow release. Experimental results reported by Castelletto and Hamley [89] revealed that FFFF-PEG tends to aggregate via hydrophobic interactions, even at moderately low concentrations, with a characteristic critical aggregation concentration. Above it, β-sheet organizations are detectable even before straight fibril structures start growing and depositing. These aggregates are much shorter than those observed for amyloid peptides though. Finally, PEG crystallization does not disrupt local β-sheet structure, even though on longer length scales the β-sheet fibrillar structure might be perturbed by the formation of spherulites from PEG crystallization. Theoretical studies using a combination of quantum mechanical calculations and atomistic molecular dynamics simulations allowed us to conclude that the two counterparts of FFFF-PEG amphiphile tend to organize as independent modules [90], as was also proved for FF-PLA.

Recently our lab has developed a new strategy for the preparation of peptide-polymer conjugates. This approach is based on the concept of chemical similarity of the two components of the conjugate. In order to achieve this similarity, exotic amino acids bearing the chemical characteristics of the polymer are designed and, subsequently, synthesized (e.g., Fig. 6). For example, synthetic amino acids bearing a 3,4-ethylenedioxythiophene (EDOT) were

prepared to produce conjugates with poly(3,4-ethylenedioxithiophene) [91, 92], abbreviated PEDOT. The latter is among the most successful electroactive conducting polymers due to its excellent electrochemical and thermal properties, high conductivity, good environmental stability in its doped state, mechanical flexibility, relative ease of preparation, and fast doping-undoping process [93, 94]. We showed that the conjugates obtained by linking such synthetic amino acids with PEDOT (named PEDOT-I and PEDOT-II in Scheme 1) exhibit electrochemical and electrical activity. Furthermore, cell adhesion and proliferation assays showed that the behavior of both PEDOT-I and PEDOT-II as cellular matrices is better than is PEDOT counterpart, the latter being a well-known electro-biocompatible material [91, 95].

Inspired by such results, we have recently used the strategy based on chemical similarity to design an electroactive Arg-Gly-Asp (RGD)-based peptide-PEDOT conjugate. For this purpose, the Gly residue of the RGD sequence has been replaced by amino acids bearing a 3,4-ethylenedioxythiophene as a side group [96]. The resulting sequence, hereafter denoted $RG^{E}D$ has been attached to the end of PEDOT chains forming the PEDOT-$RG^{E}D$ conjugate (Fig. 6b). This conjugate, which has been found to combine the cell adhesive activity of the RGD sequence with the electrochemical activity of PEDOT, behaves as an excellent soft bioelectroactive support for tissue regeneration through electrostimulation.

From a theoretical point of view, studies on PEDOT-I, PEDOT-II and PEDOT-$RG^{E}D$ pointed that they differ from FF-PLA and FFFF-PEG. PEDOT is a relatively rigid polymer and the most relevant properties of the electroactive conjugates refer to electron delocalization and electronic transitions. The conformational flexibility of the amino acids and the $RG^{E}D$ peptide were examined using quantum mechanical methods [92, 96]. The most stable conformers were coupled to a small PEDOT chain and the electronic properties in different environments were predicted using methods such as time-dependent density functional theory, and TD-DFT calculations to rationalize experimental observations.

3.1 Amyloid Peptides

Amyloid peptides, regardless of their sizes, functions, and sequences, have great potential as building blocks in the creation of dysfunctional/functional nanostructures, because they have natural ability to self-assemble into nanofibrillar structures and can be easily modified with various functional groups. Under the disease conditions, amyloid peptides can misfold and self-assemble into different dysfunctional nanostructures at the intermediate and final aggregation stages including linear, micelles, and annular organizations [97–99]. These dysfunctional amyloid nanostructures are known to be associated with more than 20 neurodegenerative

diseases, including Alzheimer's, Parkinson's, Huntington's, type II diabetes, and prion diseases [100–103]. Dysfunctional amyloid nanostructures adopt different structural morphologies, but they all contain certain degrees of cross-β-sheet structures, suggesting that amyloid oligomerization/fibrillization proceeds through different assembly pathways [104]. On the other hand, amyloid peptides can also form functional nanostructures, which help to regulate biological functions in synapse formation [105], hormone reservoir manufacture [84], and antimicrobial properties [106, 107]. Amyloid fibrils as final aggregation products of different amyloid peptides are robust, with mechanical strength similar to spider silk [108] and structural stability similar to barnacle cement [109]. Amyloid fibrils are also highly resistant to degradation and damage by proteases [110], UV light exposure [111], and high temperature of water [112, 113]. Thus amyloid fibers have been functionalized for applications in metal nanowires [114–116], tissue engineering [117, 118], and drug and gene delivery [119, 120].

Both dysfunctional and functional amyloid nanostructures are biologically important for different applications. Thus, obtaining atomic-level structures of amyloid aggregates is an important step towards not only understanding amyloid functions and its underlying aggregation principles, but also structural-based design of functional amyloids. Different experimental techniques are used to probe structural information and biological function of amyloidosis. Solid-state NMR and X-ray diffraction are good approaches for resolving atomic-level structural information [78, 121, 122], but the nature of protein aggregation (noncrystallization and insolubility of fibrils, small sizes, short-lived states, involvement of cell membrane) renders these experimental studies extremely challenging [123, 124]. AFM and EM techniques can provide morphological images at nanoscale [125–127], but detailed structural and kinetic information are not reliable, even though EM is now approaching atomic scale resolution. The difficulties and limitations of these experimental methods in structural determination have inspired intensive computational studies to complement experiments. Most computer simulations of amyloid-forming peptides fall into two levels, atomic and coarse-grained with explicit and implicit solvent models [128]. All-atom molecular dynamics (MD) simulations have been applied to study relatively small amyloid oligomers by testing different candidate β-sheet arrangements of preformed oligomers mimicking possible nucleus seeds at the very early stage of amyloid formation [72, 129–131]. This approach can determine the most stable conformation for minimal nucleus seeds at the lowest free energy state, but cannot provide the aggregation scenario of amyloid intermediates/fibril growth since aggregation is an extremely slow process on the timescale of minutes to days, which is typically beyond the timescale of nanoseconds for conventional MD simulations. To overcome computational

limitations, alternative computer simulations using low-resolution models (e.g., coarse-grained protein models and implicit solvent models) have been used to directly study the formation of oligomers (small species) and even fibrils (large species) [123, 132]. These simulations can qualitatively provide information on the kinetic pathways of protein aggregation, but cannot adequately capture different detailed interactions, such as hydrophobic interactions, electrostatic interactions, and hydrogen bonding. Once the amyloid structures are determined, structure-based design of functional amyloids becomes achievable. Experimental and theoretical methods have strengths and weaknesses, but a combination of experimental, theoretical, and computational methods can capture amyloid nanostructures at different length and time scales.

3.1.1 General Protocol

From a computational point of view, amyloid oligomers should be (meta)stable in solution so that they do have enough time to interact with the cell membrane and impair the cells by either forming specific-ion-leakage channel or thinning/damaging cell membrane. If the oligomers are unstable, they quickly disassociate into monomers or aggregate into mature fibrils, which have been shown to have less damage to cells. Thus, identification of stable oligomers that are able to retain their initial structural organization at the lowest free energy state in simulations is the first step to correlate amyloid structures with their biological functions. In this section, we present a general computational protocol of our peptide-packing program, which is used to predict atomic structures of amyloid fibrils/oligomers. Figure 7 depicts the overview of

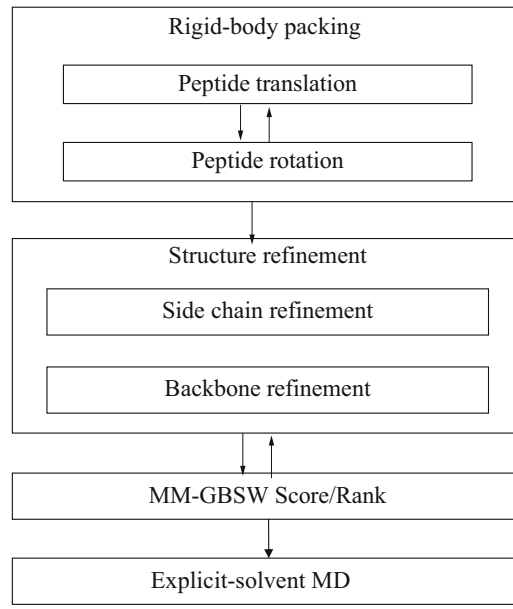


Fig. 7 An overview of the peptide packing procedure

the protocol workflow of the peptide packing program, consisting of the following steps:

1. *Rigid-body packing*. The rigid-body packing module is used for "coarse" structural prediction: (a) The building block can be either amyloid monomer or oligomer. (b) The assembly symmetry should be given. (c) The key peptide-peptide reaction coordinates should be predefined. For a twofold symmetry, the displacement and orientation of one β-sheet with respect to the other are key coordinates; for a threefold symmetry, the rotation of β-sheet with respect to the others is a key coordinate, and for a spherical symmetry, peptide self-rotation, peptide-to-peptide displacement and orientation, and layer-to-layer orientation are key coordinates. (d) The distance between β-sheets should be set to 10 Å, which corresponds to the average distance in a cross-β structure; the distance between β-strands should be 4.7 Å, which allows chains to form hydrogen bonds. (e) A local energy minimization is used to remove any steric clash. (f) Hydrophobic contacts, hydrogen bonding, shape-complementary parameters are calculated and used as criteria to tune rigid-body movement for optimizing backbone-backbone and side chain–side chain interactions.
2. *Structural refinement*. Peptide flexibility presents a major challenge in molecular docking and assembly [133], because peptide flexibility, including backbone and side chain movements significantly extends the search space for optimal structure of the assembly. In this module, a Monte Carlo Minimization (MCM) method will be used to handle backbone and side chain flexibility. Each MCM cycle consists of (a) rigid body perturbation (i.e., peptide translation and peptide rotation), (b) backbone and side chain optimization (i.e., torsion angle rotation), and (c) steepest descent minimization.
3. *Molecular mechanic generalized-born surface area (MM-GBSA)*. The MM-GBSW method has been implemented in the peptide packing program and used to score and rank all peptide assemblies in terms of free energy. The MM-GBSA approach, combined the molecular mechanics with the implicit solvent generalized-born method and CHARMM force field [68, 134] can accurately reproduce the folding and assembly of membrane proteins in aqueous solution and in heterogeneous biological membranes [135], but is much less computationally demanding due to the largely reduced number of degrees of freedom. The free energy of the system (G) is computed by $G = G_{\text{polar}} + G_{\text{nonpolar}} + E_{\text{mm}} - \text{TS}$, where a polar solvation energy (G_{polar}) is computed by the GB model; a nonpolar solvation energy (G_{nonpolar}) is computed from a solvent-accessible surface area model; a molecular mechanics energy (E_{mm}) is a sum of bonds,

angles, torsions, van der Waals, and electrostatic interactions; and the entropy effect by solute vibration is estimated by the normal mode calculation.

4. *Explicit-solvent molecular dynamics (MD) simulations.* Once amyloid nanostructured aggregates are obtained from **steps 1–3**, they are subject to explicit-solvent MD simulations to validate their structural stability. In general, amyloid aggregates are solvated in a TIP3P water cubic box with a margin of at least 15 Å from any edge of the water box to any peptide atom. Water molecules within 2.4 Å of the amyloid aggregates are removed to avoid initial overlapping. The systems are then neutralized by adding counter ions of Cl^- and Na^+ to reach ionic strength of interest (i.e., 100 ~ 150 mM). The resulting systems are minimized in energy for 5000 steps with peptides restrained, followed by additional 5000 steps of minimization for the whole system to remove unfavorable contacts between solvent and peptides. Next, the systems are subject to 1 ns MD run with harmonica constrained on the backbone atoms of the peptides. The production runs are carried out in the NPT ensemble (i.e., 1 atm and 300 K). Constant pressure and temperature in the system are maintained by an isotropic Langevin barostat and a Langevin thermostat, respectively. Long-range electrostatics interactions are treated by the particle mesh Ewald sum method, while short-range van der Waals (VDW) interactions are typically evaluated by a switching method with a twin range cutoff of 10 and 12 Å. The integration time step is 2 fs with the RATTLE algorithm applied to constrain bonds involving hydrogen atoms. Periodic boundary condition with the minimum image convention is applied to all directions. All models are run twice to validate simulation convergence by using the same starting coordinates but different initial velocities assigned by the Maxwell-Boltzman distribution. In our studies, all MD simulations are performed by the NAMD program [67] with all-atom CHARMM27 force field [134].

3.1.2 Representative Example of Aβ Micelles

We have used the peptide packing program, combined with structural information available from experiments, to determine a series of atomic structures of amyloid-β (Aβ) linears [136, 137], micelles [138], triangulars [139], snowflakes [140], annulars [141], and globulomers [142] (Fig. 8), hIAPP stacking-sandwich oligomers [143] and wrapping-cord triangular oligomers [144], and Tau octamers with three- and four-repeat segments [145–147]. These oligomers vary considerably in β-sheet packing and orientation, but all display high structural stability, reflecting a highly polymorphic nature of amyloids in a rugged energy landscape along different aggregation pathways. In addition, structural analysis also reveals that different β-sheet associations provide different driving forces to

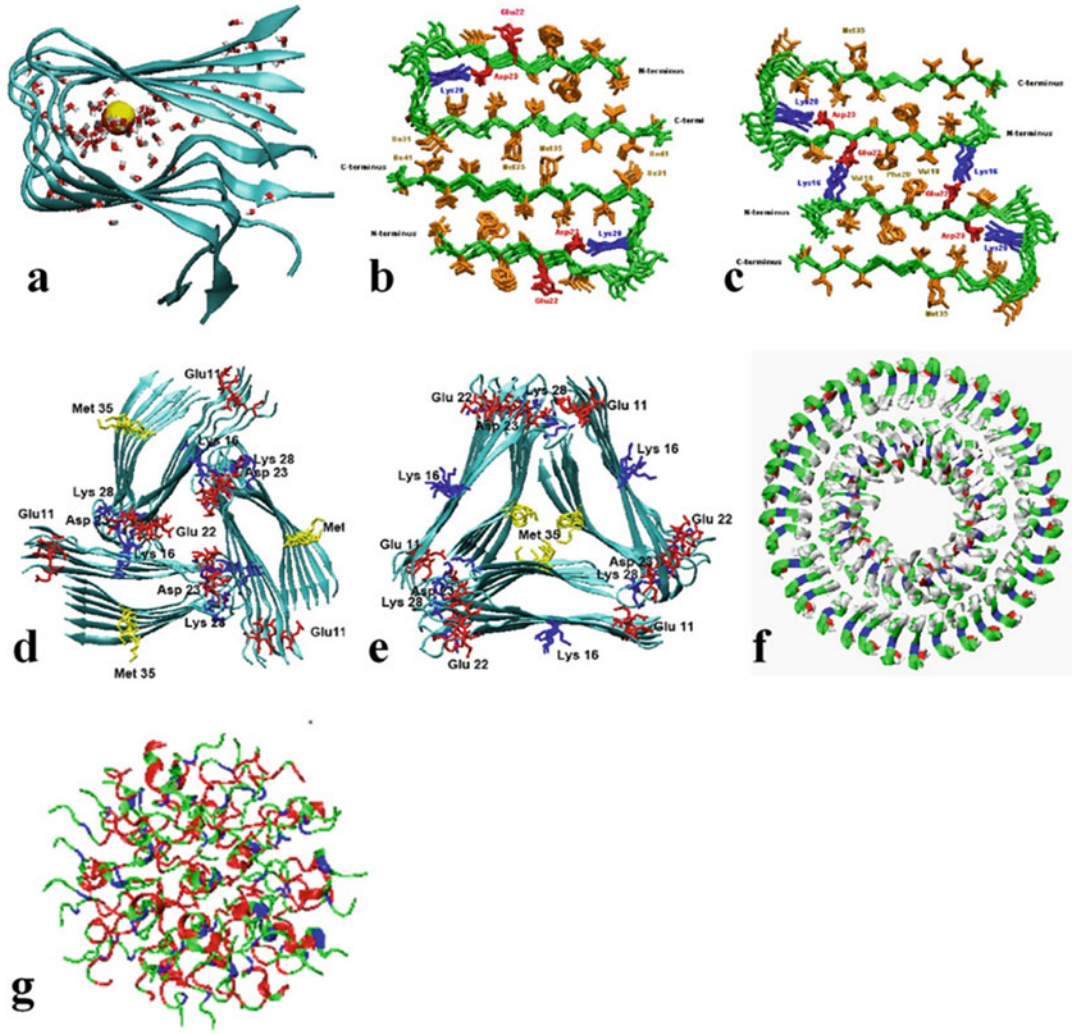


Fig. 8 Atomic structures of amyloid oligomers formed by Aβ peptides. Each structure is computationally optimized and determined from thousands of conformers at the lowest energy state. Aβ oligomers include (**a**) single-layer linears; (**b**, **c**) double-layered linear, dimeric pentamers stacked in an antiparallel fashion via either C-terminal-C-terminal (CC) or N-terminal-N-terminal (NN) interface; (**d**, **e**) threefold triangular 18-mers with loop-next-to-tail or loop-next-to-strand organization; (**f**) double-layered annular 60-mer with the CC interface; (**g**) micelle with antiparallel peptide orientation

stabilize intra-sheet organization via Asn/Gln ladders, aromatic stacking, and continuous hydrogen bonding.

Here we presented a protocol to construct more complex Aβ micelles as an example. Figure 9 shows a three-step procedure to build a micelle. First, single $A\beta_{25-35}$ peptide was aligned to the *z* axis with a minimal distance of ~4 Å from the origin of the Cartesian coordinate. Second, the peptide was replicated and rotated along the *y* axis at every 30° to form a semi-circle by seven A, B, C, D, E,

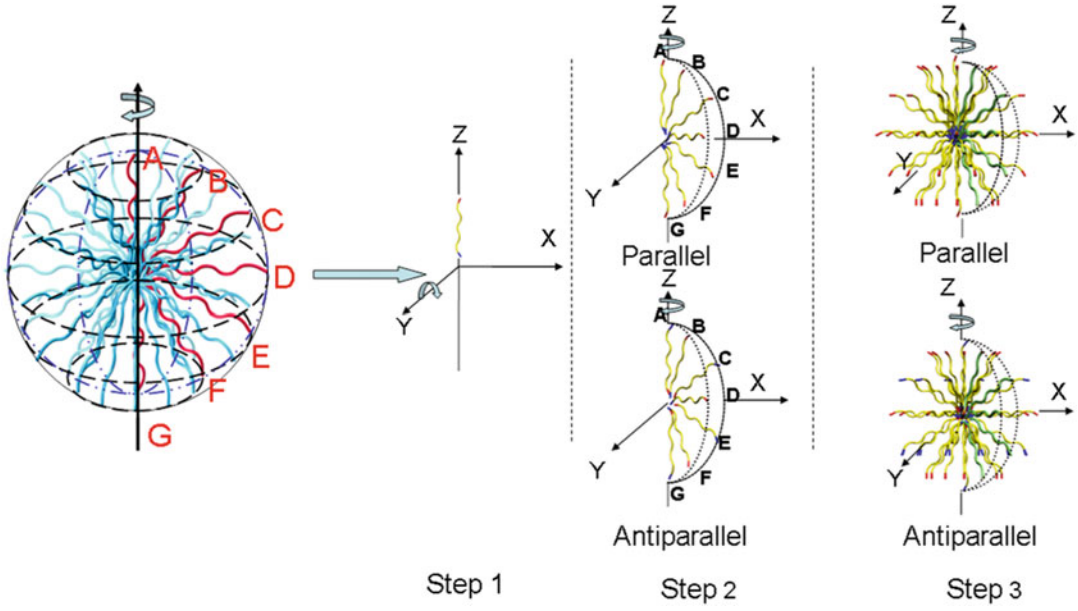


Fig. 9 A three-step strategy to construct $A\beta_{25–35}$ micelles with parallel and antiparallel peptide orientations

F, G peptides with the same parallel orientation in the *xz* plane. Then, peptides of B, D, and F are rotated additional 15° along the *z* axis so that peptides of A, C, E, and G and peptides of B, D, and F are located in different planes. For the antiparallel packing, peptides of A, C, E, and G were reversed to impose opposite orientation relative to peptides of B, D, and F. Finally, five peptides of B, C, D, E, and F (exclusion of A and G) were rotated and copied along the *z* axis at every 30° to form a micelle consisting of different circle layers, namely B, C, D, E, and F layers. Each layer consists of 12 peptides except A and G layers, leading to total 62 peptides in the micelle. Four micelle structures were subject to "coarse" structure optimization by using energy minimization with generalized born of a simple switching function (GBSW) implicit solvent model [148]. For each "coarse-optimized" micelle, we further refined the structure by adjusting peptide self-rotating angle (Φ) along the helical axis and peptide displacement between different layers (λ), i.e., each peptide was rotated along its helical axis at every 15° to avoid side-chain clash, while peptides of A, C, E, and G were moved with respect to peptides of B, D, F along the opposite direction. The structure-refining procedure generated 504 distinct structures for each "coarse-optimized" micelle. A total of 2016 micelles were energy-minimized by using 300 steps of steepest decent with backbone constrained, followed by 200 steps conjugate and 300 steps steep decent minimization without position constrains in the presence of the GBSW implicit solvent. Four different lowest-energy micelles, one from each category (i.e., parallel or antiparallel

orientations with N- or C-terminal exposed to solvent)), were selected and subject to explicit-solvent MD simulation for examining their structural and energetic aspects at the early stage of aggregation process. Collective MD simulations identified the Aβ micelles with antiparallel orientations as not only with high structural stability, but also high binding affinity to an antibody, suggesting that these Aβ micelles may present more biologically relevant species.

Acknowledgements

J.Z. thanks for financial supports from the National Science Foundation (CAREER Award 0952624, 1510099, and 1607475) and Alzheimer Association—New Investigator Research Grant (2015-NIRG-341372), and National Natural Science Foundation of China (NSFC-21528601). The calculations were carried out in part on the UMass Boston research cluster. This work has been funded in whole or in part with Federal funds from the National Cancer Institute, National Institutes of Health, under contract number HHSN261200800001E. The content of this publication does not necessarily reflect the views or policies of the Department of Health and Human Services, nor does mention of trade names, commercial products, or organizations imply endorsement by the U.S. Government. This research was supported (in part) by the Intramural Research Program of the NIH, National Cancer Institute, Center for Cancer Research.

References

1. Ferrari M (2005) Cancer nanotechnology: opportunities and challenges. Nat Rev Cancer 5(3):161–171
2. Ferrari M (2005) Nanovector therapeutics. Curr Opin Chem Biol 9(4):343–346
3. Blanco E et al (2011) Nanomedicine in cancer therapy: innovative trends and prospects. Cancer Sci 102(7):1247–1252
4. Gradišar H, Jerala R (2014) Self-assembled bionanostructures: proteins following the lead of DNA nanostructures. J Nanobiotechnol 12:4
5. Rubin DJ et al (2015) Structural, nanomechanical, and computational characterization of d,l-cyclic peptide assemblies. ACS Nano 9 (3):3360–3368
6. Balzani V, Credi A, Venturi M (2009) Light powered molecular machines. Chem Soc Rev 38(6):1542–1550
7. King NP et al (2012) Computational design of self-assembling protein nanomaterials with atomic level accuracy. Science 336 (6085):1171–1174
8. Zhang Z-J et al (2013) A double-leg donor–acceptor molecular elevator: new insight into controlling the distance of two platforms. Org Lett 15(7):1698–1701
9. Pei H et al (2014) Functional DNA nanostructures for theranostic applications. Acc Chem Res 47(2):550–559
10. Rangnekar A, LaBean TH (2014) Building DNA nanostructures for molecular computation, templated assembly, and biological applications. Acc Chem Res 47(6):1778–1788
11. Bindewald E et al (2008) RNAJunction: a database of RNA junctions and kissing loops for three-dimensional structural analysis and nanodesign. Nucleic Acids Res 36(database issue):D392–D397
12. Mathur D, Henderson ER (2013) Complex DNA nanostructures from oligonucleotide ensembles. ACS Synth Biol 2(4):180–185

13. Li H, LaBean TH, Leong KW (2011) Nucleic acid-based nanoengineering: novel structures for biomedical applications. Interface Focus 1 (5):702–724
14. Grabow WW et al (2011) Self-assembling RNA nanorings based on RNAI/II inverse kissing complexes. Nano Lett 11(2):878–887
15. Miller BG, Raines RT (2005) Reconstitution of a defunct glycolytic pathway via recruitment of ambiguous sugar kinases†. Biochemistry 44(32):10776–10783
16. Bang D, Kent SBH (2005) His(6) tag-assisted chemical protein synthesis. Proc Natl Acad Sci U S A 102(14):5014–5019
17. Miller Y, Ma B, Nussinov R (2015) Polymorphism in self-assembly of peptide-based β-hairpin contributes to network morphology and hydrogel mechanical rigidity. J Phys Chem B 119(2):482–490
18. Tsai H-H et al (2004) In silico protein design by combinatorial assembly of protein building blocks. Protein Sci 13(10):2753–2765
19. Main ERG et al (2005) A recurring theme in protein engineering: the design, stability and folding of repeat proteins. Curr Opin Struct Biol 15(4):464–471
20. Jenkins J, Mayans O, Pickersgill R (1998) Structure and evolution of parallel [beta]-helix proteins. J Struct Biol 122 (1–2):236–246
21. Haspel N et al (2006) De novo tubular nanostructure design based on self-assembly of [beta]-helical protein motifs. Structure 14 (7):1137–1148
22. Bernstein FC, The protein data bank et al (1977) Eur J Biochem 80(2):319–324
23. Haspel N et al (2007) Changing the charge distribution of {beta}-helical-based nanostructures can provide the conditions for charge transfer. Biophys J 93(1):245–253
24. Schmitz K (2010) Amino acids, peptides and proteins in organic chemistry. Volume 1–Origins and synthesis of amino acids. Edited by Andrew B. Hughes. Angew Chem Int Ed 49 (22):3717–3718
25. Vadim AS, Kunisuke I (2009) Asymmetric synthesis and application of α-amino acids. In: ACS symposium series, vol 1009. American Chemical Society, p. 512.
26. Christophorou MA et al (2014) Citrullination regulates pluripotency and histone H1 binding to chromatin. Nature 507 (7490):104–108
27. Cativiela C, Ordonez M (2009) Recent progress on the stereoselective synthesis of cyclic quaternary alpha-amino acids. Tetrahedron Asymmetry 20(1):1–63
28. Michaux J, Niel G, Campagne JM (2009) Stereocontrolled routes to beta, beta′-disubstituted alpha-amino acids. Chem Soc Rev 38 (7):2093–2116
29. Degenkolb T, Bruckner H (2008) Peptaibiomics: towards a myriad of bioactive peptides containing C(alpha)-dialkylamino acids? Chem Biodivers 5(9):1817–1843
30. Schwarzer D, Finking R, Marahiel MA (2003) Nonribosomal peptides: from genes to products. Nat Prod Rep 20(3):275–287
31. Nestor JJ Jr (2009) The medicinal chemistry of peptides. Curr Med Chem 16 (33):4399–4418
32. Horne WS, Gellman SH (2008) Foldamers with heterogeneous backbones. Acc Chem Res 41(10):1399–1408
33. Ersmark K, Del Valle JR, Hanessian S (2008) Chemistry and biology of the aeruginosin family of serine protease inhibitors. Angew Chem Int Ed Engl 47(7):1202–1223
34. Voloshchuk N, Montclare JK (2010) Incorporation of unnatural amino acids for synthetic biology. Mol Biosyst 6(1):65–80
35. Wu X, Schultz PG (2009) Synthesis at the interface of chemistry and biology. J Am Chem Soc 131(35):12497–12515
36. Murakami H et al (2000) Site-directed incorporation of fluorescent nonnatural amino acids into streptavidin for highly sensitive detection of biotin. Biomacromolecules 1 (1):118–125
37. Vazquez ME et al (2003) Fluorescent caged phosphoserine peptides as probes to investigate phosphorylation-dependent protein associations. J Am Chem Soc 125 (34):10150–10151
38. Alfonta L et al (2003) Site-specific incorporation of a redox-active amino acid into proteins. J Am Chem Soc 125(48):14662–14663
39. Lemke EA et al (2007) Control of protein phosphorylation with a genetically encoded photocaged amino acid. Nat Chem Biol 3 (12):769–772
40. Wang W et al (2007) Genetically encoding unnatural amino acids for cellular and neuronal studies. Nat Neurosci 10(8):1063–1072
41. Cellitti SE et al (2008) In vivo incorporation of unnatural amino acids to probe structure, dynamics, and ligand binding in a large protein by nuclear magnetic resonance spectroscopy. J Am Chem Soc 130(29):9268–9281
42. Summerer D et al (2006) A genetically encoded fluorescent amino acid. Proc Natl Acad Sci U S A 103(26):9785–9789
43. Lee HS et al (2009) Genetic incorporation of a small, environmentally sensitive, fluorescent

probe into proteins in Saccharomyces cerevisiae. J Am Chem Soc 131(36):12921–12923
44. Zanuy D et al (2007) Use of constrained synthetic amino acids in beta-helix proteins for conformational control. J Phys Chem B 111 (12):3236–3242
45. Zanuy D et al (2009) Protein segments with conformationally restricted amino acids can control supramolecular organization at the nanoscale. J Chem Inf Model 49 (7):1623–1629
46. Tu RS, Tirrell M (2004) Bottom-up design of biomimetic assemblies. Adv Drug Deliv Rev 56(11):1537–1563
47. Lee MR et al (2009) Nylon-3 copolymers that generate cell-adhesive surfaces identified by library screening. J Am Chem Soc 131 (46):16779–16789
48. Revilla-Lopez G et al (2010) NCAD, a database integrating the intrinsic conformational preferences of non-coded amino acids. J Phys Chem B 114(21):7413–7422
49. Aleman C (1997) Conformational properties of [alpha]-amino acids disubstituted at the [alpha]-carbon. J Phys Chem B 101 (25):5046–5050
50. Alemán C, Casanovas J, Galembeck SE (1998) PAPQMD parametrization of molecular systems with cyclopropyl rings: conformational study of homopeptides constituted by l-aminocyclopropane-l-carboxylic acid. J Computer-Aided Molecular Design 12 (3):259–273
51. Gomez-Catalan J, Aleman C, Perez JJ (2000) Conformational profile of 1-aminocyclopropanecarboxylic acid. Theor Chem Acc 103(5):380–389
52. Benedetti E et al (1989) Structural versatility of peptides containing C-alpha, alpha-dialkylated glycines. An X-ray diffraction study of 6 1-aminocyclopropane-1-carboxylic acid rich peptides. Int J Biol Macromol 11 (6):353–360
53. Benedetti E et al (1989) Structural versatility of peptides from C[alpha, alpha] dialkylated glycines: linear Ac3c homo-oligopeptides. Biopolymers 28(1):175–184
54. Fabiano N et al (1993) Conformational versatility of the N-alpha-acylated tripeptide amide tail of oxytocin - synthesis and crystallographic characterization of 3C-2-alpha-backbone modified, conformationally restricted analogs. Int J Pept Protein Res 42 (5):459–465
55. Valle G et al (1989) Linear oligopeptides. 200. crystallographic characterization of conformation of 1-aminocyclopropane-1-carboxylic acid residue (Ac3c) in simple derivatives and peptides. Int J Pept Protein Res 34 (1):56–65
56. Aleman C et al (2002) Influence of the phenyl side chain on the conformation of cyclopropane analogues of phenylalanine. J Phys Chem B 106(45):11849–11858
57. Toniolo C et al (2001) Control of peptide conformation by the Thorpe-Ingold effect (C-alpha-tetrasubstitution). Biopolymers 60 (6):396–419
58. Hruby VJ et al (1997) Design of peptides, proteins, and peptidomimetics in chi space. Biopolymers 43(3):219–266
59. Jimenez AI, Cativiela C, Marraud M (2000) A gamma-turn induced by a highly constrained cyclopropane analogue of phenylalanine (c(3) diPhe) in the solid state. Tetrahedron Lett 41 (28):5353–5356
60. Jimenez AI et al (1998) Beta-turn preferences induced by 2,3-methanophenylalanine chirality. J Am Chem Soc 120(37):9452–9459
61. Jimenez AI et al (1997) Folding types of dipeptides containing the diastereoisomeric cyclopropanic analogues of phenylalanine. Tetrahedron Lett 38(43):7559–7562
62. Casanovas J et al (2003) N-Acetyl-N′-methylamide derivative of (2S,3S)-1-amino-2,3-diphenylcyclopropanecarboxylic acid: theoretical analysis of the conformational impact produced by the incorporation of the second phenyl group to the cyclopropane analogue of phenylalanine. J Org Chem 68 (18):7088–7091
63. Jiménez AI, Ballano G, Cativiela C (2005) First observation of two consecutive gamma turns in a crystalline linear dipeptide. Angew Chem Int Ed 44(3):396–399
64. Casanovas J et al (2006) Conformational analysis of a cyclopropane analogue of phenylalanine with two geminal phenyl substituents. J Phys Chem B 110(11):5762–5766
65. Crisma M et al (2006) Preferred 3D-structure of peptides rich in a severely conformationally restricted cyclopropane analogue of phenylalanine. Chemistry 12(1):251–260
66. Royo S et al (2005) Turn and helical peptide handedness governed exclusively by side-chain chiral centers. J Am Chem Soc 127 (7):2036–2037
67. Kale L et al (1999) NAMD2: greater scalability for parallel molecular dynamics. J Comput Phys 151(1):283–312
68. MacKerell AD et al (1998) All-atom empirical potential for molecular modeling and dynamics studies of proteins. J Phys Chem B 102 (18):3586–3616

69. William LJ et al (1983) Comparison of simple potential functions for simulating liquid water. J Chem Phys 79(2):926–935
70. Darden T et al (1999) New tricks for modelers from the crystallography toolkit: the particle mesh Ewald algorithm and its use in nucleic acid simulations. Structure 7(3): R55–R60
71. Haspel N et al (2004) A comparative study of amyloid fibril formation by residues 15–19 of the human calcitonin hormone: a single beta-sheet model with a small hydrophobic core. J Mol Biol 345(5):1213–1227
72. Ma B, Nussinov R (2002) Stabilities and conformations of Alzheimer's beta-amyloid peptide oligomers (Abeta 16–22, Abeta 16–35, and Abeta 10–35): sequence effects. Proc Natl Acad Sci U S A 99(22):14126–14131
73. Ma B, Nussinov R (2002) Molecular dynamics simulations of alanine rich {beta}-sheet oligomers: insight into amyloid formation. Protein Sci 11(10):2335–2350
74. Ma B, Nussinov R (2003) Molecular dynamics simulations of the unfolding of {beta}2-microglobulin and its variants. Protein Eng 16(8):561–575
75. Zanuy D, Ma B, Nussinov R (2003) Short peptide amyloid organization: stabilities and conformations of the islet amyloid peptide NFGAIL. Biophys J 84(3):1884–1894
76. Zanuy D, Nussinov R (2003) The sequence dependence of fiber organization. A comparative molecular dynamics study of the islet amyloid polypeptide segments 22–27 and 22–29. J Mol Biol 329(3):565–584
77. Luhrs T et al (2005) 3D structure of Alzheimer's amyloid-{beta}(1–42) fibrils. Proc Natl Acad Sci U S A 102(48):17342–17347
78. Petkova AT et al (2002) A structural model for Alzheimer's beta-amyloid fibrils based on experimental constraints from solid state NMR. Proc Natl Acad Sci U S A 99 (26):16742–16747
79. Reches M, Porat Y, Gazit E (2002) Amyloid fibril formation by pentapeptide and tetrapeptide fragments of human calcitonin. J Biol Chem 277(38):35475–35480
80. Zanuy D et al (2004) Peptide sequence and amyloid formation: molecular simulations and experimental study of a human islet amyloid polypeptide fragment and its analogs. Structure 12(3):439–455
81. Thompson JD, Gibson TJ, Higgins DG (2002) Multiple sequence alignment using ClustalW and ClustalX. Curr Protoc Bioinformatics Chapter 2:Unit 2.3.
82. Zhang S (2003) Fabrication of novel biomaterials through molecular self-assembly. Nat Biotechnol 21(10):1171–1178
83. Liu TY et al (2013) Self-adjuvanting polymer-peptide conjugates as therapeutic vaccine candidates against cervical cancer. Biomacromolecules 14(8):2798–2806
84. Maji SK et al (2009) Functional amyloids as natural storage of peptide hormones in pituitary secretory granules. Science 325 (5938):328–332
85. Vandermeulen GWM, Klok H-A (2004) Peptide/protein hybrid materials: enhanced control of structure and improved performance through conjugation of biological and synthetic polymers. Macromol Biosci 4 (4):383–398
86. Haspel N et al (2012) Conformational exploration of two peptides and their hybrid polymer conjugates: potentialities as self-aggregating materials. J Phys Chem B 116 (48):13941–13952
87. Murase SK et al (2014) Molecular characterization of l-phenylalanine terminated poly(l-lactide) conjugates. RSC Adv 4 (44):23231–23241
88. Fan Y et al (2005) l-Phe end-capped poly(l-lactide) as macroinitiator for the synthesis of poly(l-lactide)-b-poly(l-lysine) block copolymer. Biomacromolecules 6 (6):3051–3056
89. Castelletto V, Hamley IW (2009) Self assembly of a model amphiphilic phenylalanine peptide/polyethylene glycol block copolymer in aqueous solution. Biophys Chem 141 (2–3):169–174
90. Zanuy D, Hamley IW, Aleman C (2011) Modeling the tetraphenylalanine-PEG hybrid amphiphile: from DFT calculations on the peptide to molecular dynamics simulations on the conjugate. J Phys Chem B 115 (28):8937–8946
91. Fabregat G et al (2013) An electroactive and biologically responsive hybrid conjugate based on chemical similarity. Polym Chem 4 (5):1412–1424
92. Fabregat G et al (2013) Design of hybrid conjugates based on chemical similarity. RSC Adv 3(43):21069–21083
93. Groenendaal L et al (2003) Electrochemistry of poly(3,4-alkylenedioxythiophene) derivatives. Adv Mater 15(11):855–879
94. Kirchmeyer S, Reuter K (2005) Scientific importance, properties and growing applications of poly(3,4-ethylenedioxythiophene). J Mater Chem 15(21):2077–2088

95. Maione S et al (2014) Electro-biocompatibility of conjugates designed by chemical similarity. J Pept Sci 20(7):537–546
96. Maione S, Gil A, Fabregat G, Del Valle LJ, Triguero J, Laurent A, Jacquemin D, Estrany F, Zanuy D, Cativiela C, Alemán C (2015) Electroactive polymer-peptide conjugates for adhesive biointerfaces. Biomater. Sci. (3):1395–1405.
97. Gosal WS et al (2005) Competing pathways determine fibril morphology in the self-assembly of [beta]2-microglobulin into amyloid. J Mol Biol 351(4):850–864
98. Lashuel HA et al (2002) Amyloid pores from pathogenic mutations. Nature 418 (6895):291
99. Lashuel HA et al (2002) Alpha-synuclein, especially the parkinson's disease-associated mutants, forms pore-like annular and tubular Protofibrils. J Mol Biol 322(5):1089–1102
100. Dauer W, Przedborski S (2003) Parkinson's disease: mechanisms and models. Neuron 39 (6):889–909
101. Dobson CM (2005) Structural biology: prying into prions. Nature 435 (7043):747–749
102. Meredith SC (2006) Protein denaturation and aggregation: cellular responses to denatured and aggregated proteins. Ann N Y Acad Sci 1066(1):181–221
103. Selkoe DJ (2001) Alzheimer's disease: genes, proteins, and therapy. Physiol Rev 81 (2):741–766
104. Tycko R (2004) Progress towards a molecular-level structural understanding of amyloid fibrils. Curr Opin Struct Biol 14 (1):96–103
105. Si K et al (2010) Aplysia CPEB can form prion-like multimers in sensory neurons that contribute to long-term facilitation. Cell 140 (3):421–435
106. Kagan BL, et al. (2011) Antimicrobial properties of amyloid peptides. Mol Pharm, Mol. Pharmaceutics, 2012, 9 (4):708–717
107. Zhang M, Zhao J, Zheng J (2014) Molecular understanding of a potential functional link between antimicrobial and amyloid peptides. Soft Matter 10(38):7425–7451
108. Slotta U et al (2007) Spider silk and amyloid fibrils: a structural comparison. Macromol Biosci 7(2):183–188
109. Sullan RMA et al (2009) Nanoscale structures and mechanics of barnacle cement. Biofouling 25(3):263–275
110. Ryu J, Park CB (2010) High stability of self-assembled peptide nanowires against thermal, chemical, and proteolytic attacks. Biotechnol Bioeng 105(2):221–230
111. Peralta MDR et al (2015) Engineering amyloid fibrils from β-solenoid proteins for biomaterials applications. ACS Nano 9 (1):449–463
112. Arora A, Ha C, Park CB (2004) Insulin amyloid fibrillation at above 100°C: new insights into protein folding under extreme temperatures. Protein Sci 13(9):2429–2436
113. Kardos J et al (2011) Reversible heat-induced dissociation of β2-microglobulin amyloid fibrils. Biochemistry 50(15):3211–3220
114. Reches M, Gazit E (2003) Casting metal nanowires within discrete self-assembled peptide nanotubes. Science 300(5619):625–627
115. Sakai H et al (2013) Formation of functionalized nanowires by control of self-assembly using multiple modified amyloid peptides. Adv Funct Mater 23(39):4881–4887
116. Scheibel T et al (2003) Conducting nanowires built by controlled self-assembly of amyloid fibers and selective metal deposition. Proc Natl Acad Sci U S A 100(8):4527–4532
117. Gras SL et al (2008) Functionalised amyloid fibrils for roles in cell adhesion. Biomaterials 29(11):1553–1562
118. Holmes TC et al (2000) Extensive neurite outgrowth and active synapse formation on self-assembling peptide scaffolds. Proc Natl Acad Sci U S A 97(12):6728–6733
119. Koutsopoulos S et al (2009) Controlled release of functional proteins through designer self-assembling peptide nanofiber hydrogel scaffold. Proc Natl Acad Sci U S A 106(12):4623–4628
120. Li D et al (2014) Structure-based design of functional amyloid materials. J Am Chem Soc 136(52):18044–18051
121. Chamberlain AK et al (2001) Characterization of the structure and dynamics of amyloidogenic variants of human lysozyme by NMR spectroscopy. Protein Sci 10(12):2525–2530
122. Tycko R (2000) Solid-state NMR as a probe of amyloid fibril structure. Curr Opin Chem Biol 4(5):500–506
123. Mousseau N, Derreumaux P (2005) Exploring the early steps of amyloid peptide aggregation by computers. Acc Chem Res 38 (11):885–891
124. Makabe K et al (2006) Atomic structures of peptide self-assembly mimics. Proc Natl Acad Sci U S A 103(47):17753–17758
125. Benseny-Cases N, Cocera M, Cladera J (2007) Conversion of non-fibrillar [beta]-sheet oligomers into amyloid fibrils in

Alzheimer's disease amyloid peptide aggregation. Biochem Biophys Res Commun 361 (4):916–921

126. Legleiter J et al (2004) Effect of different anti-A[beta] antibodies on a[beta] fibrillogenesis as assessed by atomic force microscopy. J Mol Biol 335(4):997–1006
127. Wang Z et al (2003) AFM and STM study of [beta]-amyloid aggregation on graphite. Ultramicroscopy 97(1–4):73–79
128. Morriss-Andrews A, Shea J-E (2014) Simulations of protein aggregation: insights from atomistic and coarse-grained models. J Phys Chem Lett 5(11):1899–1908
129. Tsai H-H et al (2005) Energy landscape of amyloidogenic peptide oligomerization by parallel-tempering molecular dynamics simulation: significant role of Asn ladder. Proc Natl Acad Sci U S A 102(23):8174–8179
130. Zheng J et al (2006) Structural stability and dynamics of an amyloid-forming peptide GNNQQNY from the yeast prion sup-35. Biophys J 91(3):824–833
131. Zheng J et al (2007) Nanostructure design using protein building blocks enhanced by conformationally constrained synthetic residues. Biochemistry 46(5):1205–1218
132. Nguyen HD, Hall CK (2006) Spontaneous fibril formation by polyalanines: discontinuous molecular dynamics simulations. J Am Chem Soc 128(6):1890–1901
133. Andrusier N et al (2008) Principles of flexible protein-protein docking. Proteins 73 (2):271–289
134. Brooks BR et al (1983) CHARMM: a program for macromolecular energy, minimization, and dynamics calculations. J Comput Chem 4(2):187–217
135. Im W, Feig M, Brooks CL III (2003) An implicit membrane generalized born theory for the study of structure, stability, and interactions of membrane proteins. Biophys J 85 (5):2900–2918
136. Wang Q et al (2011) Structural, morphological, and kinetic studies of beta-amyloid peptide aggregation on self-assembled monolayers. Phys Chem Chem Phys 13 (33):15200–15210
137. Yu X et al (2010) Atomic-scale simulations confirm that soluble [beta]-sheet-rich peptide self-assemblies provide amyloid mimics presenting similar conformational properties. Biophys J 98(1):27–36
138. Yu X, Wang Q, Zheng J (2010) Structural determination of A[beta]25-35 micelles by molecular dynamics simulations. Biophys J 99(2):666–674
139. Zheng J et al (2010) Molecular modeling of two distinct triangular oligomers in amyloid beta-protein. J Phys Chem B 114(1):463–470
140. Li L, Zheng J (2010) Computational modeling of amyloid oligomeric structures. Inter J Liquid State Sci 1(1):1–13
141. Zheng J et al (2008) Annular structures as intermediates in fibril formation of alzheimer A[beta]17-42. J Phys Chem B 112 (22):6856–6865
142. Yu X, Zheng J (2011) Polymorphic structures of Alzheimer's β-amyloid globulomers. PLoS One 6(6):e20575
143. Zhao J et al (2011) Structural polymorphism of human islet amyloid polypeptide (hIAPP) oligomers highlights the importance of interfacial residue interactions. Biomacromolecules 12(1):210–220
144. Zhao J et al (2011) Heterogeneous triangular structures of human islet amyloid polypeptide (amylin) with internal hydrophobic cavity and external wrapping morphology reveal the polymorphic nature of amyloid fibrils. Biomacromolecules 12(5):1781–1794
145. Luo Y et al (2013) Molecular insights into the reversible formation of tau protein fibrils. Chem Commun 49(34):3582–3584
146. Siddiqua A et al (2012) Conformational basis for asymmetric seeding barrier in filaments of three- and four-repeat tau. J Am Chem Soc 134(24):10271–10278
147. Yu X et al (2012) Cross-seeding and conformational selection between three- and four-repeat human Tau proteins. J Biol Chem 287 (18):14950–14959
148. Im W, Lee MS, Brooks CL III (2003) Generalized born model with a simple smoothing function. J Comput Chem 24 (14):1691–1702

Chapter 18

Probing Oligomerized Conformations of Defensin in the Membrane

Wenxun Gan, Dina Schneidman, Ning Zhang, Buyong Ma, and Ruth Nussinov*

Abstract

Computational prediction and design of membrane protein–protein interactions facilitate biomedical engineering and biotechnological applications. Due to their antimicrobial activity, human defensins play an important role in the innate immune system. Human defensins are attractive pharmaceutical targets due to their small size, broad activity spectrum, reduced immunogenicity, and resistance to proteolysis. Protein engineering based modification of defensins can improve their pharmaceutical properties. Here we present an approach to computationally probe defensins' oligomerization states in the membrane. First, we develop a novel docking and rescoring algorithm. Then, on the basis of the 3D structure of Sapecin, an insect defensin, and a model of its antimicrobial ion-channel, we optimize the parameters of our empirical scoring function. Finally, we apply our docking program and scoring function to the hBD-2 (human β-defensin-2) molecule and obtain structures of four possible oligomers. These results can be used in higher level simulations.

Key words Molecular docking, Empirical scoring function, Human defensin, Membrane protein, Peptide design, Protein–protein interaction

1 Introduction

Prediction and design of membrane protein–protein interactions have the potential to facilitate biomedical engineering for medical and biotechnological applications [1]. Computational study for weakly stable β-structures in membrane is important to engineer the biophysical properties including oligomerization state [2]. Defensins are crucial to innate immunity. They contribute to the antimicrobial action of granulocytes in the mucosa in the small intestine, in the epithelial host defense in the skin and elsewhere [3, 4]. They have antiviral activity against both enveloped and

* Corresponding author

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_18,

non-enveloped viruses [5], and they are important in HIV infection [6]. The oligomerization of defensins either forms of ion pores in bacterial membranes or aggregate into positively charged patches which disrupt the integrity of the lipid bilayer [7–9].

Humans express two types of defensins, α and β. Three human β-defensins: hβ-defensin-1, -2, and -3, have similar sequences, however, different properties [10]. It has been reported that several molecules can induce or enhance the production of defensins, for example, NOD2/CARD15 [11], TLR2 and TLR4 [12], and IL-12/IL-23/IL-27 [13]. Inducible hBD-2 could play a critical role in the protection of *M. pneumoniae* infection [14]. Human defensins also have complex roles in tumor growth, tumor monitoring, and cancer treatment [15]. hBD-2 exerts its growth suppression effect toward human melanoma cells via downregulation of B-Raf, cyclin D1, and cyclin E expression, upregulation of p21(WAF1) expression and activation of pRB [16]. hBD-2 may also control cell growth via arrest of G1/S transition and pRB activation [17]. Due to their well-established antimicrobial properties, defensins are also being investigated as therapeutics agents, especially as potential source to combat resistant bacteria. Human defensins are also attractive pharmaceutical targets due to their small size, broad activity spectrum, reduced immunogenicity and resistance to proteolysis [10, 18, 19].

Defensins perform their biological functions through three mechanisms: (1) Direct binding and modulation of host cell surface receptors and disruption of intracellular signaling which can inhibit viral replication [20]; (2) an indirect antiviral mechanism, where they function as chemokines to augment and alter adaptive immune responses; and (3) membrane disruption and pore formation [7–9]. The membrane-bound structure and topology of a human α-defensin indicate membrane pores consisting of dimers [21].

The characteristic folds of defensins are β-sheets stabilized with three disulfide bonds (Fig. 1). Their structural features, such as the helical N-terminal domains and oligomerization at the membrane surface, may modulate the efficiency of membrane insertion and selectivity for microbial or host-cell membranes. Both defensin-2 and -3 can interact with membranes as extended β-sheet platforms that present amphipathic helices for insertion into the lipid bilayer [22]. Nonetheless, many questions regarding the antiviral activities of defensins remain. Although significant mechanistic data are known for α-defensins, molecular details for β-defensins inhibition are mostly lacking [5]. The typical β-defensin action mechanism is not yet established, and one of the main challenges for the activation mechanism of the defense is the assembly in the membrane and the mechanism of membrane disruption.

Computational approaches have been employed to explore the dimerization of human β-defensin-2 [23], and to design sequences de novo based on flexible templates [24]. Here we present a

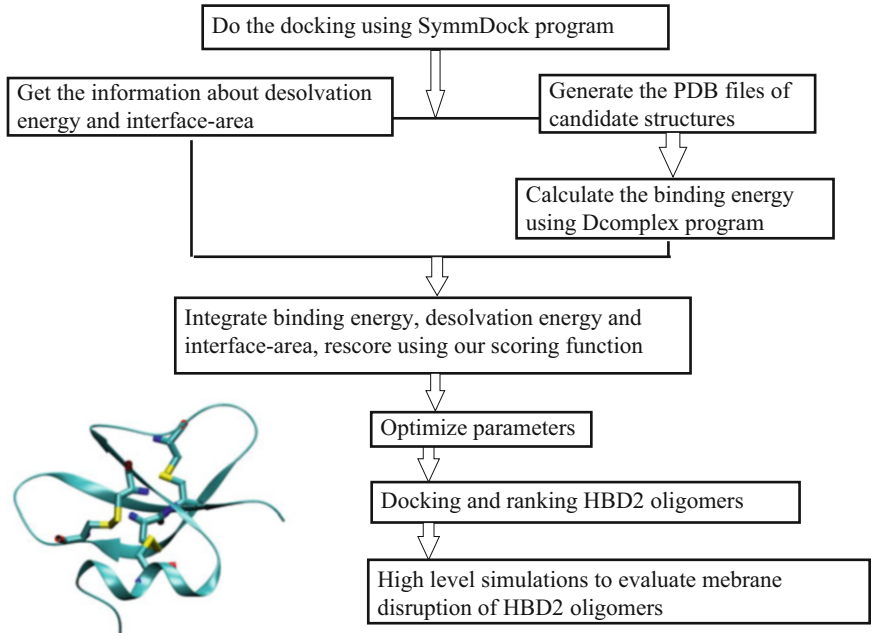


Fig. 1 Flowchart for the strategy to investigate the defensin oligomerization in membrane. The ribbon structure in *left corner* highlights three disulfide bonds in human defensin

computational protocol to probe possible oligomerization states of defensin in the membrane. We evaluate candidate states by a multiple protein–protein docking protocol. We focus on two β-defensins, one is the insect defensin Sapecin and another is human β-defensin-2 (hBD-2). The reason for choosing the two systems are (1) experimental information is available for possible protein–protein interactions and protein–membrane interactions for the insect Sapecin [25](**Notes 1** and **2**); and (2) human β-defensin-2 (hBD-2) is biologically important. Understanding the mechanism is a necessary first step to design novel antimicrobial peptides.

2 Methods

The system-specific docking protocol uses the following strategies:

2.1 Dock Sapecin Using SymmDock to Test Its Trimeric Assembly

SymmDock is a program to dock proteins and generate protein oligomers in C_n symmetries ($n \geq 2$) [26, 27]. The program can run through a webserver http://bioinfo3d.cs.tau.ac.il/SymmDock/ or a standalone version. The program can be installed in unix environment by running ./install_SymmDock.pl from the directory with SymmDock program files.

1. Download experimental pdb structures of Sapecin (1l4v) and defensing-2 (1FD4). Prepare a pdb file with the molecule you want to dock: unit.pdb (remove hydrogen atoms if the pdb structure of the protein is obtained by NMR).

2. Create parameter file by running the script: *buildParams.pl n unit.pdb*, where n = the number of symmetric units and unit.pdb is the name of the PDB file of one unit. The script will create parameter file named params.txt. All the parameters are explained within the parameter file.
3. Create the Connolly surface for the molecule by running the script: *runMSPoint.pl*.
4. Additional input may include potential binding site residues for the molecule, which reduces running time and improve the docking quality. The format of the active site file is as follows: each line includes residue number and chain id for one residue. For example

   347 A

   348 A

   The name of the file with the binding site residues is specified in the parameter file. Add or uncomment the activeSiteParams line: *activeSiteParams siteFile.txt 2 0.7*.

   Binding site residues can be used in the matching and scoring stage. The integer parameter of activeSiteParams can control the usage of the binding site in the matching stage: (0) don't use, (1) use only for first base point, (2) use for both base points. The last parameter (0.7 here) is for the scoring stage, which specifies the minimal ratio of the active site score in the results. Docking solutions with smaller ratios are discarded.
5. Running the symmetry dock program: *symm_dock.Linux <params_file> <output_file>*

   The *params_file* is the parameter file "params.txt" that was previously created by "buildParams.pl". *output_file* is the name of the file that will include the results, which contain the ranking and transformation matrix to create docked pdb structures.

   Each line represents one solution, with the following format:

   # |score | pen. |int. area| as1 | as2 | desolv. | Transformation

   1 | 6967 | −2.72 | 1761.00 | 0 | 0 | 461.34 | −2.04 −1.07 −2.82 34.36 2.80 19.23

   #—trans number

   score—geometric score

   pen.—maximal surface penetration of surface points

   int. area—buried surface area of the interface

   as1—geometric score based only on residues that were given as potential binding site for one side of the interface

   as2—geometric score based only on residues that were given as potential binding site for other side of the interface

   desolv.—DeLisi desolvation energy [28]

Transformation—transformation matrix to generate oligomer structure: three rotational angles and three translational parameters

6. Generating docked PDB files by running: *transOutput.pl output file n1 n2*.

   The output file is the file created by the program earlier. *n*1 and *n*2 are the numbers of transformations to generate. For example running: "*transOutput.pl output.txt 1 10*" will create PDB files with the first ten transformations. The script generates a file named result.transNumber.pdb, where 'number' is the transformation number.

2.2 Rescore the Docking Solution Using DFIRE2 to Evaluate the Protein–Protein Interaction Energy

1. DFIRE2 is a program to calculate protein–protein interactions using knowledge-based functions [29, 30], which is available from http://sparks-lab.org/. For each solution generated from Symmetry docking, DFIRE2 energy can be evaluated by running: *DFIRE dfire_pair.lib result.transNumber.pdb*.
2. Refine the scoring function for defensin assembly in the membrane using experimental information as a guide. Normal docking and scoring functions are designed for interactions in aqueous solution or in the crystal complex. In order to reevaluate the docking solutions specifically for defensin in the membrane environment, we re-designed the scoring function to rank the docked defensin oligomer as:

$$E_{\text{membrane}} = \text{binding-energy}^{*}a + \text{desolvation-energy}^{*}b + \text{interface-area}^{*}c$$

   Where the parameters a, b, and c are to be optimized from docking of the Sapecin trimer in the membrane to fit experimental observations. The binding energy is calculated with the DFIRE2, and the desolvation energy and the interface area are calculated with the SymmDock. Based on extensive docking of the Sapecin and re-ranking of the solution to fit experimental binding modes, we obtained the optimized parameters: $a = 1$, $b = 0.006$, and $c = 0.003$ (**Note 3**).

2.3 Docking and Ranking the Human Defensin-2 Oligomers

1. Repeat the symmetry docking procedure using human β-defensin-2 (hBD-2) dimer structure as input to construct hBD-2 octamers.
2. Using the optimized scoring function to evaluate the hBD-2-octamer in the membrane. The flowchart of the computational approaches is in Fig. 1. (**Notes 4** and **5**)

3 Notes

1. NMR experiments have indicated the likely oligomeric state for Sapecin in the membrane, with Asp4 and Arg23 intermolecular interactions [25]. Our docking and rescoring has identified the two best solutions that have arrangements similar to the conformers suggested from experiment (Fig. 2).
2. In solution, the NMR structure for hBD-2 does not show oligomerization. However, crystal structures of defensins indicate dimerization and higher oligomerization states [31, 32]. A crystal packing pattern of human defensin might also provide information regarding pore formation in the membrane. A pore formed by an octameric assembly could accommodate four water molecules [31]. The question is, though, if the assembly will re-arrange in the membrane. We try to use the parameters developed from docking of Sapecin to investigate the potential oligomerization states of the hBD-2 in the membrane.
3. We apply the scoring functions developed from the Sapecin oligomer to probe the oligomerization of hBD-2. The new scoring function clearly helps to identify possible channel forming oligomers. The 20 top ranking octamers have many candidate structures with appropriate channel forming orientations (Table 1 and Fig. 3).
4. High level simulations, for example, explicit water molecular dynamics simulations can be performed using the selected hBD-2 oligomers to examine oligomer stability and membrane disruption. Our studies of a similar protein, the pg-1 monomer and dimer, on the membrane surface [33] or in the membrane

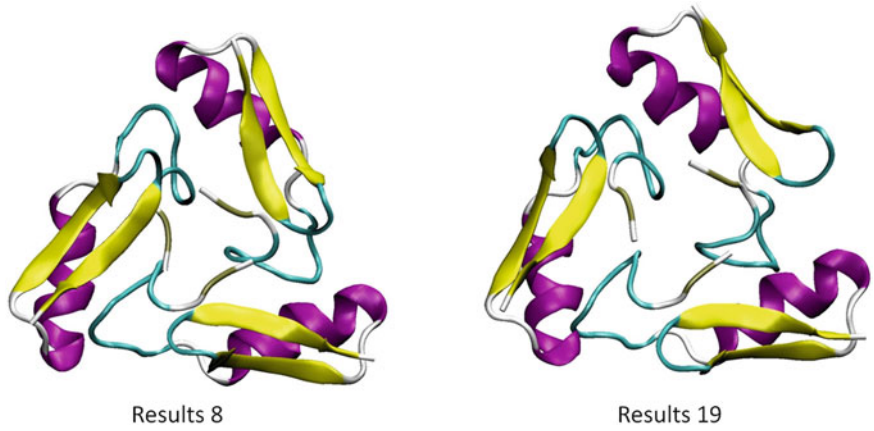


Fig. 2 Two best conformations with the highest ranking in optimized score function, which fit NMR observation, the *numbers* indicate the ranking from the initial symmetry docking

Table 1
Top 20 ranked hBD2 octamers from symmdock with new scoring function

Rank	Score (new ranking)	Result (symmetry dock ranking)
1	14.42336	result.103.pdb
2	13.41008	result.216.pdb
3	12.74822	result.172.pdb
4	12.48742	result.74.pdb
5*	12.48012	result.193.pdb
6	12.3413	result.97.pdb
7	12.02262	result.170.pdb
8	11.99962	result.251.pdb
9	11.91628	result.151.pdb
10	11.77326	result.133.pdb
11	11.53318	result.9.pdb
12*	11.4873	result.270.pdb
13	11.33686	result.175.pdb
14	11.27296	result.295.pdb
15	11.23988	result.429.pdb
16	11.14096	result.23.pdb
17	11.09774	result.255.pdb
18*	11.07944	result.87.pdb
19*	11.06372	result.124.pdb
20	10.91142	result.298.pdb

The conformers with * are pore-forming octamers

[34], have shown that MD simulations are a powerful tool to investigate membrane disruption by antimicrobial peptides.

5. The outlined approach may not be restricted to symmetric oligomerization. Other protein–protein docking programs can also be used for protein engineering. Such programs include, but not limited to, PatchDock [35] and FireDock [36].

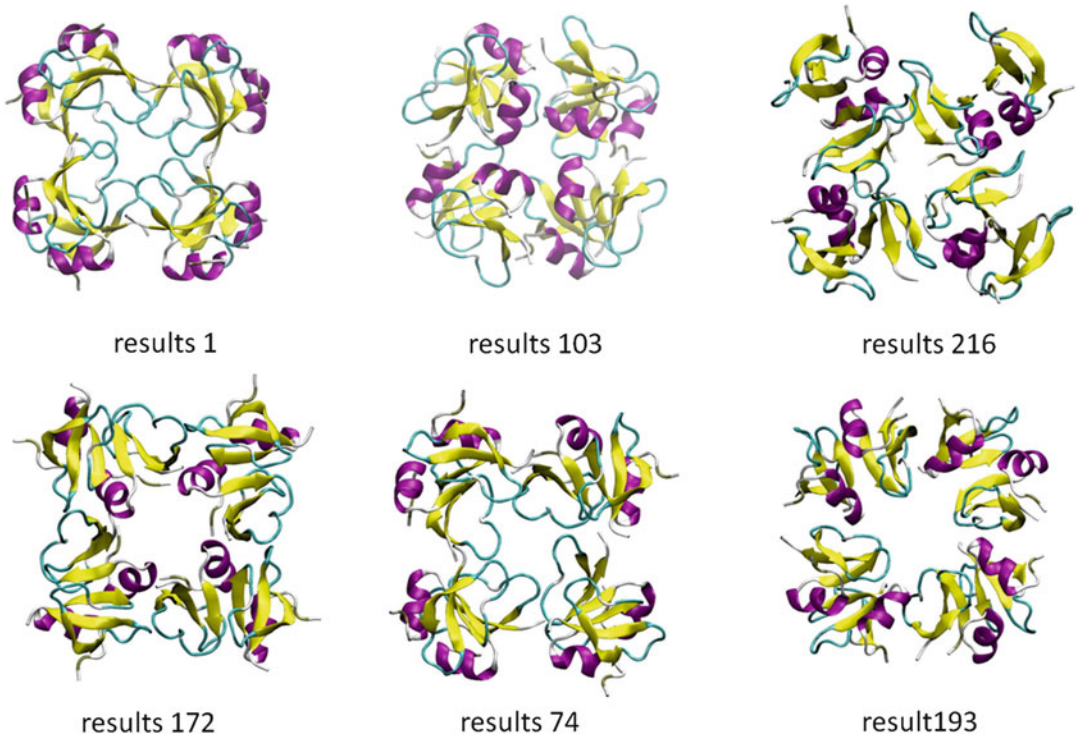


Fig. 3 The octamer structures of hBD2 with pore conformation obtained from symmetry docking. The *numbers* indicate the ranking from the initial symmetry docking. The result 1 is the top ranking structure from the initial symmetry docking, and results 103, 216, 172, 74, and 193 are five top ranking structure from our reoptimized score function in membrane. It can be seen that the top re-ranked structures have larger pore sizes

4 Conclusion

Re-parameterizing symmetry dock for membrane environment can provide insight into the oligomerization structures of the membrane damaging antibacterial defensin in membrane. If combined with high level simulations in further optimization of protein structure and sequence, the integrated approach could be a valuable method in computational protein design (**Notes 4** and **5**).

Acknowledgments

The authors are funded in whole or in part with Federal funds from the National Cancer Institute, National Institutes of Health, under contract number HHSN261200800001E. This research was supported (in part) by the Intramural Research Program of the NIH, National Cancer Institute, Center for Cancer Research. This research was funded in part by the US Army Medical Research

Acquisition Activity under grant W81XWH-05-1-0002. N.Z. thanks Chinese NFSC grant 30772529 and 973 program grants 2011CB933100 and 2010CB933900.

References

1. Nanda V, Hsieh D, Davis A (2013) Prediction and design of outer membrane protein-protein interactions. Methods Mol Biol 1063:183–196. doi:10.1007/978-1-62703-583-5_10
2. Naveed H, Liang J (2014) Weakly stable regions and protein-protein interactions in beta-barrel membrane proteins. Curr Pharm Des 20(8):1268–1273
3. Ganz T (2003) Defensins: antimicrobial peptides of innate immunity. Nat Rev Immunol 3 (9):710–720
4. Chandrababu KB, Ho B, Yang D (2009) Structure, dynamics, and activity of an all-cysteine mutated human beta defensin-3 peptide analogue. Biochemistry 48(26):6052–6061. doi:10.1021/bi900154f
5. Wilson SS, Wiens ME, Smith JG (2013) Antiviral mechanisms of human defensins. J Mol Biol 425(24):4965–4980. doi:10.1016/j.jmb.2013.09.038
6. Garzino-Demo A (2007) Chemokines and defensins as HIV suppressive factors: an evolving story. Curr Pharm Des 13(2):163–172
7. Krishnakumari V, Nagaraj R (2012) Binding of peptides corresponding to the carboxy-terminal region of human-beta-defensins-1-3 with model membranes investigated by isothermal titration calorimetry. Biochim Biophys Acta 1818(5):1386–1394. doi:10.1016/j.bbamem.2012.02.016
8. Krishnakumari V, Rangaraj N, Nagaraj R (2009) Antifungal activities of human beta-defensins HBD-1 to HBD-3 and their C-terminal analogs Phd1 to Phd3. Antimicrob Agents Chemother 53(1):256–260. doi:10.1128/AAC.00470-08
9. Brogden KA (2005) Antimicrobial peptides: pore formers or metabolic inhibitors in bacteria? Nat Rev Microbiol 3(3):238–250. doi:10.1038/nrmicro1098
10. Spudy B, Sonnichsen FD, Waetzig GH, Grotzinger J, Jung S (2012) Identification of structural traits that increase the antimicrobial activity of a chimeric peptide of human beta-defensins 2 and 3. Biochem Biophys Res Commun 427(1):207–211. doi:10.1016/j.bbrc.2012.09.052
11. Voss E, Wehkamp J, Wehkamp K, Stange EF, Schroder JM, Harder J (2006) NOD2/CARD15 mediates induction of the antimicrobial peptide human beta-defensin-2. J Biol Chem 281(4):2005–2011. doi:10.1074/jbc.M511044200
12. Gariboldi S, Palazzo M, Zanobbio L, Selleri S, Sommariva M, Sfondrini L, Cavicchini S, Balsari A, Rumio C (2008) Low molecular weight hyaluronic acid increases the self-defense of skin epithelium by induction of beta-defensin 2 via TLR2 and TLR4. J Immunol 181 (3):2103–2110
13. Kanda N, Watanabe S (2008) IL-12, IL-23, and IL-27 enhance human beta-defensin-2 production in human keratinocytes. Eur J Immunol 38(5):1287–1296. doi:10.1002/eji.200738051
14. Kuwano K, Tanaka N, Shimizu T, Kida Y (2006) Antimicrobial activity of inducible human beta defensin-2 against Mycoplasma pneumoniae. Curr Microbiol 52(6):435–438. doi:10.1007/s00284-005-0215-7
15. Droin N, Hendra JB, Ducoroy P, Solary E (2009) Human defensins as cancer biomarkers and antitumour molecules. J Proteomics 72 (6):918–927. doi:10.1016/j.jprot.2009.01.002
16. Gerashchenko O, Zhuravel E, Skachkova O, Khranovska N, Pushkarev V, Pogrebnoy P, Soldatkina M (2014) Involvement of human beta-defensin-2 in regulation of malignant potential of cultured human melanoma cells. Exp Oncol 36(1):17–23
17. Zhuravel E, Shestakova T, Efanova O, Yusefovich Y, Lytvin D, Soldatkina M, Pogrebnoy P (2011) Human beta-defensin-2 controls cell cycle in malignant epithelial cells: in vitro study. Exp Oncol 33(3):114–120
18. Jarczak J, Kosciuczuk EM, Lisowski P, Strzalkowska N, Jozwik A, Horbanczuk J, Krzyzewski J, Zwierzchowski L, Bagnicka E (2013) Defensins: natural component of human innate immunity. Hum Immunol 74(9):1069–1079. doi:10.1016/j.humimm.2013.05.008
19. Bai Y, Liu S, Jiang P, Zhou L, Li J, Tang C, Verma C, Mu Y, Beuerman RW, Pervushin K (2009) Structure-dependent charge density as a determinant of antimicrobial activity of peptide analogues of defensin. Biochemistry 48 (30):7229–7239. doi:10.1021/bi900670d

20. Taylor K, Barran PE, Dorin JR (2008) Structure-activity relationships in beta-defensin peptides. Biopolymers 90(1):1–7. doi:10.1002/bip.20900
21. Zhang Y, Lu W, Hong M (2010) The membrane-bound structure and topology of a human alpha-defensin indicate a dimer pore mechanism for membrane disruption. Biochemistry 49(45):9770–9782. doi:10.1021/bi101512j
22. Morgera F, Antcheva N, Pacor S, Quaroni L, Berti F, Vaccari L, Tossi A (2008) Structuring and interactions of human beta-defensins 2 and 3 with model membranes. J Pept Sci 14 (4):518–523. doi:10.1002/psc.981
23. Suresh A, Verma C (2006) Modelling study of dimerization in mammalian defensins. BMC Bioinformatics 7(Suppl 5):S17. doi:10.1186/1471-2105-7-S5-S17
24. Fung HK, Floudas CA, Taylor MS, Zhang L, Morikis D (2008) Toward full-sequence de novo protein design with flexible templates for human beta-defensin-2. Biophys J 94 (2):584–599. doi:10.1529/biophysj.107.110627
25. Takeuchi K, Takahashi H, Sugai M, Iwai H, Kohno T, Sekimizu K, Natori S, Shimada I (2004) Channel-forming membrane permeabilization by an antibacterial protein, sapecin: determination of membrane-buried and oligomerization surfaces by NMR. J Biol Chem 279 (6):4981–4987
26. Schneidman-Duhovny D, Inbar Y, Nussinov R, Wolfson HJ (2005) PatchDock and SymmDock: servers for rigid and symmetric docking. Nucleic Acids Res 33(Web Server issue): W363–W367
27. Schneidman-Duhovny D, Inbar Y, Nussinov R, Wolfson HJ (2005) Geometry-based flexible and symmetric protein docking. Proteins 60 (2):224–231. doi:10.1002/prot.20562
28. Zhang C, Vasmatzis G, Cornette JL, DeLisi C (1997) Determination of atomic desolvation energies from the structures of crystallized proteins. J Mol Biol 267(3):707–726. doi:10.1006/jmbi.1996.0859
29. Zhang C, Liu S, Zhu Q, Zhou Y (2005) A knowledge-based energy function for protein-ligand, protein-protein, and protein-DNA complexes. J Med Chem 48(7):2325–2335
30. Yang Y, Zhou Y (2008) Specific interactions for ab initio folding of protein terminal regions with secondary structures. Proteins 72 (2):793–803. doi:10.1002/prot.21968
31. Hoover DM, Rajashankar KR, Blumenthal R, Puri A, Oppenheim JJ, Chertov O, Lubkowski J (2000) The structure of human beta-defensin-2 shows evidence of higher order oligomerization. J Biol Chem 275(42):32911–32918
32. Hoover DM, Chertov O, Lubkowski J (2001) The structure of human beta-defensin-1: new insights into structural properties of beta-defensins. J Biol Chem 276(42):39021–39026
33. Jang H, Ma B, Nussinov R (2007) Conformational study of the protegrin-1 (PG-1) dimer interaction with lipid bilayers and its effect. BMC Struct Biol 7:21
34. Jang H, Ma B, Lal R, Nussinov R (2008) Models of toxic beta-sheet channels of protegrin-1 suggest a common subunit organization motif shared with toxic Alzheimer beta-amyloid ion channels. Biophys J 95(10):4631–4642
35. Schneidman-Duhovny D, Inbar Y, Polak V, Shatsky M, Halperin I, Benyamini H, Barzilai A, Dror O, Haspel N, Nussinov R, Wolfson HJ (2003) Taking geometry to its edge: fast unbound rigid (and hinge-bent) docking. Proteins 52(1):107–112. doi:10.1002/prot.10397
36. Mashiach E, Schneidman-Duhovny D, Andrusier N, Nussinov R, Wolfson HJ (2008) FireDock: a web server for fast interaction refinement in molecular docking. Nucleic Acids Res 36(Web Server issue):W229–W232. doi:10.1093/nar/gkn186

Chapter 19

Computational Design of Ligand Binding Proteins

Christine E. Tinberg and Sagar D. Khare

Abstract

The ability to design novel small-molecule binding sites in proteins is a stringent test of our understanding of the principles of molecular recognition, and would have many practical applications, in synthetic biology and medicine. Here, we describe a computational method in the context of the macromolecular modeling suite Rosetta to designing proteins with sites featuring predetermined interactions to ligands of choice. The required inputs for the method are a model of the small molecule and the desired interactions (e.g., hydrogen bonding, electrostatics, steric packing), and a set of crystallographic structures of proteins containing existing or predicted binding pockets. Constellations of backbones surrounding the putative pocket are searched for compatibility with the desired binding site conception using RosettaMatch and surrounding amino acid side chain identities are optimized using RosettaDesign. Validation of the design is performed using metrics that evaluate the interface energy of the predicted binding pose, the preformation of key binding site features in the apo-state, and the local compatibility of the designed sequence changes with the wild type backbone structure, and top-ranking candidate designs are generated for experimental validation. This approach can allow for the creation of novel binding sites and for the rational tuning of specificity for congeneric ligands by altering the programmed interactions by design, thus offering a general computational protocol for construction and modulation of protein–small molecule interfaces.

Key words Protein design, Rosetta software, Ligand binding, Small molecule binding, Steroid binding

1 Introduction

The ability to de novo design binding sites for small molecules with programmable binding affinities and selectivities encoded by predefined interactions will have many practical applications in synthetic biology and medicine, including the construction of small molecule-responsive genetic circuits, novel biosensors, and therapeutic scavengers for preventing drug overdoses. Current approaches for designing ligand binding proteins for medical [1] and biotechnological uses rely upon raising antibodies against a target antigen in immunized animals [2, 3] and/or performing laboratory directed evolution of proteins with an existing low affinity for the desired ligand [4–6], both of which offer incomplete

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_19,

control over molecular details. Computational design could provide a general, complementary approach for small molecule recognition in which design features and selectivity are rationally programmed.

Recent advances in computational protein design have resulted in novel enzymes with bio-orthogonal functions [7–9], but the catalytic efficiencies of these designed biocatalysts are modest compared to those of their natural counterparts [10]. One reason for the low efficiencies is inaccurate modeling of protein–ligand interactions: crystal structures of several designed enzymes bound to substrate analogs show that although many of the designed residues adopt their modeled conformations, the ligands and some key catalytic side chains are oriented differently than in the computational models [11]. Achieving accuracy in the computational design of protein–ligand interfaces would, therefore, also aid in the design of high efficiency novel biocatalysts.

Native protein–small molecule interfaces are defined by three main characteristics: (1) highly optimized specific interactions between protein and ligand, such as hydrogen bonding, electrostatic, and van der Waals interactions, (2) high overall shape complementarity, and (3) pre-organization of interacting side chains in binding competent conformations in the unbound protein state [12]. Guided by these observations, we developed a method in the framework of the macromolecular modeling software Rosetta to introduce preselected interactions to a chosen ligand (the steroid digoxigenin, DIG) in a set of scaffold proteins. The protocol described below uses DIG as an example, but can, in principle, be extended to any small molecule (**Note 1**). Methodological challenges and the caveats associated with the choice of ligand with varying physiochemical properties (high charge, high flexibility) are likely to determine generalizability; these are also described in Subheading 3.

2 Methods

Starting from a model of the small molecule of choice (DIG) interacting with protein side chain functional groups and a set of Protein Data Bank (PDB) files with existing or predicted binding pockets, we computationally generate a design model and evaluate it. The overall workflow involves the following steps:

1. Generation of ligand and ligand conformer library.
2. Protein scaffold selection.
3. Geometric placement of ligand using a set of preselected interactions.
4. Rosetta sequence design.

5. Evaluation of designs.
6. Compatibility of designed sequence with local backbone structure.
7. Final Design Selection.
8. Protein expression, experimental characterization, and affinity maturation (not discussed here).

2.1 Generation of Models for the Ligand and Ligand Conformer Library

The three-dimensional structure of the ligand of choice can be obtained from a small molecule (e.g., Cambridge Crystallographic Database) or macromolecular (e.g., PDB) structure database. The latter is generally preferable as this describes a protein-bound structure and is likely to not suffer from artifacts arising from packing in a small molecule crystal. In our case, the three-dimensional structure of DIG (Fig. 1a) was obtained from PDB file 1LKE [13]. Because our experimental validation and selection methods relied on the presence of a linker that connects the DIG molecule (Fig. 1b) to either biotin or carrier protein, we included this linker in our ligand model. Linker atoms were added to DIG

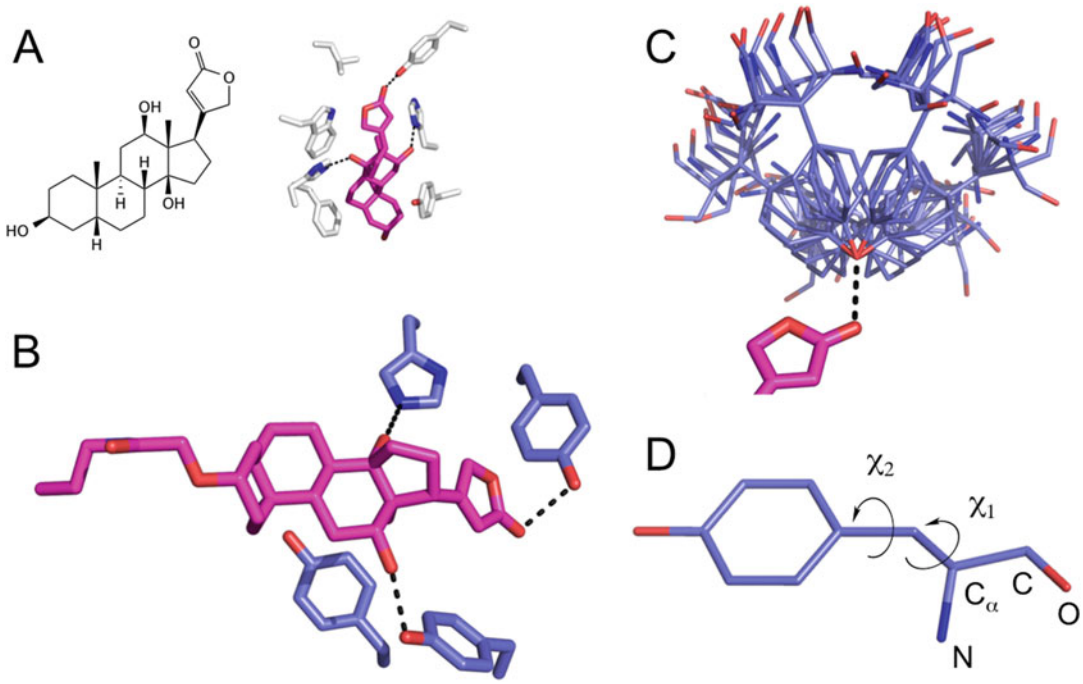


Fig. 1 Generating the desired binding site model. (**a**) The structure of DIG and its binding mode in an engineered lipocalin that binds the small molecule with high affinity (PDB code: 1LKE). (**b**) One of the binding conceptions for DIG in which polar amino acid side chain groups make defined hydrogen bonds to DIG and aromatic side chain groups make steric interactions. (**c**) In the modeled disambiguated active sites, all possible functional group orientations that can make the desired interaction (hydrogen bond in the case depicted) are considered. (**d**) Side chain dihedral angles are also sampled to generate these disambiguated binding sites

using the Build functionality of MacPyMOL (Schrödinger, LLC). A ligand conformer library was generated by sampling conformations around the relevant degrees of freedom, in our case C3–O5 and N1–C26 bonds (Fig. 1) at −60° ± 30°, 60° ± 30°, and 180° ± 30°. Conformers were rejected if there were significant clashes within the molecule by using an intra_fa_rep cutoff value of 0.25 Rosetta energy units (Reu). User inputs in choosing the internal degrees of freedom of the small molecule and the fine-ness of the sampling these conformations is desirable. Ligand ensembles can be generated from Rosetta or custom software such as Omega. Desired interactions with the ligand also should be identified at this stage. We chose hydrogen bonds and hydrophobic interactions as these were observed in crystal structures of DIG bound proteins (Fig. 1b) and idealized binding sites incorporating these functional groups along with compatible geometries were enumerated using RosettaMatch (Fig. 1c, d; also *see* below).

2.2 Scaffold Selection

A set of several hundred scaffolds for use as input structures is typically generated. It is desirable that this set contains a variety of structural and functional classes for maximizing diversity of candidate designs. In our case with DIG, we included periplasmic binding proteins and lipid-binding proteins [8, 9, 14], as well as 344 structural homologs [15] of a subset of highly expressing scaffolds (PDB codes 1m4w, 1oho, 1a53, 1thf, 1dl3, and 1e1a) having a DALI Z-score cutoff value of eight from the input search model. These six scaffolds were chosen because of previous enzyme-design successes in these fold classes [8, 9, 14] and/or because of their thermostability, as directed evolution experiments have shown that more stable scaffolds can acquire new functions more easily than their less stable counterparts [16, 17]. The homolog subset comprised 8 Concanavalin A-like lectins/glucanases (homologs of 1m4w from *Nonomurea flexuosa*), 91 cystatin-like proteins (homologs of 1oho from *Pseusomonas putida*), 208 TIM β/α-barrels (28 homologs of 1a53 from *Sulfobus solfataricus*, 46 unique homologs of 1thf from *Thermotoga maritima*, and 134 unique homologs of 1dl3 from *Thermotoga maritima*), and 37 6-bladed β-propeller proteins (homologs of 1e1a from *Loligo vulgaris*). All of these proteins are enzymes that bind small molecule substrates, but not all proteins contain a bound ligand in their crystallographic structures. All 401 scaffolds comprise <350 amino acids, have been expressed previously in *E. coli*, and were stripped of their cognate bound small molecules and water molecules before use. To identify residue positions to be used for matching (*see* below) in the homolog scaffolds, each homolog crystal structure was superimposed on that of its parent scaffold using the CEAlign plug-in of the PyMOL molecular visualization program, and then homolog residue positions within 5.0 Å of any ligand heavy atom present in the parent scaffold were identified. For PDBs 1a53, 1dl3, and 1oho, ligands

present in the crystal structures were used in this search. For 1m4w, 1e1a, and 1thf, ligand positions from the computational design models of a retroaldolase (RA60) [8], a Diels-Alderase (DA_20) [9], and a Kemp Eliminase (KE_007) [14] were used, respectively.

2.3 Geometric Placement of Ligand Using a Set of Preselected Interactions

Geometric criteria for enforcing binding site interactions can be obtained from existing structures that feature the ligand, or from quantum mechanical or molecular mechanics-based calculations as performed for theozymes. The former approach has the advantage that existing binding modes may have a higher chance of being recapitulated by design whereas the latter allow for sampling novel binding modes, and would be the preferred choice when no bound structures are available. For DIG, these binding interactions were determined by inspecting structures of digoxin bound to the anti-digoxigenin antibody 26-10, PDB ID 1IGJ [18], and of digoxigenin bound to the engineered lipocalin DigA16, PDB ID 1LKE [13]. From these structures we defined five interface criteria: (1) hydrogen bond between the lactone carbonyl oxygen and a Tyr side chain, (2) hydrogen bond between the O2 hydroxyl and a histidine or Tyr side chain, (3) hydrogen bond between the O3 hydroxyl and a His or Tyr side chain, (4) hydrophobic packing interaction on the top face of the ligand, and (5) hydrophobic packing interaction on the bottom face of the ligand (Fig. 1c). Two active site configurations were specified: one having Tyr, Tyr, His, Phe/Tyr, and Phe/Tyr/Trp satisfying design criteria 1–5 (DIG_yyhff), and one having Tyr, His, His, Phe/Tyr/Trp, and Tyr/Trp satisfying design criteria 1–5 (DIG_yhhff).

Geometric criteria were defined using six degrees of freedom between the ligand and the desired interacting side chain using a matching constraints file [19]. Extra rotamer sampling (two half step standard deviations) was performed around all side chain torsion angles (Fig. 1d). To enforce burial of the lactone head group within a binding pocket, we considered only those residue positions in the binding site that had a minimum of 14 neighboring residues during matching for constraint 1 (hydrogen bond to the lactone carbonyl oxygen). A neighbor was defined as a residue having Cα within 10 Å of the Cα of the binding site position under consideration. Secondary matching [19] was used for constraints 3, 4, and 5. To eliminate high-energy rotamer conformations, a maximum Dunbrack energy (fa_dun) cutoff of 4.5 Reu (unweighted) was used while building rotamers for all constraints. Using these matching criteria, 29,274 and 30,861 matches were found for the two different binding conceptions, DIG_yyhff and DIG_yhhff, respectively (Fig. 2).

2.4 Rosetta Sequence Design

Active site amino acid sequences of each match are designed to maximize binding affinity to the ligand according to the Rosetta energy function [19, 20]. Design moves are followed by steepest

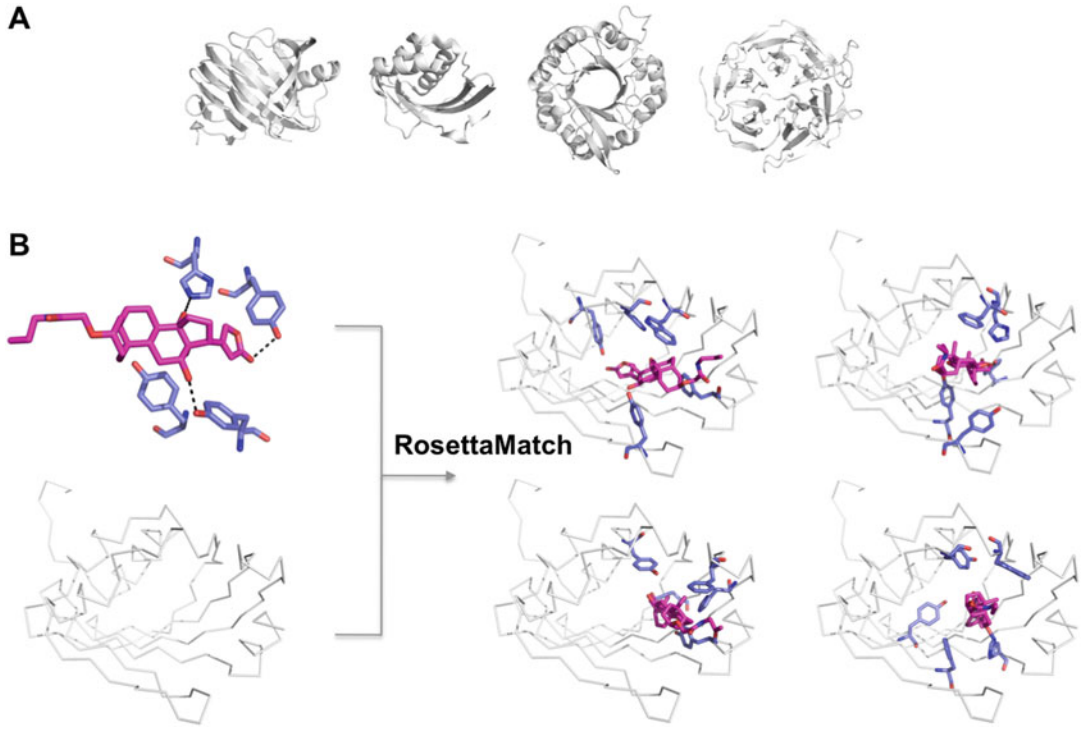


Fig. 2 Matching binding sites to scaffolds. (**a**) A set of scaffolds with varying topology are chosen from the PDB and, (**b**) Backbone constellations compatible with the envisioned binding sites as well as space for the ligand molecule are searched in the scaffold set using the RosettaMatch algorithm. A diversity of scaffold topologies and high stability are generally desirable for higher success rates

descent gradient minimization in which side chain degrees of freedom and the relative orientation of the ligand with respect to the protein are allowed to minimize freely [21] but backbone minimization is restricted such that Cα atoms were only allowed to move $\leq$0.05 Å from their pre-minimization positions. Internal torsions of the ligand can be allowed to minimize but typically are constrained to be within 5° of their initial values (**Note 2**).

Two successive rounds of sequence design can be used to generate designs. The purpose of the first round is to maximize binding affinity for the ligand [19]. To prevent destabilization of the apo-protein that can result from mutating potentially stabilizing residues having side chains important for core packing, aromatic residues in the scaffold can be allowed to mutate to other aromatics during this round of design. A RosettaScripts XML file entitled ligdes.xml for running the first round of sequence design is provided.

After the first round, a second round of binding site sequence design is performed on the output from the first round. The goal of this round is to optimize protein stability while maintaining the binding interface designed during the first round as much as

possible. Ligand–protein interactions are up-weighted by a factor of 1.5 relative to intra-protein interactions during sequence optimization in attempt to ensure that the interface binding affinity is maintained, and two different criteria are used to optimize protein stability: (1) native scaffold residues identities are favored by 1.5 Rosetta energy units (Reu), and (2) no more than five residues are allowed to change from identities observed in a multiple sequence alignment (MSA) if (a) these residues are present in the MSA with a frequency greater than 0.6 as specified by a position-specific sequence matrix (PSSM) and, (b) if the calculated ΔΔG for mutation of the scaffold residue to alanine was greater than 1.5 Reu in the context of the wild type scaffold sequence. The ΔΔG for mutation to alanine can be estimated as described [22] and PSSM files can be generated using NCBI PSI-BLAST. RosettaScripts XML files, ligdes_fix_cst.xml and ligdes_flex_hb.xml, respectively, are provided.

2.5 Evaluation of Designs

Designs passing the filters encoded in the XML files (see attached files), including the calculated interface energy (Fig. 3a), are subjected to several additional filtering criteria (also *see* **Notes 3** and **4**). High shape complementary (Fig. 3b) is enforced by rejecting designs having $S_c < 0.6$. Shape complementary is computed using the CCP4 package v.6.0.2 [23] using the S_c program [24] and the Rosetta radii library. A common feature of the many high-affinity protein small molecule interfaces, e.g., engineered DIG-binding lipocalin DigA16 (PDB IDs 1LKE and 1KXO) [13] and the anti-DIG 26-10 antibody (PDB IDs 1IGJ and 1IGI) [18], is that the binding site is largely pre-organized; there are very few structural changes between the bound and unbound forms of the proteins. We therefore attempt to enforce pre-organization of the binding-competent conformation of the apo-protein by two metrics: (1) introducing second-shell amino acids that hold the preselected residues in place via hydrogen bonding or sterics using Foldit [25], and (2) eliminating designs having Boltzmann-weighted side chain probabilities [26] <0.1 for more than one of the key binding residues (Fig. 3c).

2.6 Compatibility of Designed Sequence with Local Backbone Structure

Binding site pre-organization would be further compromised if substitution of amino acid side chains during (fixed backbone) design leads to a change in the backbone conformational preference in regions sequence-local to the sites of substitution. Therefore, we developed a metric to estimate the impact of design on local backbone structure and use this metric to discard designs that are predicted to lead to backbone structure changes (Fig. 3d). Using the structure prediction modules of Rosetta [27], we generate a set of 9-mer fragment structures for each designed and wild type scaffold sequence and compare the average RMSD of these fragments to those of the scaffold backbone structures. If the average

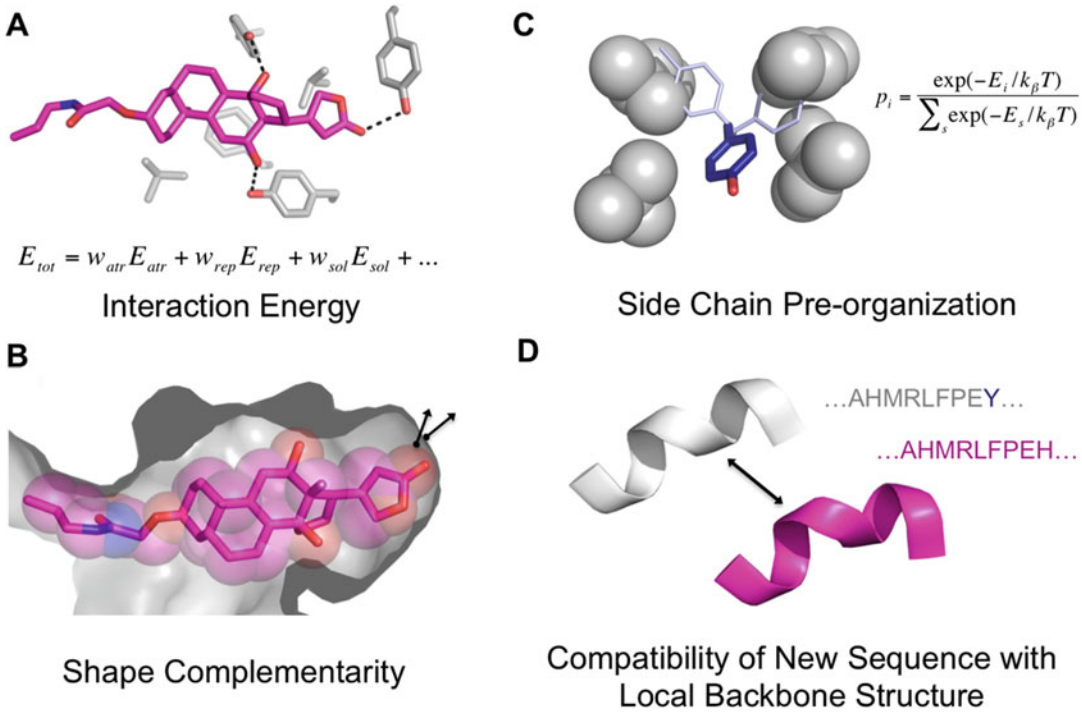


Fig. 3 Design Selection Criteria. (**a**) Interface energy as calculated by the Rosetta energy function, (**b**) Geometric shape complementarity between the molecular surfaces of the ligand and the protein cavity, (**c**) Pre-organization of key side chain groups is a strong predictor of design success, and (**d**) The compatibility of the designed sequence to adopt the scaffold backbone

RMSD of conformations predicted in these fragments (200 9-mers) near any designed position is greater (>0.8 Å) for the designed sequence than the wild type scaffold sequence, we flag that region of the designed protein as unlikely to adopt the local backbone conformation of the scaffold protein and reject that designed protein.

2.7 Final Design Selection

Following automated filtering, all designs are inspected manually using Foldit [25] and some ligand-proximal residues are manually reverted back to their native scaffold identity to increase the likelihood of design stability. Typically, for every binding site conception, a total of ~10–20 designs are selected with each design featuring ~5–20 substitutions compared to the wild type scaffold. Finally, synthetic genes corresponding to these designed proteins can be ordered and the designs can be experimentally tested and evaluated as described elsewhere [28].

3 Notes

While the ultimate goal of computational design methods is to automate all design steps, in practice most protocols rely upon the chemical intuition and domain knowledge of the user. Our method is no exception and so below we give some suggestions about aspects that need to be considered by the user while evaluating the designs generated by the protocol described above.

1. The Rosetta force field, as other molecular mechanics force fields, does not accurately model all interactions of protein functional groups, especially functional interactions that are introduced to encode selectivity at the expense of local instabilities. Furthermore, accurate modeling of charge–charge interactions in proteins and between protein functional groups and ligands remains a challenge. Therefore, the design of proteins to bind charged ligands is generally considerably more difficult than for polar and hydrophobic ligands. Even for the latter, it may be necessary to treat the pre-defined binding interactions with geometrical restraints to ensure binding selectivity. The weights used in the geometrically defined restraints will be system dependent and may require tuning.
2. In the generation of the ligand ensemble or during minimization with Rosetta, it is useful to vary some internal torsional angles of a ligand model, but the resulting conformations may not be the global energy minimum conformations of the ligand. The Rosetta force field has several database-derived terms that make it suitable for protein design but these terms are generally inapplicable to scoring ligand conformational ensembles. Furthermore, ligand conformational entropy loss upon binding is largely ignored in the method. Therefore, the design of highly flexible ligands is likely more difficult than design with rigid ones. Practically, ensembles that are too large may be expensive to deal with computationally, but significant errors can be accrued if the sampling is too coarse. Therefore, the variance in the conformational ensembles of the ligand models should be chosen carefully and tuned to strike a balance between chemical accuracy and computational cost.
3. A metric that is currently evaluated by human intuition in our protocol is whether the ligand can enter the designed active site and that access to the active site has not been blocked by new mutations introduced in the design protocol. Conformational changes upon substrate binding are not modeled and system-dependent knowledge of the dynamics of the closure and opening of the active site should be kept in mind when picking out scaffolds for design and/or evaluating designs by inspection.

4. Chemical intuition is almost always required to evaluate the goodness of designs and the goal of all protocols is generally to provide the user with as many "good" designs with their plausible binding modes. With continuing method developments, computational algorithms will increase the fraction of "good" designs, but the need for solid chemical intuition is unlikely to diminish.

References

1. de Wolf FA, Brett GM (2000) Ligand-binding proteins: their potential for application in systems for controlled delivery and uptake of ligands. Pharmacol Rev 52(2):207–236
2. Hunter MM, Margolies MN, Ju A, Haber E (1982) High-affinity monoclonal antibodies to the cardiac glycoside, digoxin. J Immunol 129 (3):1165–1172
3. Shen XY, Orson FM, Kosten TR (2012) Vaccines against drug abuse. Clin Pharmacol Ther 91:60–70
4. Bradbury ARM, Sidhu S, Dübel S, McCafferty J (2011) Beyond natural antibodies: the power of *in vitro* display technologies. Nat Biotechnol 29(3):245–254
5. Brustad EM, Arnold FH (2011) Optimizing non-natural protein function with directed evolution. Curr Opin Chem Biol 15(2):201–210. doi:10.1016/j.cbpa.2010.11.020
6. Chen G, Hayhurst A, Thomas JG, Harvey BR, Iverson BL, Georgiou G (2001) Isolation of high-affinity ligand-binding proteins by periplasmic expression with cytometric screening (PECS). Nat Biotechnol 19(6):537–542
7. Rothlisberger D, Khersonsky O, Wollacott AM, Jiang L, DeChancie J, Betker J, Gallaher JL, Althoff EA, Zanghellini A, Dym O, Albeck S, Houk KN, Tawfik DS, Baker D (2008) Kemp elimination catalysts by computational enzyme design. Nature 453(7192):190–195. doi:10.1038/nature06879
8. Jiang L, Althoff EA, Clemente FR, Doyle L, Rothlisberger D, Zanghellini A, Gallaher JL, Betker JL, Tanaka F, Barbas CF 3rd, Hilvert D, Houk KN, Stoddard BL, Baker D (2008) De novo computational design of retro-aldol enzymes. Science 319(5868):1387–1391. doi:10.1126/science.1152692
9. Siegel JB, Zanghellini A, Lovick HM, Kiss G, Lambert AR, St Clair JL, Gallaher JL, Hilvert D, Gelb MH, Stoddard BL, Houk KN, Michael FE, Baker D (2010) Computational design of an enzyme catalyst for a stereoselective bimolecular Diels-Alder reaction. Science 329 (5989):309–313. doi:10.1126/science.1190239
10. Khersonsky O, Rothlisberger D, Dym O, Albeck S, Jackson CJ, Baker D, Tawfik DS (2010) Evolutionary optimization of computationally designed enzymes: Kemp eliminases of the KE07 series. J Mol Biol 396 (4):1025–1042. doi:10.1016/j.jmb.2009.12.031
11. Khare SD, Fleishman SJ (2013) Emerging themes in the computational design of novel enzymes and protein-protein interfaces. FEBS Lett 587(8):1147–1154. doi:10.1016/j.febslet.2012.12.009
12. Fleishman SJ, Khare SD, Koga N, Baker D (2011) Restricted sidechain plasticity in the structures of native proteins and complexes. Protein Sci 20(4):753–757. doi:10.1002/pro.604
13. Korndörfer IP, Schlehuber S, Skerra A (2003) Structural mechanism of specific ligand recognition by a lipocalin tailored for the complexation of digoxigenin. J Mol Biol 330 (2):385–396
14. Röthlisberger D, Khersonsky O, Wollacott AM, Jiang L, DeChancie J, Betker J, Gallaher JL, Althoff EA, Zanghellini A, Dym O, Albeck S, Houk KN, Tawfik DS, Baker D (2008) Kemp elimination catalysts by computational enzyme design. Nature 453(7192):190–195
15. Holm L, Rosenström P (2010) Dali server: conservation mapping in 3D. Nucleic Acids Res 38(suppl 2):W545–W549
16. Bloom JD, Labthavikul ST, Otey CR, Arnold FH (2006) Protein stability promotes evolvability. Proc Natl Acad Sci U S A 103 (15):5869–5874
17. Tokuriki N, Tawfik DS (2009) Chaperonin overexpression promotes genetic variation and enzyme evolution. Nature 459(7247):668–673
18. Jeffrey PD, Strong RK, Sieker LC, Chang CY, Campbell RL, Petsko GA, Haber E, Margolies MN, Sheriff S (1993) 26-10 Fab–digoxin complex: affinity and specificity due to surface complementarity. Proc Natl Acad Sci U S A 90 (21):10310–10314

19. Richter F, Leaver-Fay A, Khare SD, Bjelic S, Baker D (2011) De novo enzyme design using Rosetta3. PLoS One 6(5), e19230
20. Fleishman SJ, Leaver-Fay A, Corn JE, Strauch E-M, Khare SD, Koga N, Ashworth J, Murphy P, Richter F, Lemmon G, Meiler J, Baker D (2011) RosettaScripts: a scripting language interface to the Rosetta macromolecular modeling suite. PLoS One 6(6), e20161
21. Davis IW, Baker D (2009) RosettaLigand docking with full ligand and receptor flexibility. J Mol Biol 385(2):381–392
22. Kellogg EH, Leaver-Fay A, Baker D (2010) Role of conformational sampling in computing mutation-induced changes in protein structure and stability. Proteins 79(3):830–838. doi:10.1002/prot.22921
23. Collaborative Computational Project N (1994) The *CCP4* suite: programs for protein crystallography. Acta Crystallogr Sect D **50**(5), 760–763. doi: 10.1107/S0907444994003112
24. Lawrence MC, Colman PM (1993) Shape complementarity at protein/protein interfaces. J Mol Biol 234(4):946–950
25. Cooper S, Khatib F, Treuille A, Barbero J, Lee J, Beenen M, Leaver-Fay A, Baker D, Popovic Z, Players F (2010) Predicting protein structures with a multiplayer online game. Nature 466(7307):756–760
26. Fleishman SJ, Khare SD, Koga N, Baker D (2011) Restricted sidechain plasticity in the structures of native proteins and complexes. Protein Sci 20(4):753–757
27. Raman S, Vernon R, Thompson J, Tyka M, Sadreyev R, Pei J, Kim D, Kellogg E, DiMaio F, Lange O, Kinch L, Sheffler W, Kim B-H, Das R, Grishin NV, Baker D (2009) Structure prediction for CASP8 with all-atom refinement using Rosetta. Proteins 77(S9):89–99. doi:10.1002/prot.22540
28. Tinberg CE, Khare SD, Dou J, Doyle L, Nelson JW, Schena A, Jankowski W, Kalodimos CG, Johnsson K, Stoddard BL, Baker D (2013) Computational design of ligand-binding proteins with high affinity and selectivity. Nature 501(7466):212–216. doi:10.1038/nature12443

Chapter 20

EpiSweep: Computationally Driven Reengineering of Therapeutic Proteins to Reduce Immunogenicity While Maintaining Function

Yoonjoo Choi, Deeptak Verma, Karl E. Griswold, and Chris Bailey-Kellogg

Abstract

Therapeutic proteins are yielding ever more advanced and efficacious new drugs, but the biological origins of these highly effective therapeutics render them subject to immune surveillance within the patient's body. When recognized by the immune system as a foreign agent, protein drugs elicit a coordinated response that can manifest a range of clinical complications including rapid drug clearance, loss of functionality and efficacy, delayed infusion-like allergic reactions, more serious anaphylactic shock, and even induced autoimmunity. It is thus often necessary to deimmunize an exogenous protein in order to enable its clinical application; critically, the deimmunization process must also maintain the desired therapeutic activity.

To meet the growing need for effective, efficient, and broadly applicable protein deimmunization technologies, we have developed the EpiSweep suite of protein design algorithms. EpiSweep seamlessly integrates computational prediction of immunogenic T cell epitopes with sequence- or structure-based assessment of the impacts of mutations on protein stability and function, in order to select combinations of mutations that make Pareto optimal trade-offs between the competing goals of low immunogenicity and high-level function. The methods are applicable both to the design of individual functionally deimmunized variants as well as the design of combinatorial libraries enriched in functionally deimmunized variants. After validating EpiSweep in a series of retrospective case studies providing comparisons to conventional approaches to T cell epitope deletion, we have experimentally demonstrated it to be highly effective in prospective application to deimmunization of a number of different therapeutic candidates. We conclude that our broadly applicable computational protein design algorithms guide the engineer towards the most promising deimmunized therapeutic candidates, and thereby have the potential to accelerate development of new protein drugs by shortening time frames and improving hit rates.

Key words Biologics, Therapeutic proteins, Computational protein design, Protein engineering, Immunogenicity, T cell epitope, Deimmunization, Combinatorial library, Pareto optimization

1 Introduction

Protein drugs are the most advanced tools in physicians' arsenal of therapeutic agents, and these complex molecular entities continue to improve outcomes for a range of familiar diseases as well as yield new treatment options for previously intractable illnesses [1].

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_20,

One critical barrier to the development and clinical translation of therapeutic proteins is their susceptibility to immune surveillance within the patient's body, a process fundamentally driven by molecular recognition of immunogenic peptide fragments known as T cell epitopes. Upon eliciting an immune response, therapeutic proteins may cause a range of clinical complications including rapid drug clearance, loss of functionality and efficacy, delayed infusion-like allergic reactions, more serious anaphylactic shock, and even induced autoimmunity [2, 3].

To mitigate protein immunogenicity, biomolecular engineers have previously sought to identify highly immunogenic T cell epitopes and delete them by mutagenic substitution of key amino acid residues [4–6]. However, experimental strategies for T cell epitope deletion are time- and labor-intensive, costly, and not universally successful. These limitations have led to the application of computational T cell epitope predictors as tools to accelerate the deimmunization process [7–9]. Yet prediction and mutagenic deletion of T cell epitopes is not sufficient for therapeutic protein deimmunization. Specifically, efficacious protein drugs require a folded, stable, and active structure; thus combinations of epitope-deleting mutations must be selected for compatibility with each other and with the native protein architecture and function. Functional deimmunizing mutations can be selected purely experimentally [10], or guided by analysis of homolog sequences [11–14] and structural energies [15–18].

To create the next generation of computational tools for therapeutic protein deimmunization, we have integrated computational T cell epitope prediction with computational analysis of the structural and functional consequences of epitope-deleting mutations [11, 13, 18–21]. As opposed to serial application of T cell epitope predictors followed by bioinformatics-based or experimental mutation analysis, our protein design algorithms simultaneously optimize therapeutic candidates for both low immunogenicity and high-level stability and activity. Furthermore, they do so over an entire protein, considering the global implications of mutations on immunogenicity and function. Finally, by employing a powerful combinatorial optimization framework, our methods are guaranteed to generate globally optimal protein designs (with respect to the implemented predictors).

Here, we provide a step-by-step guide to the application of the EpiSweep suite of deimmunization algorithms, as introduced in our series of algorithmic papers [11, 13, 18, 19, 21] and prospectively applied in our series of experimental papers [12, 14, 22–25]. To assess immunogenicity, the software utilizes any pocket profile-based epitope predictor; we illustrate here with the publicly available ProPred matrices [26]. To gauge the structural and functional acceptability of epitope deleting mutations, it employs either sequence-based design or structure-based design, with score

functions in the form of one- and two-body potentials based either on amino acid statistics or structural energy evaluations. To generate designs for experimental evaluation, it either optimizes single variants to be tested individually, or entire combinatorial libraries to be subjected to screening or selection techniques. The design process is based on a "sweep" approach that maps the Pareto optimal frontier of the design space, identifying those designs that best optimize epitope score and multi-body potentials simultaneously, with no other single design better than the selected ones in terms of both objectives.

2 Materials

2.1 Software

EpiSweep brings together a number of different protein analysis techniques within a powerful combinatorial optimization process. Many, and for some design problems, all of the steps may be performed on a standard desktop machine; for some larger design tasks, a computer cluster may be required to achieve satisfactory run times. Necessary software components are as follows:

1. The EpiSweep software.
   (a) Make a root directory (e.g., [*/home/user/episweep*]). Henceforth, directories and file names are in [*italics*]; customize as desired.
   (b) Download the EpiSweep suite (available by registration at http://www.cs.dartmouth.edu/~cbk/episweep/). EpiSweep Python modules are in [*episweep/episweep*], data files required to run EpiSweep are in [*episweep/data*], all preprocessing and postprocessing scripts described here are in [*episweep/bin*], other files including configuration files of third party programs are in [*episweep/misc*] and lysostaphin protein example files are in [*episweep/targets/lst_cwbd*].
2. Python 2.6 or higher, not Python 3.x (https://www.python.org).
3. IBM ILOG CPLEX API; freely available for academic research through the Academic initiative program (http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/). Ensure that the Python modules are installed and accessible from the Python interpreter, setting the PYTHONPATH environment variable if necessary.
4. Set environment variables so that the EpiSweep modules can be imported. Under Bash, add the following lines to the [*.bashrc*] file; apply analogous commands for other shells.

```
export EPISWEEP=/home/user/episweep
export PYTHONPATH=$PYTHONPATH:$EPISWEEP
export PATH=$PATH:$EPISWEEP/bin
```

5. Rotamer-based design and structure-based library design use third-party programs to generate potentials used by EpiSweep.
   (a) For extracting a rotamer potential:
      - OSPREY 2.x [27, 28] (http://www.cs.duke.edu/donaldlab/osprey.php).
      - Java JDK 1.6.x (or higher) (http://www.java.com).
      - Compile OSPREY and add the OSPREY binary files to the Java CLASSPATH in [*.bashrc*].

        export CLASSPATH=/home/user/OSPREY/bin:$CLASSPATH
      - Compile the OSPREY energy converter.

        $ javac $EPISWEEP/bin/eMatrixConverter.java
      - Add the OSPREY energy converter to the Java CLASSPATH in [*.bashrc*].

        export CLASSPATH=$EPISWEEP/bin:$CLASSPATH

   (b) For extracting a structure-based sequence potential:
      - CLEVER 1.0 [29, 30] (http://keatinglab.mit.edu).

        Add to [*.bashrc*] an environment variable referencing the CLEVER installation.

        export PATH=$PATH:/home/user/clever1.0/compiled
      - Rosetta suite (3.4 or higher) [31, 32] (https://www.rosettacommons.org/software), if using it to generate structures for CLEVER training.
6. The epitope analysis and postprocessing procedures demonstrated here are performed with R scripts and, when a structure is available, PyMOL [33]. The EpiSweep output files are in comma-separated values (CSV) format, so custom analysis procedures can readily be developed in any programming language or with spreadsheet software.
   (a) R 3.x or higher (http://www.r-project.org).
   (b) PyMOL 1.4 or higher (http://sourceforge.net/projects/pymol/).

2.2 Background Files

Background files are required by EpiSweep for preprocessing and performing epitope analysis on target specific files. The files can be downloaded from their respective sources and should be placed in [*episweep/data*]. The pre- and postprocessing scripts described below look for background files in this directory. For correct readability by EpiSweep, the files should follow specific formats described in Subheading 4.

1. **propred.csv**: T cell epitope score matrices, in CSV format (*see* **Note 1**); demonstrated here with ProPred [26] (http://www.imtech.res.in/raghava/propred).
2. **mccaldon.csv**: Amino acid background threshold frequencies for filtering mutations from the MSA, in CSV format (*see* **Note 2**). For the example presented here, the threshold values are obtained from the McCaldon scale [34].

2.3 Target Specific Files

Target specific data files include the sequence, structure, and other information for the protein of interest. For the example presented here, the files are stored in [*episweep/targets/lst_cwbd*]:

1. Target amino acid sequence, in FASTA format (*see* **Note 3**).
2. Target structure in the Protein Data Bank (PDB) format if structure-based design is to be performed (*see* **Note 4**).
3. Multiple sequence alignment (MSA) of target homologs, in FASTA format (*see* **Note 5**), if an amino acid potential is to be constructed or conservation is to be used to determined allowed mutations.
4. Any additional prior knowledge regarding mutational constraints for the target protein, in CSV format (*see* **Note 6**).

3 Methods

EpiSweep can employ sequence potentials ("sequence-based" design) or rotamer potentials ("rotamer-based" design) for designing deimmunized protein variants. Sequence potentials can be generated from sequence analysis (for example, using an MSA) or structural analysis (for example, using CLEVER [29, 30]), whereas rotamer potentials can be derived from rotameric structural analysis (for example, using OSPREY [27, 28]). EpiSweep can also construct a combinatorial library of variants ("library-based design") using multiple sets of amino acids at specific positions. These sets of amino acids are referred to as "tubes" and the potentials derived as tube potentials. Tube potentials are derived from averaged sequence potentials, which in turn can be either be MSA-based or structure-based. Once sequence, rotamer, or tube potentials are derived for a specific target, EpiSweep can design individual variants at specified mutational loads, or combinatorial libraries of specified sizes and numbers of mutated positions.

The methods are illustrated with case study application to a domain from a therapeutic protein that we have been developing [24, 25, 35]: the cell wall binding domain (CWBD) of *Staphylococcus simulans* lysostaphin (LST). Lysostaphin is a potent anti-staphylococcal enzyme, effective even against drug-resistant *S. aureus* strains [36]. It is a two-domain enzyme with a cell wall

binding domain that targets the bacterial peptidoglycan and a catalytic domain that cleaves the cell wall. Unfortunately, due to its bacterial origin, its immunogenicity has kept it out of clinical application.

Using background and target specific files, the following steps design deimmunized protein variants using EpiSweep (Fig. 1). First, create design specifications for the type of design and experimental construction to be performed—sequence-based, rotamer-based, or library-based. The design specification indicates what to consider at each position (allowed amino acids, rotamers, or tubes), which pairs of positions to score, and the scoring potentials to use. It also specifies parameters controlling the EpiSweep design process—number of mutated sites, epitope scoring matrix, and so forth. The EpiSweep process generates sets ("frontiers") of Pareto optimal and (if desired) increasingly suboptimal designs, trading off epitope content for potential score. The resulting designs are analyzed with postprocessing scripts in order to help select those for experimental evaluation.

3.1 Design Specification

An EpiSweep design problem is specified in a CSV-format file that lists the input files (both background and target-specific), the mutational choices, the scoring potential, and parameters controlling the design process. Example design specification files are provided for each lysostaphin example within corresponding folders located at [*episweep/targets/*].

1. EpiSweep can design individual variants and protein libraries based on sequence, structure or rotamer information. This specification starts by specifying the basis for design (**sequence, rotamer,** or **tube**).

   design, rotamer

2. Target sequence in FASTA format (*see* **Note 3**):

   wt_seq, lst_cwbd.fasta

3. Target structure in PDB format, if available (*see* **Note 4**):

   structure, lst_cwbd.pdb

4. MSA of homologs in FASTA format, if available (*see* **Note 5**):

   msa, lst_cwbd_MSA.fasta

5. Target-specific mutational constraints file, if available (*see* **Note 6**):

   mut_constraint, lst_cwbd_mut_constraints.csv

6. Epitope score matrix file (background file, in [*episweep/data*]; *see* **Note 1**) and threshold value. In this example, the threshold value of the scoring matrix is top 5%.

   epitope_score, propred.csv
   epitope_threshold, 5

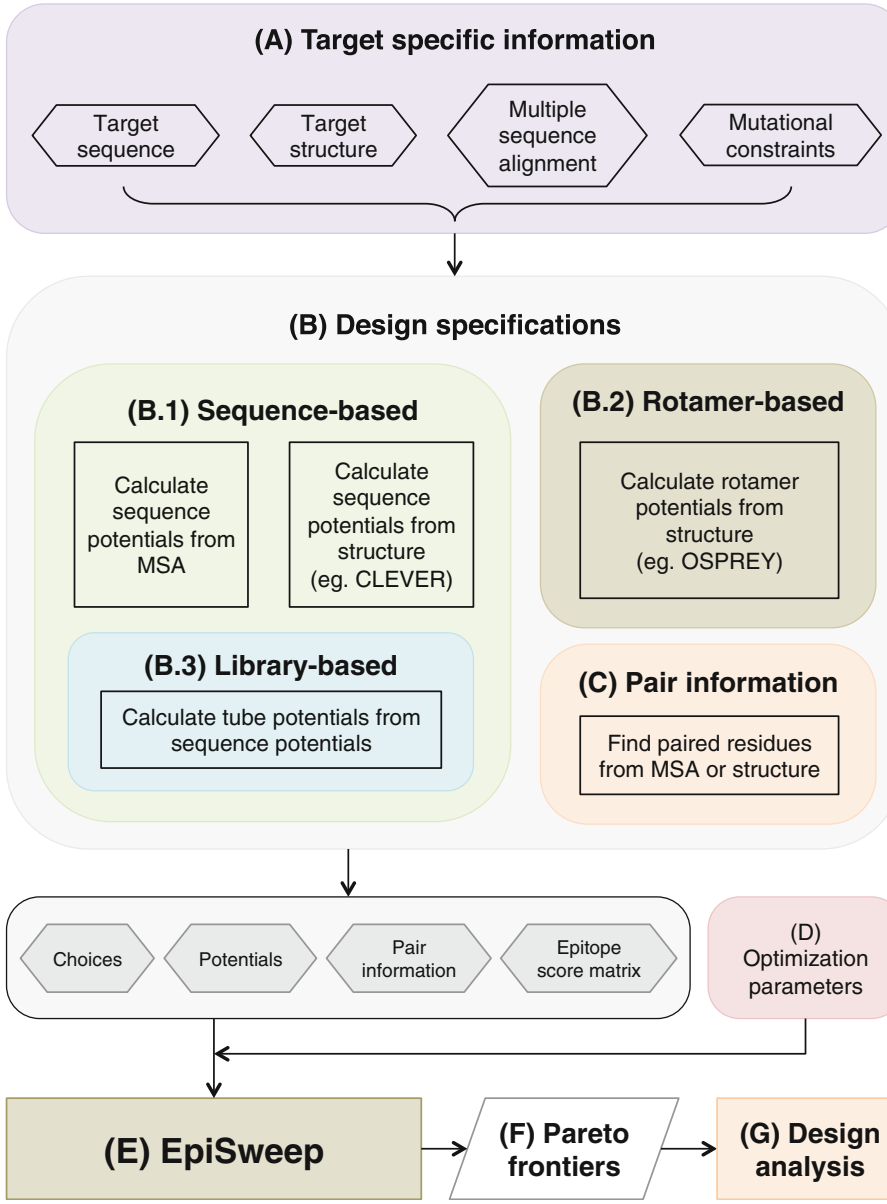


Fig. 1 Overview of the EpiSweep design strategy. (**a**) Collect target-specific information, including protein sequence, structure, homologs and knowledge-based mutational constraints. (**b–d**) Specify the details of the design problem. (**b**) Identify design choices and scoring potentials according to design strategy: (**b.1**) Sequence, (**b.2**) Rotamer, or (**b.3**) Library. (**c**) Restrict pair potentials to positions deemed worthy of scoring, according to sequence/structure analysis. (**d**) Provide other optimization parameters controlling mutational load, epitope scoring, etc. (**e**) Apply EpiSweep to design Pareto frontiers trading off epitope score and potential score. (**g**) Analyze designs to select those worthy of experimental construction and evaluation

7. Position-specific design choices (*see* **Note 7**):

   choices, lst_cwbd_rot_choices.csv

8. Position pairs to evaluate in CSV format (*see* **Note 8**):

   pairs, lst_cwbd_pairs.csv

9. Scoring potential, in either CSV format for sequence potential and tube potential (*see* **Note 9**) or binary format for rotamer potential (*see* **Note 10**):

   (a) For sequence potential:

   seq_potential, lst_cwbd_seq_pot.csv

   (b) For rotamer potential:

   rot_potential, lst_cwbd_rot_pot.dat

   (c) For tube potential, which also needs the originating sequence potential:

   seq_potential, lst_cwbd_seq_pot.csv
   tube_potential, lst_cwbd_tube_pot.csv

10. Optimization parameters. For all design problems, give the number of optimal/suboptimal curves (default optimal curves = 1) and the mutational load.

    num_curves, 5
    mut_load, 8

    In this example, the number of curves to be generated is 5 (one Pareto optimal and four suboptimal) and the number of allowed mutations (mutated positions for library design) is 9. The mutational load can also be given as a range, in which case all loads in the range will be designed:

    mut_load, 4-8

    For library design, also specify the minimum and maximum number of variants.

    library_size, 10000-20000

11. Finally, the filename for the EpiSweep output, in which the designs will be listed (*see* **Note 11**):

    design_output, lst_cwbd_plans.csv

3.2 Design Specification Generation

While all of the files in the design specification can be generated manually, EpiSweep provides a rich set of tools to help. An EpiSweep design specification generator is a simple Python script, in which a **Design** object is created and its methods are invoked in order to identify allowed mutations, generate scoring potentials, generate input files for external programs, etc. The following Python commands, from [*lst_cwbd_design_spec_gen.py*], illustrate the case study example.

1. Import EpiSweep library and initialize a Design object.

   from episweep import *
   design = Design()

2. Load the design specification CSV file.

   design.load_spec('design_spec.csv')

3. Allowed mutation choices (*see* **Note** 7 for format) can be extracted from the MSA of the target using the following filters.

   (a) Initialize the filter based on the wild type sequence and the MSA FASTA file.

   design.initialize_filter()

   (b) Filter the MSA to a subset of diverse, non-gappy sequences, with at least a specified identity to the wild type, at most another specified identity to others in the set, and at most a given fraction of gaps:

   design.msa_filter(low_thresh=0.30, high_thresh=0.95, fraction_gap_allowed=0.25)

   (c) Only keep amino acids that are sufficiently frequent in the (filtered) MSA relative to background frequencies. In this example, the McCaldon scale [34] of amino acid composition is used, according to the background file in [*episweep/data*].

   design.background_frequency_filter(input='mccaldon.csv')

   (d) Only keep amino acids that by themselves contribute to removing predicted epitopes. In this example, the epitope prediction allows only those mutations which can remove at least one predicted epitope.

   design.epitope_score_filter(min_epi_del=1)

   (e) Apply mutational constraints (the file specified in Subheading 3.1, **step 5**; *see* **Note 6**).

   design.apply_mutational_constraints()

4. Pairs whose interactions should be scored can be extracted either from the target MSA or the target protein structure (*see* **Note 8**).

   (a) Use MSA amino acid pair frequencies and χ^2 statistics [37].

   design.chisquare_pair_calculator()

   (b) Use contacts in the structure. Residues whose Cβ atom (Cα for glycine) distances are within a distance cutoff value are defined as a pair. In this example (and by default), the distance cutoff value is 8 Å.

   design.structure_pair_calculator(cutoff=8)

5. Construct the scoring potential file, which is in CSV format for sequence-based design (*see* **Note 9**) or in binary format for rotamer-based design (*see* **Note 10**). For library-based design, first construct a sequence potential, and then construct the tube potential from it.
   (a) MSA-based sequence potentials.
      - Generate a sequence potential from the target MSA.

        design.generate_seq_potential()
      - Finish editing the [*lst_cwbd_design_spec_gen.py*] file. Save and run the script.

        $ **python lst_cwbd_design_spec_gen.py**
      - The resulting potentials will be stored in [*lst_cwbd_seq_pot.csv*].

   (b) Structure-based sequence potentials, via CLEVER.
      - Prepare the CLEVER configuration file. A target name must be specified and the output file is [*lst_cwbd_CLEVER_design.dat*] (*see* **Note 12**), which contains a list of amino acid choices and pair information.

        design.CLEVER_design_file_maker(output='lst_cwbd_CLEVER_design.dat')
      - Finish editing the [*lst_cwbd_design_spec_gen.py*] file. Save and run the script to generate the [*lst_cwbd_CLEVER_design.dat*] file.

        $ **python lst_cwbd_design_spec_gen.py**
      - Generate random sequences by providing the design configuration file [*lst_cwbd_CLEVER_design.dat*] and using CLEVER sequence generator script [*GenSeqs*]. In this example, 6000 random sequences (*see* **Note 13**) are generated and saved in [*train6000.seq*]. This step also assigns zero energy values to each protein next to its sequence (*see* **Note 14**).

        $ GenSeqs -n 6000 -d lst_cwbd_CLEVER_design.dat -s train6000.seq
      - Use [*mutate_pdb.py*], providing the design spec file [*design_spec.csv*] and the random sequence file [*train6000.seq*]. This script creates a PDB file for each sequence, with file names in a sequential order (here *1.pdb, 2.pdb, . . . , 6000.pdb*).

        $ mutate_pdb.py –i design_spec.csv –s train6000.seq
      - Design structures and extract energies for the resulting sequences using any protein design program. For example, if employing Rosetta, use the **fixbb** program

to design each structure and **relax** to minimize them. The calculated energies will be used in CLEVER model training in the following steps.

- Enter calculated structure energy values into [*train6000.seq*] in the specified format (*see* **Note 14**), i.e., replace the zero values for each designed sequence in the initial file with their calculated energy values. Save this new file as [*train6000_energy.seq*]. An example file with this name and substituted energy values is provided for reference.
- Estimate amino acid energies using the CLEVER script [*CETrFile*], providing the sequence list with energy values and the configuration file. The output file [*lst_cwbd_CLEVER.log*] is the standard output of the script.

  $ CETrFile -d lst_cwbd_CLEVER_design.dat -s train6000_energy.seq -l lst_cwbd_CLEVER.log
- Convert the output file [*lst_cwbd_CLEVER.log*] to a structure-based sequence potential using the script [*clever_log_converter.py*], providing the log file. The output is a sequence potential file [*lst_cwbd_seq_pot.csv*] in CSV format (*see* **Note 9**).

  $ clever_log_converter.py –i lst_cwbd_CLEVER.log –o lst_cwbd_seq_pot.csv

(c) Rotamer potentials, via OSPREY.

- Prepare the OSPREY configuration files (*see* **Note 15**). [*KStar.cfg*] contains path information of OSPREY and rotamer libraries, and has to be separately prepared. An example file is provided in [*episweep/misc/osprey*]. The other two configuration files can be generated by the **design** object. The 'target' argument to the method is used to indicate final output file names. The following example generates [*lst_cwbd.DEE.cfg*], which contains amino acid choices at each position, and [*lst_cwbd.System.cfg*], which has structure information.

  design.OSPREY_config_maker(target='lst_cwbd')
- Finish editing the [*lst_cwbd_design_spec_gen.py*] file. Save and run the script.

  $ **python lst_cwbd_design_spec_gen.py**
- Run OSPREY, providing all the configuration files. For example, the following script runs OSPREY using 8 threads and 30 GB of memory. The standard output including rotamer choices after rotamer pruning will be saved in [*lst_cwbd_DEE.log*].

$ java -Xmx30000M KStar -t 8 -c KStar.cfg lst_cwbd.System.cfglst_cwbd.DEE.cfg doDEE > lst_cwbd_DEE.log

- Extract rotamer choices from the standard output file using the script [*extract_rotamer_choice.py*], providing the output log file [*lst_cwbd_DEE.log*]. The output file [*lst_cwbd_rotamer_choice.csv*] lists rotamer choices in EpiSweep CSV format (*see* **Note 7**).

  $ extract_rotamer_choice.py –i lst_cwbd_DEE.log –o lst_cwbd_rotamer_choice.csv

- OSPREY generates an energy matrix file [*lst_cwbd_0.dat*] and a reference energy file [*lst_cwbd.eref.dat*] in Java binary format. Convert the output energy matrix files to a binary file [*lst_cwbd_rot_pot.dat*] (*see* **Note 10**) using the converter [*eMatrix_converter.class*].

  $ java eMatrix_converter –i lst_cwbd_0.dat –r lst_cwbd.eref.dat –o lst_cwbd_rot_pot.dat

(d) Tube potentials.

- Generate a sequence potential via **step 5a** or **b**.
- Generate "tubes" [20] from the allowed mutations, using either degenerate oligonucleotides or combinatorial sets of amino acids. To specify tube generation parameters, *see* **Note 16**.

  design.generate_tube()

- Generate tube potentials from the sequence potential and the previously generated tubes.

  design.generate_tube_potential()

- Finish editing the [*lst_cwbd_design_spec_gen.py*] file. Save and run the script to generate and save tubes in [*lst_cwbd_tube_choices.csv*] and tube potentials in [*lst_cwbd_tube_pot.csv*].

  $ **python lst_cwbd_design_spec_gen.py**

3.3 EpiSweep Execution

EpiSweep execution only requires the design specification [*design_spec.csv*] file. The optimizer will set up an integer program to design variants or libraries according to the specification, and save the designs in the specified output file.

$ sweep.py design_spec.csv

3.4 Design Analysis

1. Assess overall predicted immunogenicity of the wild type.

   (a) Predict epitopes, using script [*epitopes.py*], providing the design spec file [*design_spec.csv*], design index number (0 for wild type) and output filename. The output file is saved in CSV format (*see* **Note 17**).

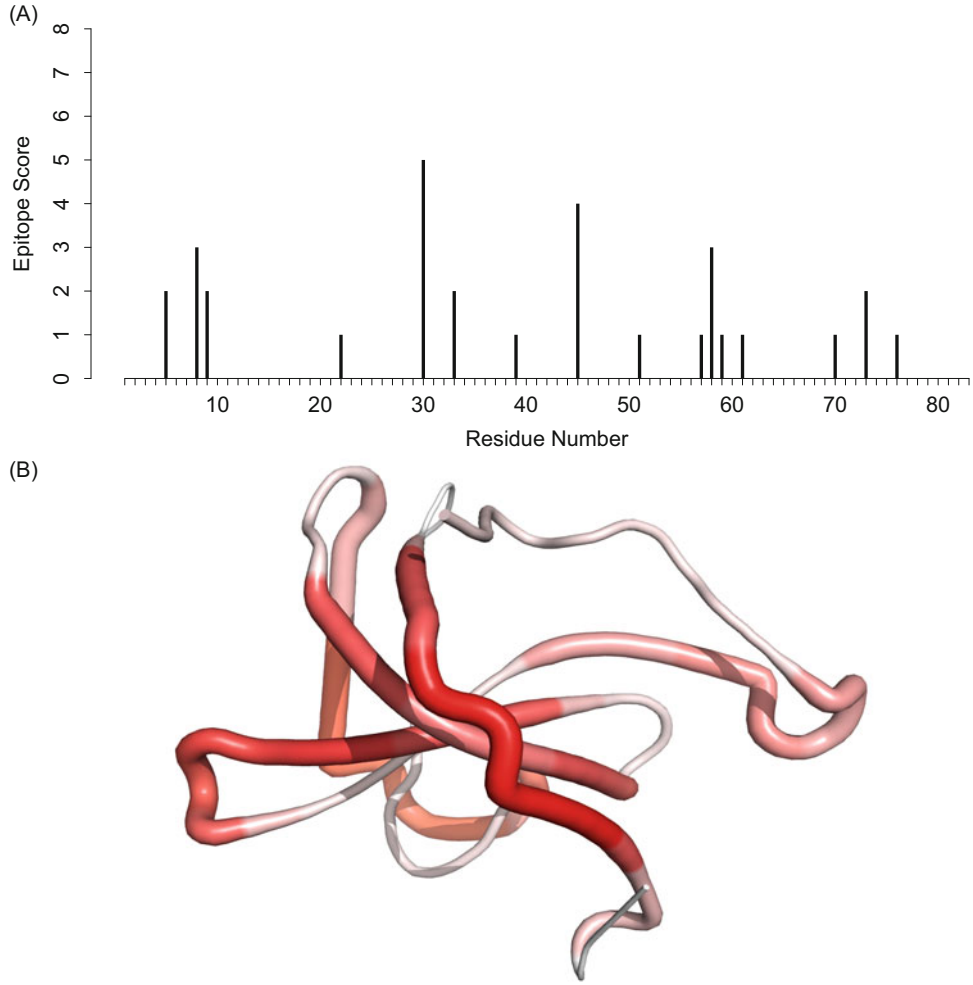


Fig. 2 Representation of epitope map: (**a**) sequence representation and (**b**) structure representation (rendered by PyMOL). The target protein shown here is the cell-wall binding domain of lysostaphin (PDB code: 4LXC chain A 403~493). In the sausage representation, the redder, the higher the epitope content

$ epitopes.py –i design_spec.csv –n 0 -o lst_cwbd_epitope.csv

(b) Plot the epitope "hits" (Fig. 2a), using script [*epitope_map.R*], providing the epitope prediction output file.

$ epitope_map.R –i lst_cwbd_epitope.csv –o lst_cwbd_epitope.pdf

(c) If a structure is available, visualize epitopes as "sausage" in PyMOL (Fig. 2b), using the script [*sausage_epitope.py*]; *see* **Note 18**.

PyMOL> from episweep import *
PyMOL> sausage(spec="design_spec.csv", palette="white red", overlap=True)

2. Plot the Pareto optimal frontiers (Fig. 3a for a rotamer-based individual design and Fig. 4a for a structure-based library design), using [*pareto_curve.R*] with the design spec file [*design_spec.csv*].

   $ pareto_curve.R –i design_spec.csv –o lst_cwbd_design_pareto.pdf

3. Predict epitopes for a variant of interest and compare to wild type.

   (a) Generate an epitope map [*lst_cwbd_var.csv*], using script [*epitopes.py*], providing the design spec file [*design_spec.csv*], design index number (design number 4 in this case) and output filename.

   $ epitopes.py –i design_spec.csv –n 4 -o lst_cwbd_var_epitope.csv

   (b) Plot differential epitope maps of individual designs (Fig. 3b). Use script [*epitope_deletion_map.R*], providing epitope prediction files of the wild type and the particular variant.

   $ epitope_deletion_map.R -w lst_cwbd_epitope.csv -v lst_cwbd_var_epitope.csv -o epitope_deletion_map.pdf

4. Enumerate and analyze the variants comprising a designed combinatorial library.

   (a) Extract a specific design from the Pareto frontier and enumerate a random subset or all designs (complete enumeration is potentially very time consuming), generating a CSV format file of individual variant designs [*lib_plan_enumerated.csv*] (*see* **Note 11**). In this example, library design index 10 is selected from the library Pareto frontier (Fig. 4a) and 5000 variants are randomly sampled.

   $ enumerate_library.py –i design_spec.csv –n 10 –e 5000 –o lib_plan_enumerated.csv

   (b) Score and plot the variants (Fig. 4b), using script [*plot_enumerated_lib.R*].

   $ plot_enumerated_lib.R –i lib_plan_enumerated.csv –o lib_plan_enumerated.pdf

4 Notes

1. The file for the epitope scoring matrices starts with a line indicating the name of MHC alleles to be evaluated, and then score entries followed by threshold values.

   Name

   Matrix

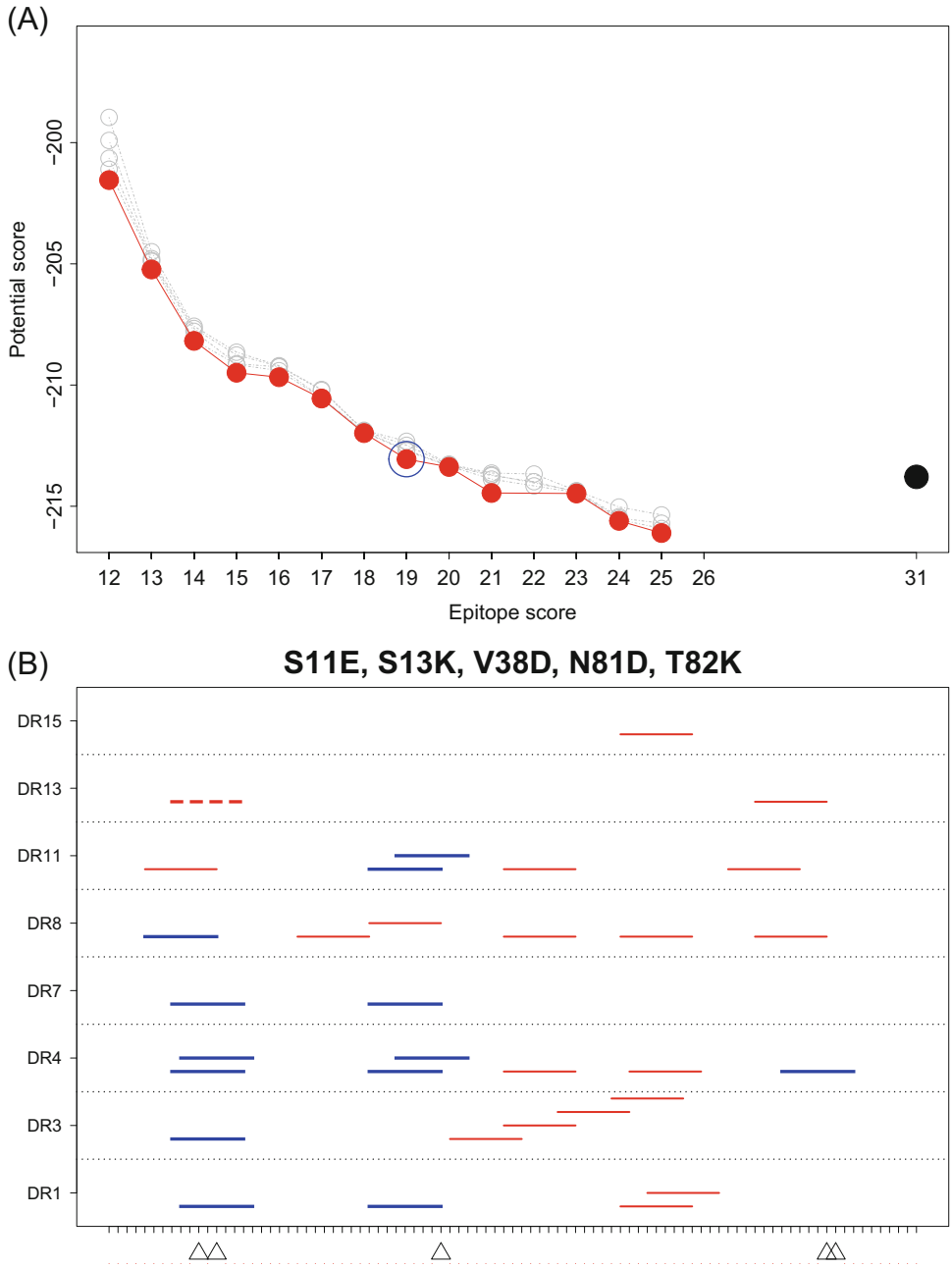


Fig. 3 Postprocessing results plots. Default scripts with default options were used. (**a**) Pareto optimal variants of the lysostaphin cell-wall binding domain from rotamer-based design (five mutations, one Pareto optimal and four suboptimal curves). The wild type is the *black solid circle*. The Pareto frontier is depicted with *red solid circles* and suboptimal designs are in *gray circles*. A particular design (in the *blue circle*, highlighted only for this figure) is further analyzed for epitope mapping. (**b**) An example epitope map of the design compared to the target wild type. *Red lines* are epitopes of the wild type and *blue lines* are deleted epitopes. The *broken red line* is the newly introduced epitope by the mutation. The mutated positions are marked at the *bottom*.

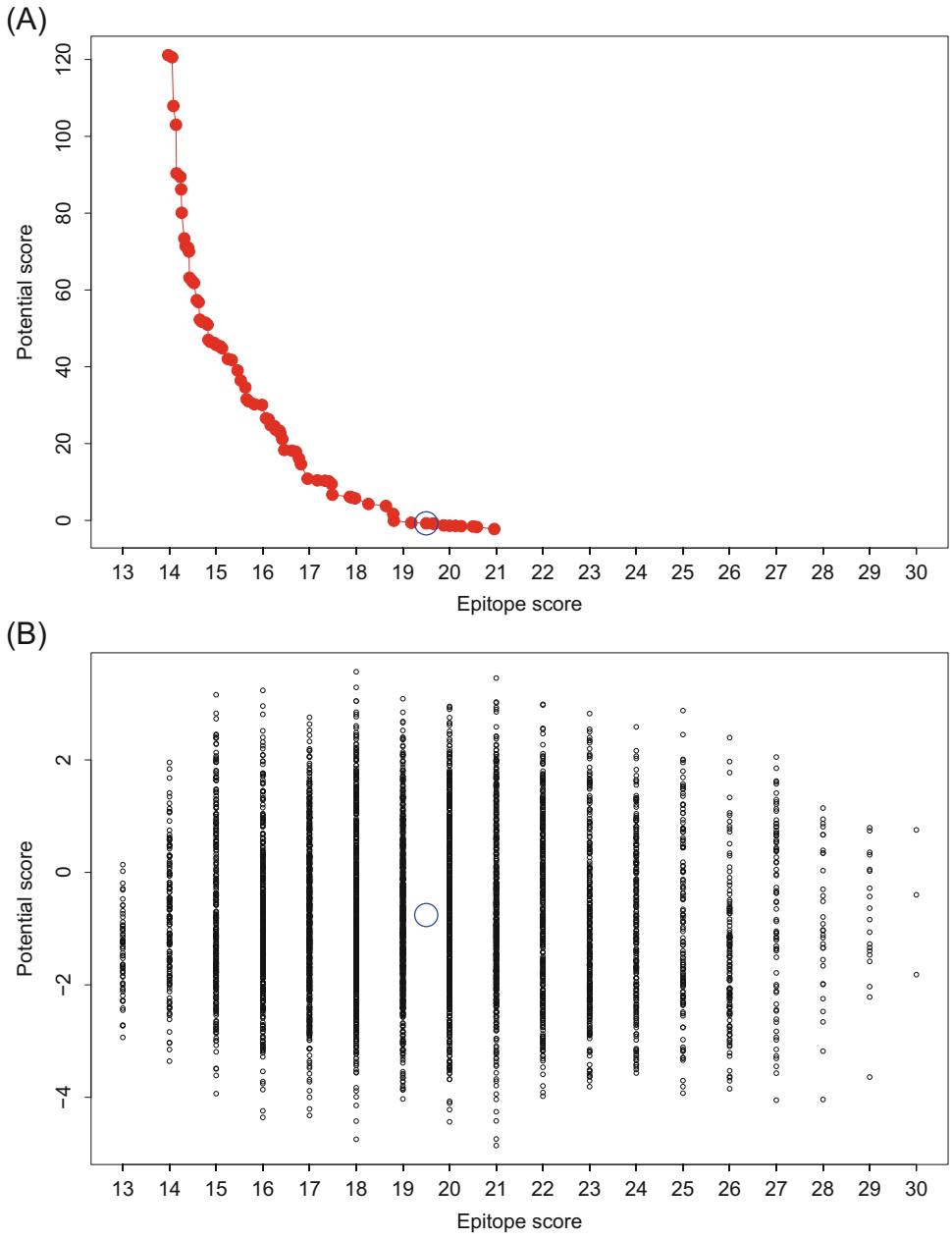


Fig. 4 Structure-based library designs. (**a**) Pareto curve, with each *red point* representing a library. The wild type is shown by a *black solid circle*. (**b**) Enumeration of the *blue circled* library from (**a**), with 5000 randomly selected variants illustrated by *small black solid circles*

Threshold values

where

(a) *Name* is for the user's reference and can be an arbitrary string (e.g., HLA-DRB1*0101).

(b) *Matrix* has 20 rows (amino acids in alphabetical order by 1-letter code), each with nine comma separated columns (peptide positions, P1 through P9) of position-specific floating point values.

(c) Given a query peptide, the position-specific values are summed and then the sum is compared to one of the *threshold values*. The peptide is considered a hit if the sum is higher than the assigned threshold value.

(d) Our previous studies have been performed at a 5% threshold and the epitope predictions shows good correlations with experimental IC50 values [14, 22, 23].

(e) Many studies have aimed to remove predicted T-cell epitopes of the eight most common human leukocyte antigen (HLA)-DR alleles (HLA-DR*0101, 0301, 0401, 0701, 0801, 1101, 1301, and 1501) [38].

2. The background amino acid frequency file is in two-column CSV format, with each row listing an amino acid and its threshold amino acid composition (%).

```
Ex: A,0.083
```

3. FASTA file format for the target is as follows:

```
>Sequence Name
ACDEFGHIKLMNPQRSTVWY
```

The first line is an identifier and the next lines contain a sequence of standard 20 amino acids without gaps (one letter code).

4. The structure should be in standard Protein Data Bank file format (http://www.wwpdb.org/documentation/file-format). Since the starting position of the sequence assumed to be 1, ensure that the structure follows the same numbering and has exactly the same residues.

5. The MSA contains multiple entries, each is in FASTA format (*see* **Note 3**).

```
>Sequence 1
ACDEFGHIKLMNPQRSTVWY
>Sequence 2
A-DEFGHIKLMNPQ-TVWY
```

All the sequences must be of the same length. The first sequence must be the target sequence (gaps "–" are allowed here). A set of homologs can be obtained from numerous

resources such as PSI-BLAST (http://blast.ncbi.nlm.nih.gov) and Pfam (http://pfam.xfam.org/).

6. A mutational constraints file specifies allowed/disallowed amino acids (one letter code) at each position. The file has four columns: position, exclusively allowed amino acids (only these are allowed for this position), allowed amino acids (listed amino acids are allowed but others are also possible) and disallowed amino acids (other amino acids are possible but the listed one are disallowed). For example, if tryptophan (W) is disallowed for the first position, only proline (P) is allowed for the second position, no restriction for the third position and asparagine (N) and phenylalanine (F) are allowed but not cysteine (C) for the tenth position, the file is written as follows:

```
1,,,W
2,P,,
3,,,
4,,NF,C
```

If a position is missing or left blank like position 3, it is considered unconstrained. If this file is not given, all the positions are considered unconstrained.

7. A choice file is a two or three-column CSV file.
   (a) For sequence and tube choices, this file has to have two columns, position and choice.

```
Ex (sequence choice): 1,A
Ex (tube choice, degenerate oligonucleotide in lower case): 1,rvt
Ex (tube choice, combinatorial amino acids in upper case): 1,AV
```

   (b) For rotamer choice, this file must have three columns: position, choice, and rotamer index.

```
Ex (rotamer choice): 1,A,2
```

8. The pair information is used to specify residue-residue interactions to be scored. The pair information file is in two-column CSV format: position 1, position 2.

```
Ex:
2,3
3,5
...
```

Note that the row is commutative (e.g., one can write a row as '3,5' or '5,3'), yet only one must be given.

9. Amino acid or tube score potential files can be prepared in the following CSV format:
   (a) For one-body score potential values: position, amino acid or tube, score

```
Ex: 2,G,-0.34
```

(b) For two-body score potential values: position 1, amino acid or tube 1, position 2, amino acid or tube 2, score
Ex: 5,D,30,K,-0.377637

10. A binary score potential file has the following binary format (Big-endian):
    (a) [4-byte unsigned integer] The number of residues, *n*.
        - The following elements are iterated for *n* times.
        - [4-byte unsigned integer] The number of mutations at this position, *p*.

          [1-byte character] Amino acid, *a*.

          [4 byte unsigned integer] The number of degenerate elements (e.g., rotamers), *r*.

    (b) [4-byte float] One-body score (p, a, r) values.
    (c) [4-byte float] Two-body score (p_1, a_1, r_1, p_2, a_2, r_2) values.
11. The result output file is in CSV format and columns are as follows:
    (a) Design index (sequential integer numbers, 0 for wild type).
    (b) Curve identification (0: wild type, 1: Pareto optimal and >1 for suboptimal designs).
    (c) Epitope score.
    (d) Potential score (the lower the better by convention).
    (e) Mutations made ("|" separated).
        - Ex. Empty for the wild type.
        - D13E (single mutation, D to E at position 13).
        - D13E|F45W (double mutation).
        - Y5rvt (a tube with the degenerate codon 'rvt', lower-case, encoding for amino acids N, T, S, D, A, and G).
        - Y5YV (a combinatorial tube YV encoding for combinatorial amino acids Y and V).
    (f) Sequence. For individual designs, a full sequence is shown. For library designs, '*' is present where tubes are incorporated.

```
Ex:
KTNKHGTLYKSE*GSFTPNTDIITRTTGPFTSMPQ*GV*KAGQT*HY-
DEVMKQ*GHVWVGYTGN*G*RIYLPV
KTNKHGTLYK*E*GSFTPNTDIITRTTGPFTSMPQSGV*KAGQT*HY-
DEVMKQ*GHVWVGYTGN*G*RIYLPV
KTNKHGTLYKSE*GSFTPNTDIITRT
TGPFTSMPQ*GV*KAGQT*HYDEVMKQ*GHVWVGYTGN*G*RIYLPV
```

12. CLEVER [29, 30] can be used to develop a sequence potential that represents structure-based energy terms, via a cluster expansion technique. CLEVER requires a list of possible amino acids at each position with the pair information. After training with energy data, CLEVER generates an output log file listing one-body and two-body scores, which can be converted into EpiSweep sequence potentials. Example configuration files are provided in [*episweep/misc/clever*] and a converter to EpiSweep format is in [*episweep/bin*]. The CLEVER configuration file is a plain text file that contains allowed amino acids at each position and pair information. The file has the following format:

```
#design_start
[Space separated allowed amino acids for all positions] Ex: 1 A C D
#design_end
#cluster_start
[A list of single positions line by line] Ex: 1
[A list of pairs line by line] Ex: 1 3
#cluster_end
```

13. The following equation [21] provides an approximate number n of sequences to generate for CLEVER model training, based on the number of positions l, choices per position m, and pairs p. Please note that more sequences results in a better-trained model.

$$n = l \times m + p \times m^2$$

14. The random sequence file is a plain text file each of which line has the following format:

```
[Energy value] [A space-separated sequence of amino acids]
Ex: 0.000 A C D E F G H I K L M N P Q R S T V W Y
```

Initially an energy value of 0.000 is assigned to each sequence. After calculating energy values for each sequence using modeling software, the zero entries need to be substituted for new energy values. For example, if the calculated energy value of the designed structure for the above sequence is -5.000, the substituted value would be:

```
Ex: -5.000 A C D E F G H I K L M N P Q R S T V W Y
```

15. OSPREY 2.0 or higher [27, 28] can be used to calculate structure-based rotamer energies. Note that OSPREY distinguishes protonation of histidine (HIP: double protonated histidine; positively charged, HID: delta protonated, HIE: epsilon protonated). The PDB structure file must contain all hydrogen atoms. Many programs can add hydrogen atoms to a PDB format structure, e.g., PyMol [33], AMBER [39], and TINKER [40]. The OSPREY rotamer-energy matrix needs to be converted to an appropriate format (*see* **Note 10**). There

are three configuration files required to run OSPREY; example configuration files are provided in [*episweep/misc/osprey*], and the design generator can produce [*System.cfg*] and [*DEE.cfg*] for a design spec.

(a) **KStar.cfg**: contains the system path of OSPREY and the rotamer library.

(b) **System.cfg**: contains information of the protein (structure filename, protein length, etc.).

(c) **DEE.cfg**: specifies output file names and detailed parameters including force field and mutation choices generated from the filtering process.

16. Library parameters that can be specified for tube construction are discussed with respect to this example:

design.generate_tube(method='degenerate', restrict_tube_size=50, avoid_aa=['C', 'P'], avoid_stop_codon=True, junk_percent=0.34, junk_type='deimmunizing')

(a) Method: EpiSweep designs combinatorial libraries to be constructed by incorporating at specific positions either degenerate oligonucleotides (**method='degenerate'**) or combinations of specific mutations (**method='combinatorial'**). A "tube" considered at a position is the corresponding set of amino acids, either encoded by the degenerate oligonucleotide or listed as point mutations.

(b) Tube size restriction: The number of amino acids encoded within a tube can be restricted by a user specified value. Content in the tubes from the 'combinatorial' method should be restricted to a small number (e.g., 3) since possible combinations grow exponentially with increasing number of amino acid choices. Content in 'degenerate' tubes is restricted by the genetic code and thus the tube size restriction can be relaxed (e.g., 50).

(c) Amino acids restriction: A tube can be excluded if it encodes for undesired amino acids (e.g., 'C' or 'P').

(d) Stop codon allowance: A degenerate tube encoding for a stop codon can be eliminated.

(e) Junk percent: Degenerate tubes that encode for additional amino acids (due to degeneracy) beyond the desired ones are removed if there are too many undesired ones (called *junk*). For example, if the desired choices are {D,G,V} then a tube encoding for {D,E,G,V}, due to degeneracy in the genetic code, includes E as junk. The default fraction of junk allowed is 1/3; this can be changed using the junk_percent parameter. The junk within a degenerate tube can be further restricted so

that only 'deimmunizing' junk (i.e., deletes an epitope, as in **design.epitope_score_filter**) is allowed, or that 'nondeimmunizing' junk is also allowed.

17. The output file of epitope prediction is saved in CSV format and columns are as follows:
    (a) Start position.
    (b) Peptide (nonamer).
    (c) End position.
    (d) Hits (Binary, Allele specific).
    (e) Total hits.
18. The script runs inside the PyMOL command line interface. The main function of the script is '*sausage*', which takes three arguments.
    (a) Design spec: the EpiSweep design specification file (mandatory).
    (b) Palette: color set (optional). Default: "white red", i.e., the redder, the higher the epitope score.
    (c) Overlap: if this argument is false, only the first residue of each epitope is highlighted, colored by how many epitopes start at that frame; else, all residues in each epitope are highlighted, with each position colored by how many epitopes include it. Default: True.

Acknowledgments

This work was supported by NIH grant R01-GM-098977 to KEG and CBK. We also gratefully acknowledge computational resources provided by NSF grant CNS-1205521.

References

1. Aggarwal SR (2014) What's fueling the biotech engine-2012 to 2013. Nat Biotechnol 32:32–39
2. De Groot AS, Scott DW (2007) Immunogenicity of protein therapeutics. Trends Immunol 28(11):482–490
3. Schellekens H (2002) Bioequivalence and the immunogenicity of biopharmaceuticals. Nat Rev Drug Discov 1(6):457–462
4. Cizeau J, Grenkow DM, Brown JG, Entwistle J, MacDonald GC (2009) Engineering and biological characterization of VB6-845, an anti-EpCAM immunotoxin containing a T-cell epitope-depleted variant of the plant toxin bouganin. J Immunother 32(6):574–584
5. Harding FA, Liu AD, Stickler M, Razo OJ, Chin R, Faravashi N, Viola W, Graycar T, Yeung VP, Aehle W (2005) A β-lactamase with reduced immunogenicity for the targeted delivery of chemotherapeutics using antibody-directed enzyme prodrug therapy. Mol Cancer Ther 4(11):1791–1800
6. Warmerdam PA, Plaisance S, Vanderlick K, Vandervoort P, Brepoels K, Collen D, De Maeyer M (2002) Elimination of a human T-cell region in staphylokinase by T-cell screening and computer modeling. Thromb Haemost 87 (4):666–673
7. De Groot A, Knopp P, Martin W (2004) De-immunization of therapeutic proteins by T-cell epitope modification. Dev Biol 122:171–194

8. De Groot AS, Moise L (2007) Prediction of immunogenicity for therapeutic proteins: state of the art. Curr Opin Drug Discov Devel 10 (3):332
9. Moise L, Song C, Martin WD, Tassone R, De Groot AS, Scott DW (2012) Effect of HLA DR epitope de-immunization of Factor VIII in vitro and in vivo. Clin Immunol 142 (3):320–331
10. Cantor JR, Yoo TH, Dixit A, Iverson BL, Forsthuber TG, Georgiou G (2011) Therapeutic enzyme deimmunization by combinatorial T-cell epitope removal using neutral drift. Proc Natl Acad Sci 108(4):1272–1277
11. He L, Friedman AM, Bailey-Kellogg C (2012) A divide-and-conquer approach to determine the Pareto frontier for optimization of protein engineering experiments. Proteins 80 (3):790–806
12. Osipovitch DC, Parker AS, Makokha CD, Desrosiers J, Kett WC, Moise L, Bailey-Kellogg C, Griswold KE (2012) Design and analysis of immune-evading enzymes for ADEPT therapy. Protein Eng Des Sel 25(10):613–624
13. Parker AS, Zheng W, Griswold KE, Bailey-Kellogg C (2010) Optimization algorithms for functional deimmunization of therapeutic proteins. BMC Bioinformatics 11(1):180
14. Salvat RS, Parker AS, Choi Y, Bailey-Kellogg C, Griswold KE (2015) Mapping the pareto optimal design space for a functionally deimmunized biotherapeutic candidate. PLoS Comput Biol 11(1):e1003988
15. Choi Y, Griswold KE, Bailey-Kellogg C (2013) Structure-based redesign of proteins for minimal T-cell epitope content. J Comput Chem 34 (10):879–891
16. King C, Garza EN, Mazor R, Linehan JL, Pastan I, Pepper M, Baker D (2014) Removing T-cell epitopes with computational protein design. Proc Natl Acad Sci 111 (23):8577–8582
17. Mazor R, Eberle JA, Hu X, Vassall AN, Onda M, Beers R, Lee EC, Kreitman RJ, Lee B, Baker D (2014) Recombinant immunotoxin for cancer treatment with low immunogenicity by identification and silencing of human T-cell epitopes. Proc Natl Acad Sci 111 (23):8571–8576
18. Parker AS, Choi Y, Griswold KE, Bailey-Kellogg C (2013) Structure-guided deimmunization of therapeutic proteins. J Comput Biol 20(2):152–165
19. Parker AS, Griswold KE, Bailey-Kellogg C (2011) Optimization of therapeutic proteins to delete T-cell epitopes while maintaining beneficial residue interactions. J Bioinform Comput Biol 9(02):207–229
20. Parker AS, Griswold KE, Bailey-Kellogg C (2011) Optimization of combinatorial mutagenesis. J Comput Biol 18(11):1743–1756
21. Verma D, Grigoryan G, Bailey-Kellogg C (2015) Structure-based design of combinatorial mutagenesis libraries. Protein Sci 24 (5):895–908
22. Salvat RS, Choi Y, Bishop A, Bailey-Kellogg C, Griswold KE (2015) Protein deimmunization via structure-based design enables efficient epitope deletion at high mutational loads. Biotechnol Bioeng 71(24):4869–4880
23. Salvat RS, Parker AS, Guilliams A, Choi Y, Bailey-Kellogg C, Griswold KE (2014) Computationally driven deletion of broadly distributed T cell epitopes in a biotherapeutic candidate. Cell Mol Life Sci 71 (24):4869–4880
24. Blazanovic K, Zhao H, Choi Y, Li W, Salvat RS, Osopovitch DC, Fields J, Moise L, Berwin BL, Fiering SN, Bailey-Kellogg C, Griswold KE (2015) Structure-based design of lysostaphin yields potent and deimmunized anti-staphylococcal therapies. Mol Ther Methods Clin Dev 2:15021
25. Zhao H, Verma D, Li W, Choi Y, Fiering SN, Bailey-Kellogg C, Griswold KE (2015) Depletion of T cell epitopes in lysostaphin mitigates anti-drug antibody response and enhances antibacterial efficacy in vivo. Chem Biol 22 (5):629–639
26. Singh H, Raghava G (2001) ProPred: prediction of HLA-DR binding sites. Bioinformatics 17(12):1236–1237
27. Chen C-Y, Georgiev I, Anderson AC, Donald BR (2009) Computational structure-based redesign of enzyme activity. Proc Natl Acad Sci 106(10):3764–3769
28. Gainza P, Roberts KE, Donald BR (2012) Protein design using continuous rotamers. PLoS Comput Biol 8(1), e1002335
29. Grigoryan G, Reinke AW, Keating AE (2009) Design of protein-interaction specificity gives selective bZIP-binding peptides. Nature 458 (7240):859–864
30. Grigoryan G, Zhou F, Lustig SR, Ceder G, Morgan D, Keating AE (2006) Ultra-fast evaluation of protein energies directly from sequence. PLoS Comput Biol 2(6), e63
31. Rohl CA, Strauss CE, Misura KM, Baker D (2004) Protein structure prediction using Rosetta. Methods Enzymol 383:66–93
32. Simons KT, Kooperberg C, Huang E, Baker D (1997) Assembly of protein tertiary structures

from fragments with similar local sequences using simulated annealing and Bayesian scoring functions. J Mol Biol 268(1):209–225

33. Schrödinger. The PyMOL molecular graphics system.
34. McCaldon P, Argos P (1988) Oligopeptide biases in protein sequences and their use in predicting protein coding regions in nucleotide sequences. Proteins 4(2):99–122
35. Zhao H, Blazanovic K, Choi Y, Bailey-Kellogg C, Griswold KE (2014) Gene and protein sequence optimization for high-level production of fully active and aglycosylated lysostaphin in Pichia pastoris. Appl Environ Microbiol 80(9):2746–2753
36. Kokai-Kun JF (2012) 10 Lysostaphin: a silver bullet for staph. Antimicrobial Drug Discov 22:147
37. Larson SM, Di Nardo AA, Davidson AR (2000) Analysis of covariation in an SH3 domain sequence alignment: applications in tertiary contact prediction and the design of compensating hydrophobic core substitutions. J Mol Biol 303(3):433–446
38. Southwood S, Sidney J, Kondo A, del Guercio M-F, Appella E, Hoffman S, Kubo RT, Chesnut RW, Grey HM, Sette A (1998) Several common HLA-DR types share largely overlapping peptide binding repertoires. J Immunol 160(7):3363–3373
39. Weiner PK, Kollman PA (1981) AMBER: assisted model building with energy refinement. A general program for modeling molecules and their interactions. J Comput Chem 2 (3):287–303
40. Ponder JW, Richards FM (1987) An efficient newton-like method for molecular mechanics energy minimization of large molecules. J Comput Chem 8(7):1016–1024

Chapter 21

Computational Tools for Aiding Rational Antibody Design

Konrad Krawczyk, James Dunbar, and Charlotte M. Deane

Abstract

Antibodies are a group of proteins responsible for mediating immune reactions in vertebrates. They are able to bind a variety of structural motifs on noxious molecules tagging them for elimination from the organism. As a result of their versatile binding properties, antibodies are currently one of the most important classes of biopharmaceuticals. In this chapter, we discuss how knowledge-based computational methods can aid experimentalists in the development of potent antibodies. When using common experimental methods for antibody development, we often know the sequence of an antibody that binds to our molecule, antigen, of interest. We may also have a structure or model of the antigen. In these cases, computational methods can help by both modeling the antibody and identifying the antibody–antigen contact residues. This information can then play a key role in the rational design of more potent antibodies.

Key words Antibodies, Antibody modeling, Rational antibody design, Antibody–antigen interactions, Antibody VH–VL orientation, CDR loop modeling

1 Introduction

Antibodies are proteins instrumental in mediating adaptive immune responses in vertebrates. An organism has a rich repertoire of antibodies, recognizing different structural motifs, called antigens [1]. Antibodies can therefore be described as a structural scaffold that houses a binding site specific for a particular antigen [2] (*see* Fig. 1 for an overview of antibody structure). Upon antigen exposure in an organism, an immune response introduces mutations in antibodies at the antigen recognition site, the paratope, which aim to increase the specificity and affinity toward a structural motif on the antigen, the epitope. Biologically, antibodies recognize non-self molecules that they tag for removal from the organism. However, the binding versatility of antibodies means that they can be engineered against an arbitrary epitope, a property that has received a lot of attention from the pharmaceutical industry [3].

Development of antibody-based drugs has been an important driver of the biopharmaceutical industry in recent decades [4]. The exploitation of antibody properties has resulted in a large

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_21,

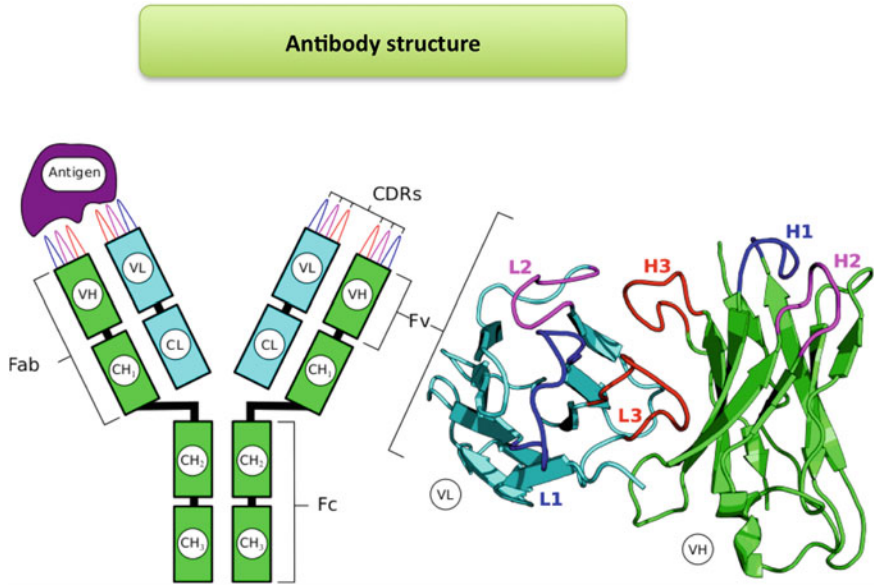


Fig. 1 Schematic of an antibody molecule (IgG isotype). The molecule consists of four polypeptide chains, two heavy (*green*) and two light (*cyan*). The variable domains, VH and VL, associate to form variable regions (Fv). The Fv, along with the first constant domains, CH1 and CL1, are known as antigen binding fragments (Fab). The remaining heavy constant domains associate to form the crystallizable or constant fragment (Fc). The complementarity determining regions (CDRs) mediate antigen binding. Three are located on each of the VH domain (H1, H2, and H3) and the VL domain (L1, L2, and L3). Between them they form the majority of the antigen binding site. The remaining portions of the VH and VL domains are called the framework (FR) regions

proportion of blockbuster drugs in recent years being antibody-derived [4–6]. Such development was made possible by our increasing understanding of antibody binding properties and the progress of experimental pipelines aiming to harness the underlying biology [7]. In order to develop an antibody for a target antigen, the most common protocol is to start with a library of candidate antibodies, all of which are potential binders. Those molecules from the set that bind to a target antigen are then mutated to give rise to a new generation of antibodies that are again tested for the quality of binding in an iterative manner. Such a procedure is very costly in time and material resources. Therefore, such experimental pipelines are now commonly supplemented by computational techniques [7].

The biological and structural properties of antibodies mean that they are well equipped for artificial design. Existence of a well defined scaffold which houses the binding site appears ideal for protein design. However, antibody–antigen complexes are asymmetric as opposed to general protein–protein binding which requires a customized approach, different from this of traditional protein design [8, 9]. Ideally, computational antibody design technology would, given an antigen sequence, be able to predict an

antibody sequence that binds to that antigen with high specificity and affinity [10]. Solving such a problem is beyond the reach of current computational techniques [7]. Here we consider instead the subproblems, which arise out of the common experimental methods for antibody development.

The goal of artificial antibody design is to produce molecules, which would bind specifically to a certain antigen with a very high affinity [11, 12]. The two most widely used methods to achieve this goal are the *humanization-based* [13] techniques and *phage-display* [14, 15], both summarized in Fig. 2. The former relies on raising antibodies in an animal, say mouse, and then engineering those molecules so that they do not elicit an immune response in the host species (human) but still bind to their respective antigen. The latter

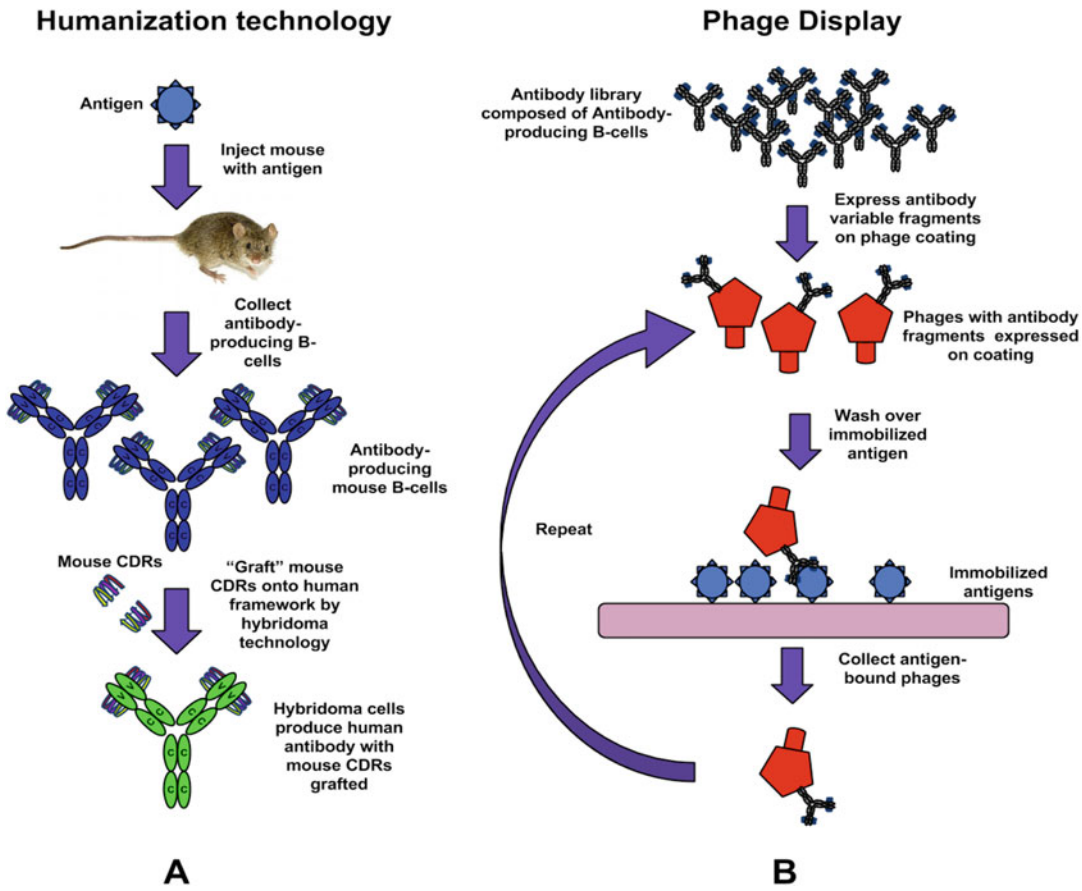


Fig. 2 Simplified descriptions of experimental antibody design pipelines. (**a**) Humanization technology. A mouse is injected with an antigen, prompting it to raise antibodies against it. The antibody producing B-cells are collected for in-vitro processing, where mouse CDRs are grafted onto a human framework. (**b**) Phage display. An appropriate antibody library is selected. The antibody variable fragments are expressed on the phage coating, and panned against immobilized antigens. Those antibodies that did not bind are washed off. The remaining, binding, antibodies are mutated and re-expressed on phage coating, starting another round of the process

method is a two-tier process where the two steps are antibody library construction and an iterative process of antibody mutation and good binder selection. In both pipelines, knowledge of specific residues that bestow structural or antigen-binding properties is crucial. Such information can be provided by currently available computational techniques for rational antibody design.

In this chapter, we describe computational methods facilitating rational antibody design starting from a sequence of the antibody to be engineered and a structure of its cognate antigen. The first step is to produce a model of the antibody in question. In the second step, the structural antibody information obtained in the first stage would be exploited to collect antibody–antigen contact information that could be used to guide rational antibody design. Here one predicts an epitope, paratope and performs antibody–antigen docking. The details of the constituent steps can be found in the following sections.

2 Knowledge-Based Modeling of Antibodies

Structural information is valuable for guiding rational antibody engineering decisions. Often an experimental structure is unavailable for each (or any) desired mutant of an antibody sequence. Computational analysis of the rapidly increasing number of known structures [16] has allowed the field to improve the accuracy with which an Fv can be modeled [17]. Such analysis also contributes to a developing structural toolbox for mutagenesis that gives insight as to how specific amino-acid changes may influence antibody structure.

Prediction of the antibody variable region can be divided into several steps (*see* Fig. 3):

- Annotation of the antibody sequence with a numbering scheme.
- Selection of templates for the VH and VL domain framework regions (FRs).
- Selection of non-CDRH3 loop templates.
- Prediction of the CDRH3 loop.
- Prediction or optimization of the VH–VL orientation.

The order in which these steps are performed varies with the antibody modeling protocol used. Many of the commercial and academic modeling protocols have been compared in the second antibody modeling assessment (AMAII) [17] and are described in corresponding publications.

Here, we describe the individual aspects listed above and explain how changes in these elements of structural variation could be introduced (or conserved) during a rational design process.

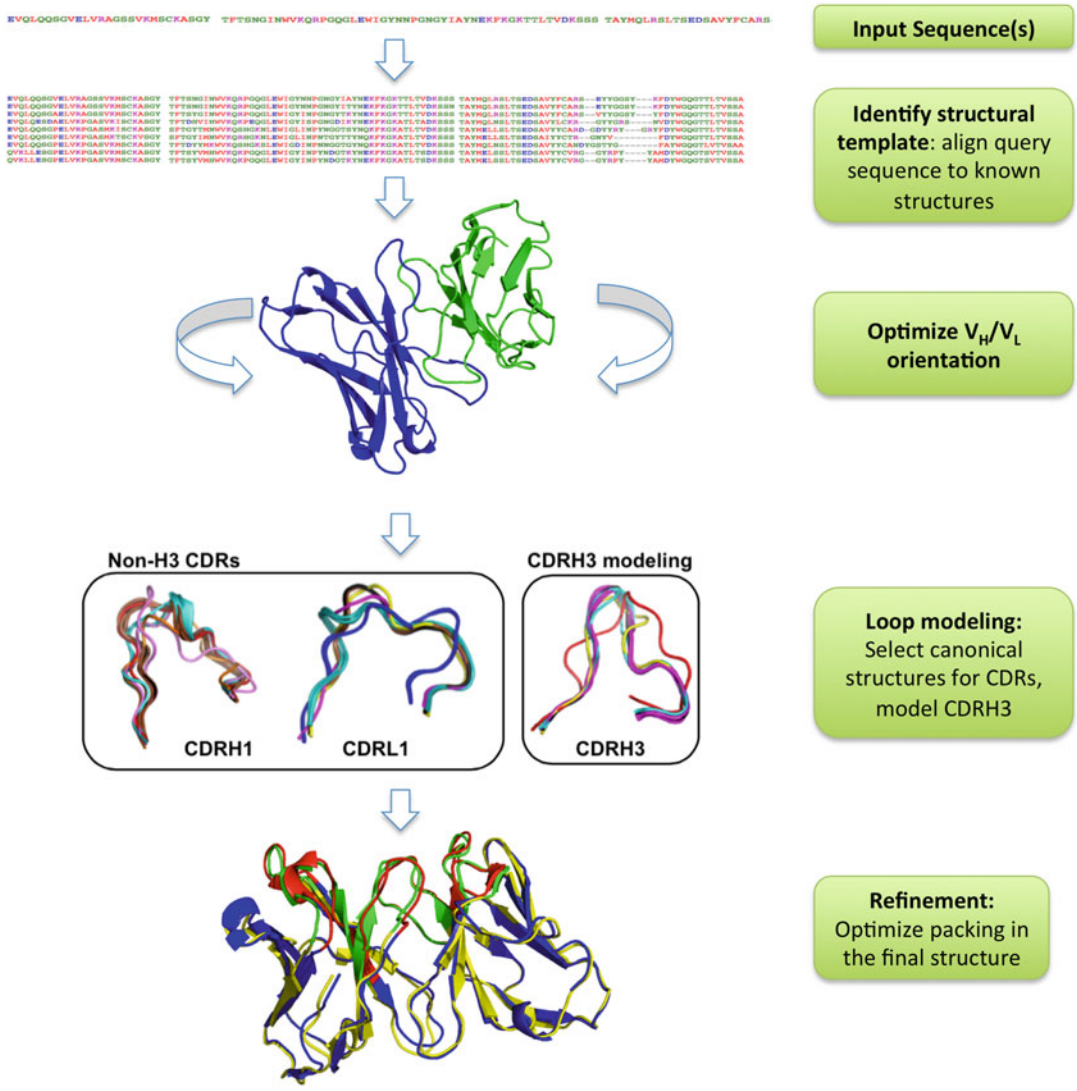


Fig. 3 Antibody modeling. The major steps in antibody modeling. Firstly an appropriate template is selected based on sequence similarity. This is followed by VH–VL orientation optimization and CDR loop modeling. Finally, the packing of the entire molecule is optimized

2.1 Annotation of the Antibody Sequence (Numbering Schemes)

Antibody numbering schemes are useful tools to annotate structurally equivalent residue positions within an antibody sequence. Thus, properties such as amino-acid preferences, structural influence of residues, and importance for antigen binding can be analyzed consistently. Multiple numbering schemes exist, all with their relative merits based on application. They vary in the nomenclature they use to label positions and the locations along the sequence at which they allow insertions and deletions. The features of the Kabat [18], Chothia [19], Enhanced Chothia [20], IMGT [21], and AHo [22] schemes are summarized in Table 1.

Table 1
Different antibody numbering schemes, their features and tools that can be used to apply them to a sequence

Scheme	Publicly available numbering tool	Scheme features
Kabat	Abnum (http://www.bioinf.org.uk/abs/abnum)	Based on analysis of VH and VL sequences
Chothia	Abnum	Places the VH CDR1 indel at the structurally correct position compared to Kabat
Enhanced Chothia	Abnum	Places framework indels at structurally correct positions compared to Chothia
IMGT	DomainGapAlign (http://www.imgt.org/3Dstructure-DB/cgi/DomainGapAlign.cgi)	Numbering is equivalent for VH and VL domains. CDR indels are labeled symmetrically about the middle of the CDR
Aho	PyIgClassify (http://dunbrack2.fccc.edu/PyIgClassify)	Numbering is equivalent for VH and VL domains. CDR indels are labeled symmetrically about the middle of the CDR. More possible positions than in IMGT

A numbering scheme can be applied to an antibody sequence using an alignment to a consensus sequence. The IMGT DomainGapAlign tool [23] provides an online service (*see* **Note 1**) to apply the IMGT scheme to nucleotide and amino-acid sequences. ABnum [20] is able to apply Kabat, Chothia, and Enhanced Chothia to Ig amino-acid sequences. It is available as an online service (*see* **Note 2**) and as a stand-alone program under license. PyIgClassify [24] includes an online tool (*see* **Note 3**) to annotate antibody sequences with the AHo numbering scheme. It is able to classify different structural regions of the antibody. Commercial antibody discovery packages also integrate the ability to apply common antibody numbering schemes.

Once a numbering scheme has been applied, the antibody sequence can be divided into Complementarity Determining Regions (CDRs) and Framework Regions (FRs) according to one of many different characterizations [18, 19, 25–27].

2.2 Selection of Templates for the VH and VL Domains

As the FRs of the VH and VL domains are relatively conserved in sequence and structure, predicting their structure is often a simple task. A high quality prediction may aim to model FRs to within at least 1 Å backbone root mean square deviation (RMSD). Typically, a template with greater than 80 % sequence identity over the FR will yield such accuracy for each domain. Indeed, almost all predictions made in AMAII [17] predicted VH and VL FRs to well within 1 Å backbone RMSD (means of 0.65 and 0.5 Å respectively).

Modeling protocols select VH and VL FR templates from databases of known and often curated structures. Publically

available tools to perform this step of the prediction procedure include IGBLAST (*see* **Note 4**) that identifies similar structures directly from the Protein Data Bank [28] and SAbDab's [16] template search tool (*see* **Note 5**) that ranks known structures by sequence identity over a specified region.

2.3 Selection of Non-CDRH3 Loop Templates

Despite their variability in sequence, a comparatively small set of different structural conformations are observed for five of the six CDR loops (L1, L2, L3, H1, and H2) [19, 27, 29–33]. These conformations are referred to as canonical classes. In many cases, the shape of a CDR loop can be recognized by the presence of certain amino-acids at particular structurally determining residue positions (SDRs) [30, 31]. Residues at other positions in the CDR loop can change to a number of different amino-acids without influencing the conformation. Rational engineering decisions to introduce mutations at CDR positions should consider their ability to influence the loop conformation.

It is not clear whether a number of possible CDR loop conformations have reached, or will reach, saturation. Al-Lazikani et al.'s 1997 study [19] found 25 possible conformations whilst a 2011 study by North et al. [27] found that the repertoire had increased to 72 conformations as the structural coverage of sequence space had grown. Recent methods for clustering antibody CDRs continuously monitor the redundant set of structures available from the PDB [16, 34]. The SAbDab interface (*see* **Note 5**) allows for the current antibody CDR space to be clustered at different cut-offs of structural similarity.

The most recent canonical classification of CDRs was performed by North et al. [27]. The authors' PyIgClassify tool [24] provides an online interface (*see* **Note 3**) and commercial licenses for databases that can be used to predict the canonical conformation of each AHo-North CDR loop from sequence. Such a method provides the ability to assess the structural influence of introducing mutations to CDR sequences. Prediction of the non-CDRH3 loops using canonical classification is commonly used by antibody modeling protocols (recently assessed in [17]).

An alternative approach to non-CDRH3 prediction is to treat it as any other loop modeling problem; that is, to do without the canonical CDR structure paradigm. Choi and Deane demonstrated that FREAD [35], a successful database loop prediction technique, is able to produce accurate predictions of CDR structures without explicit consideration of CDR classification [36]. FREAD is available as an online tool (*see* **Note 6**) with the option to use either immunoglobulin, membrane protein, or a general database to make predictions.

2.4 Prediction of the CDRH3 Loop

Prediction of the CDRH3 loop is more challenging than the other five CDRs due to larger variation in both its length and sequence. As the loop often plays a major role in antigen association, accurate CDRH3 modeling is a crucial step in structure-based rational engineering. As with general loop prediction methods there are two approaches that are generally taken: template/database methods or ab initio methods.

Although canonical structures do not exist for CDRH3 in the same sense as for the other CDRs, a degree of structural classification is observed. For example, CDRH3 structures can be classified as bulged or non-bulged in the torso region of the loop [27, 37, 38]. Rules primarily based around the presence of asparagine at Chothia position 101 show some success at guiding the choice of CDRH3 conformation [37, 38]. Such sequence-based rules are used in modeling protocols such as PIGs (*see* **Note** 7) [39] to filter a structure database for potential templates.

In comparison, the FREAD algorithm uses both sequence- and environment-specific information to select loop decoys. It is able to make accurate CDRH3 predictions [36]. However, FREAD, like other template-based methods, may not always provide a prediction for a given CDRH3 as no template exists.

Given that reliable templates are often unavailable, ab initio prediction of CDRH3 is required. Such prediction methods must be able to generate good loop decoys and to discriminate them from decoys that are less good. For example, the Kotai Antibody Builder [40] uses Spanner to build loop decoys and ranks them with the OSCAR energy function. Rosetta Antibody [51, 69, 70] uses a Monte-Carlo-based procedure followed by Cyclic Coordinate Descent (CCD) to build loops. Here, decoys are considered better if they minimize the Rosetta Energy Function.

CDRH3 prediction remains the biggest challenge for accurate antibody modeling. Better modeling accuracy may be achieved in the future by prediction methods that use a hybrid between template-based and ab initio approaches.

2.5 Prediction or Optimization of the VH–VL Orientation

In addition to CDR diversity, significant quaternary structural variation is found between antibody Fvs [16], primarily differences in orientation between the VH and VL domains. The orientation between VH and VL affects the relative placement of binding residues and thus the shape of the paratope. Changes in antigen affinity have been attributed to rearrangements in the VH–VL orientation [41–47]. Therefore, understanding what determines this property can both improve modeling accuracy and inform rational engineering decisions.

'Different approaches can be taken to predict the VH–VL orientation. The simplest is to copy the orientation from a known structure with high sequence similarity. Sequence similarity can be calculated using the whole Fv region or just at

positions that are known to frequently make contacts between VH and VL [39, 48, 49]. Alternatively, an energy function can be used to select the conformation from a set of known structures [50] or to iteratively optimize it during the prediction procedure [51].

The ABangle methodology [52] describes the VH–VL orientation in an absolute sense using a torsion angle, two tilt angles, two twist angles, and a distance. ABangle (*see* **Note 8**) can be used to compare individual structures or sets of structures to each other. Using this technique, different residues have been found to affect orientation in different directions [52].

Knowledge about which residues influence the orientation has improved the accuracy with which the VH–VL orientation can be predicted. Abhinandan and Martin [53] used a neural network to predict a torsion angle between VH and VL. Bujotzek et al. [54] used a random forest algorithm based on the identity of influential residues to predict the ABangle orientation measures. This method allows the geometry of Fv to be fully constructed.

3 Identifying Antibody–Antigen Contact Residues

Over the course of an immune response, antibodies are mutated to bind an antigen with high affinity and specificity. As a result of this one-sided accelerated evolution, an antibody interface is significantly different from that of non-immune protein–protein interfaces [55].

The goal of antibody–antigen contact residue prediction is to identify the residues on the antibody to mutate in order to increase the specificity and/or affinity against an arbitrary target (antigen). In 2007, Lippow et al. introduced the first successful approach, which achieved this [10]. Starting from a solved antibody–antigen complex, the authors introduced an exhaustive set of point mutations to the CDRs, evaluating each mutant using the CHARMM energy function. Several of the energetically favorable point-mutants were experimentally synthesized and some had higher binding affinities than the original antibody. A few of the point mutations were combined to generate double and triple mutants that achieved an even larger increase in binding affinity.

The method by Lippow et al. served primarily as a proof of concept as for any realistic applications, one cannot assume the existence of a solved antibody–antigen complex. There exist other computational design technologies, however their applicability remains to be tested in the lab [8, 9]. Nevertheless, advances in the last 10 years have increased our ability to tackle computational antibody design starting from a more realistic setup, which is mutating a sequence of an antibody, with respect to a known structure of the antigen. Computational techniques which can aid in tackling this problem can be divided into three categories:

Paratope prediction: predict the residues on the antibody which are in contact with the antigen.

Epitope prediction: predict the residues on the antigen which are in contact with the antibody.

Antibody–antigen docking: given a structure/model of an antibody and a structure/model of an antigen, aim to re-create the structural complex that they form.

We describe how the methods above can aid in rational antibody design in the sections that follow.

3.1 Paratope Prediction

Only a small number of mutations to an antibody in its binding site can lead to a radical change in specificity and affinity toward an antigen. Therefore, identifying paratope residues can greatly reduce the mutagenesis choices one would have to perform experimentally.

CDRs contain about 80 % of the paratope, thus CDR identification methods can be regarded as a form of an antibody binding site predictor [56]. These include the traditional CDR definitions such as Chothia [29], Kabat [18], IMGT [21] or Contact [25] as well as the more structurally informed Paratome (*see* **Note 9**) [56, 57]. Such annotations can be created using the Paratome online tool [57] or as described in Subheading 2.1.

Nevertheless, CDR regions still contain many residues that do not constitute part of a paratope (only around 15 residues out of the approximately 45 in the CDRs are paratope residues). Therefore from a mutagenesis point of view, it might be more beneficial to know fewer paratope residues but with greater confidence. Examples of paratope predictors which aim to perform such high-precision antibody-contact residue identification are proABC (*see* **Note 10**) [58] and Antibody i-Patch (*see* **Note 11**) [55].

The first method, proABC is a random-forest-based algorithm, which only requires sequence information at input. Antibody i-Patch uses both sequence and structural information. Antibodyi-Patch uses antibody-specific statistics for its predictions. In contrast to Paratome and the CDR definition methods which indicate the extent of the general binding region, Antibody i-Patch assigns a contact likelihood score to each residue, allowing the user to choose a cutoff so as to achieve higher precision or better coverage (*see* Fig. 4 for an example). By doing so, one can differentiate between higher and lower confidence predictions, that might provide a better guide for artificial antibody design.

Predictions from Antibody i-Patch can be used to guide mutagenesis or as constraints for other computational antibody design methodologies. It was demonstrated that the residues with higher Antibody i-Patch scores are more important energetically. Therefore, when engineering an antibody, one might first introduce mutations to the regions with high Antibody i-Patch scores. This

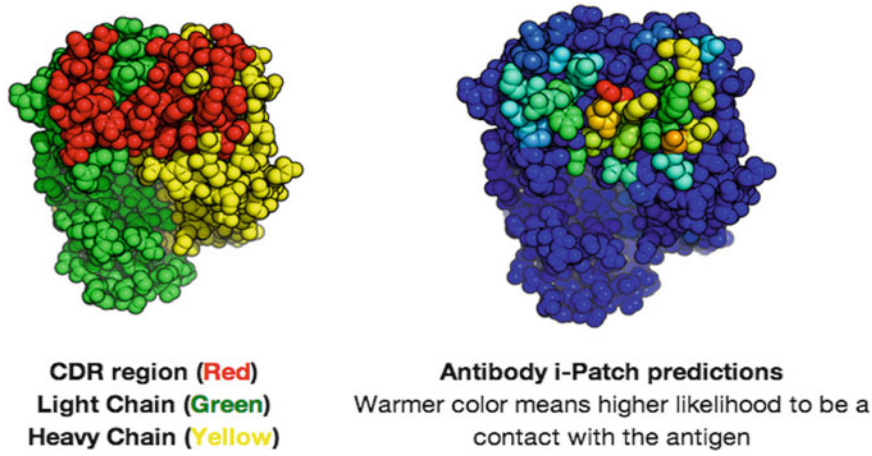


Fig. 4 Paratope prediction. *Left*: An example of a CDR annotation method. *Right*: An example annotation using the paratope prediction method Antibody i-Patch

might accelerate the process of increasing the specificity and affinity toward an antigen since it greatly reduces the combinatorial space, which would need to be explored by mutagenesis of all the residues in the CDRs.

Antibody i-Patch predictions of paratope residues successfully make use of the antibody–antigen specific statistics. However, the asymmetric nature of the antibody–antigen interaction made it impossible to make this methodology directly applicable to provide results for the related problem of epitope prediction. Furthermore, other paratope prediction methods, such as Paratome or CDR identification methods, are not directly informative of the epitope. Therefore, epitope predictors have developed into a separate field, using different methodologies to those of paratope prediction.

3.2 Epitope Prediction

Identifying epitopes can provide valuable insight into the pathogenicity of autoimmune diseases as well as characterization of immunogenic motifs [59, 60].

The majority of epitope prediction methods focus on the identification of immunogenic portions of antigens, attempting to define a structural motif capable of eliciting immune responses and thus, acting as an antibody target. In an effort to construct a map of known immunogenic motifs, databases such as the Conformational Epitope Database or the Immune Epitope Database have curated sequence and structural information related to epitopes from a variety of publicly available sources [61–63].

The majority of epitope prediction methods to date do not require any antibody information on input, operating on the assumption that there exist structural/sequential motifs that are inherently more immunogenic. However, it has been demonstrated that epitope residues are not distinguishable from the rest of the

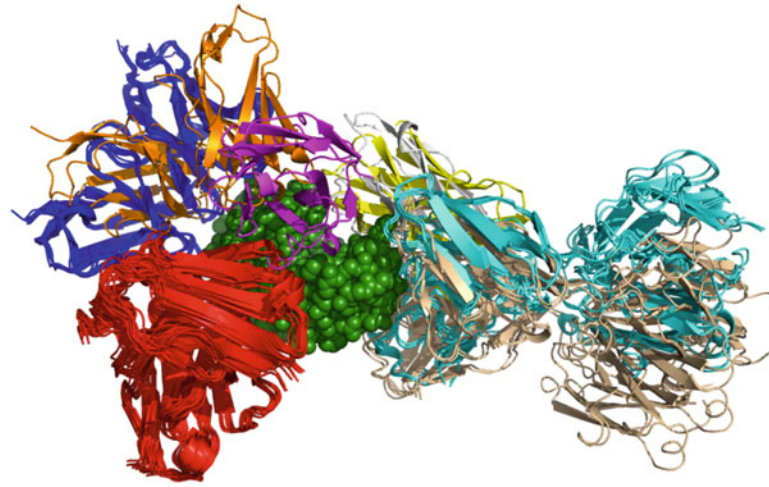


Fig. 5 Epitope prediction. Lysozyme and its binding antibodies. A structural overlay of many of the different antibodies, which bind to distinct sites on the hen egg-white lysozyme (in *green*). The many possible binding sites suggest that antibodies can bind to virtually any site on a target protein

protein surfaces [64]. This result suggests that there might be no particular structural motif capable of eliciting an immune response. Figure 5 shows many different antibody/lysozyme structures and how their binding sites cover almost the entire surface of lysozyme. This suggests that virtually any protein surface can form a portion of some epitope.

Recently, including antibody information in predicting epitopes has been shown to outperform methods, which ignore antibody information [65, 66]. A combined computational/experimental protocol demonstrated the utility of exploiting antibody information in distinguishing different epitopes (*see* **Note 12**) [65]. Another method, EpiPred, also demonstrated the utility of using antibody information without using experimental input, thus making it suitable for purely computational applications (*see* **Note 13**) [66]. EpiPred is a combination of geometric fitting and antibody-specific statistical potentials. It uses the structure of an antibody and the structure of the antigen as input. It outputs a ranked list of epitopes, deemed to be specific for the input antibody.

Epitope predictions obtained in this way can be used to inform antibody design directly. For instance, it might be the case that an antibody needs to bind in a specific spot on a therapeutic target (antigen), in order to prevent a pathogenic interaction from occurring. Scores of an antibody with respect to a certain epitope from antibody-specific predictors might indicate whether a given antibody is suitable for the destined binding site. On the other hand, epitope predictions might be used as information for more sophisticated computational antibody design methods, such as antibody–antigen docking.

3.3 Antibody–Antigen Docking

The paratope and epitope prediction methods described in the previous sections attempt to identify a subset of residues which form the antibody–antigen interface. However, they do not provide any information about the pairwise relationships between the residues on the antibody and the antigen. From an antibody engineering perspective such information might be very valuable since, it can directly indicate not only the positions on the antibody one needs to mutate but also the identity of the substitutions. This problem can be approached using antibody–antigen docking.

The field of antibody–antigen docking is a subset of the more general problem of protein–protein docking. Here, given unbound structures of two interacting proteins, one aims to recreate the complex between the two. There are two elements in docking—*decoy generation* and *decoy ordering*. In *decoy generation*, one produces a series of decoys—poses of one protein with respect to the other. In *decoy ordering*, the different poses are sorted so as to reflect the algorithm's score of which ones are the most likely to resemble the native complex.

Though antibody–antigen docking is a subset of the general protein docking problem, recent research has shown that antibody–antigen complexes are radically different from the general protein–protein complexes and as such require different methodologies [63, 64]. Using this concept antibody–antigen-specific docking methods have been developed which outperform more general methods [64, 65]. Such methods use antibody-specific information to constrain the decoy generation as well as antibody-specific methods to re-order decoys (*see* **Note 14**).

Since knowledge of the approximate binding interface increases the accuracy of pose generation, one should provide paratope and epitope information to a docking algorithm (*see* **Note 15**). Paratope prediction can be obtained by CDR identification methods or by more sophisticated methods such as Antibody i-Patch [55]. Epitope data are often available for common targets, via resources such as IEDB or relatively cheap experimental epitope mapping. If the epitope is not known, one needs to provide a prediction, which can be obtained using tools such as EpiPred [66].

Thus far there exists only one antibody–antigen-specific decoy generator, the antibody mode of ClusPro [67]. However, as the decoy generation algorithm one might also use a generic docking method as it appears that the crucial step for antibody–antigen docking is the reordering of decoys [55, 66]. The method should be fast so as to be able to provide results for many different variants of an antibody over the course of a virtual screening campaign. An example of such a method is the Fast-Fourier-Transform-based ZDOCK [68]. This decoy generation algorithm is capable of producing close-to-native antibody–antigen poses; however, as a result of non-antibody protein biases, they are not the top results in the ZDOCK output. Thus, the different poses of the antibody with

respect to the antigen need to be ordered using antibody–antigen-specific statistics. This can be achieved by an antibody-specific decoy reordering tool DockSorter [55]. This program removes the non-antibody protein biases that might be introduced by the decoy generation algorithm. In turn, DockSorter introduces antibody-specific scoring to produce a re-ordered list of decoys, bringing more native-like antibody–antigen decoys to the top of the results list.

4 Conclusions

The holy grail of computational antibody design is to produce an antibody sequence, which would bind an arbitrary antigen with high affinity and specificity. The current state of the art is far from being able to provide a solution to this problem, meaning that computational methods instead should be used to aid experimental pipelines. Nevertheless, as outlined in this chapter, computational methods are becoming more accurate and increasingly applicable to realistic antibody-design problems. We are now capable of creating antibody models with high accuracy. Such models can in turn be used as inputs to antibody contact prediction methods, which require structural information. Various statistics obtained from epitope predictors, paratope predictors, and antibody–antigen docking can indicate an initial set of mutagenesis choices one might wish to make whilst designing an antibody. Even though this is still a far-cry from being able to rapidly develop antibody therapeutics, the tools are already useful as an adjunct to experimental techniques.

5 Notes

1. The online IMGT DomainGapAlign service can be accessed via: http://www.imgt.org/3Dstructure-DB/cgi/DomainGapAlign.cgi.
2. The online antibody numbering service, Abnum, can be accessed via: http://www.bioinf.org.uk/abs/abnum.

   The service can be queried programmatically through an easy-to-use URL api.
3. The CDR clustering by the Dunbrack Lab can be accessed via:

   http://dunbrack2.fccc.edu/PyIgClassify.
4. The antibody sequence analysis suite IgBlast can be accessed via: http://www.ncbi.nlm.nih.gov/igblast.
5. Structural Antibody Database (SAbDab) template search tool, CDR clustering as well as other relevant antibody-related tools

can be accessed via: http://opig.stats.ox.ac.uk/webapps/sabdab.

Other notable online antibody resources are Abysis: http://www.bioinf.org.uk/abysis/ and DIGIT: http://circe.med.uniroma1.it/digit/help.php. Furthermore, Andrew Martin's Antibody resource pages provide a plethora of useful information on antibodies: http://www.bioinf.org.uk/abs/. In order to gather more detailed information on the antibody's cognate antigen, we suggest the Immune Epitope Database (http://www.iedb.org/) and the Conformational Epitope Database (http://immunet.cn/ced/). The IEDB resource has many analysis and prediction services, accessible via: http://tools.immuneepitope.org/main/.

6. Database-search-based loop modeling software FREAD can be accessed via an online service: http://opig.stats.ox.ac.uk/webapps/fread.

   User submits a structure without the loop coordinates, together with the loop sequence that should be modeled. If the service finds appropriate templates in its database, the gap in the submitted structure will be returned with the loop modeled. The method might fail for longer loops as those are rare and thus there are not enough suitable fragments in the database to model these.

7. Antibody modeling service PIGS is accessible online via: http://circe.med.uniroma1.it/pigs.

   One might also wish to employ one of the complementary antibody modeling services such as RosettaAntibody (http://rosie.rosettacommons.org/antibody) or Kotai Antibody Builder (http://kotaiab.org/).

8. Vh/Vl domain orientation analysis software Abangle is accessible online via: http://opig.stats.ox.ac.uk/webapps/abangle.

9. The Paratome online service can be accessed via: http://ofranservices.biu.ac.il/site/services/paratome/index.html.

10. The proABC antibody contact prediction service can be accessed via: http://circe.med.uniroma1.it/proABC/.

11. Antibody i-Patch is available through the SAbDab suite, as well as a binary distribution that can be downloaded from https://www.stats.ox.ac.uk/research/proteins/resources.

12. The epitope prediction from sequence can be accessed via: http://ofranservices.biu.ac.il/site/services/epitope/index.html.

13. The epitope prediction from structure using EpiPred can be performed by downloading the corresponding binary from: https://www.stats.ox.ac.uk/research/proteins/resources.

14. Currently, the Antibody mode of ClusPro is the only antibody-specific decoy generation method (SnugDock being for local docking [69]). The online service is accessible via: http://cluspro.bu.edu/login.php.
15. Most docking methods allow for constraining the exploration to only part of the binding partner. Notable examples here are ZDOCK and PatchDock. Constraining the search space of these rigid-body docking methods can provide a good coverage of the local conformations, which can be sorted by antibody-specific method such as DockSorter. This can be further refined using more computationally expensive flexible-docking method SnugDocky.

References

1. Robinson WH (2014) Sequencing the functional antibody repertoire-diagnostic and therapeutic discovery. Nat Rev Rheumatol 11:171–182. doi:10.1038/nrrheum.2014.220
2. Silverton EW, Navia MA, Davies DR (1977) Three-dimensional structure of an intact human immunoglobulin. Proc Natl Acad Sci U S A 74:5140–5144
3. Murad JP, Lin OA, Espinosa EV, Khasawneh FT (2012) Current and experimental antibody-based therapeutics: insights, breakthroughs, setbacks and future directions. Curr Mol Med 13:165–178
4. Reichert JM (2014) Antibodies to watch in 2014: mid-year update. MAbs 6:799–802. doi:10.4161/mabs.29282
5. Reichert JM (2013) Which are the antibodies to watch in 2013? MAbs 5:1–4. doi:10.4161/mabs.22976
6. Reichert JM (2010) Antibodies to watch in 2010. MAbs 2:84–100, doi: 10677 [pii]
7. Kuroda D, Shirai H, Jacobson MP, Nakamura H (2012) Computer-aided antibody design. Protein Eng Des Sel 25:507–521. doi:10.1093/protein/gzs024
8. Lapidoth GD, Baran D, Pszolla GM et al (2015) AbDesign: an algorithm for combinatorial backbone design guided by natural conformations and sequences. Proteins 83:1385–1406
9. Pantazes RJ, Maranas CD (2010) OptCDR: a general computational method for the design of antibody complementarity determining regions for targeted epitope binding. Protein Eng Des Sel 11:849–858
10. Lippow SM, Wittrup KD, Tidor B (2007) Computational design of antibody-affinity improvement beyond in vivo maturation. Nat Biotechnol 25:1171–1176
11. Kim SJ, Park Y, Hong HJ (2005) Antibody engineering for the development of therapeutic antibodies. Mol Cells 20:17–29
12. Martin ACR (2010) Protein sequence and structure analysis of antibody variable domains. In: Antibody engineering, vol 2. Springer, Berlin, pp 33–51
13. Safdari Y, Farajnia S, Asgharzadeh M, Khalili M (2013) Antibody humanization methods–a review and update. Biotechnol Genet Eng Rev 29:175–186. doi:10.1080/02648725.2013.801235
14. Carmen S, Jermutus L (2002) Concepts in antibody phage display. Brief Funct Genomic Proteomic 1:189–203. doi:10.1093/bfgp/1.2.189
15. Kretzschmar T, Von Rüden T (2002) Antibody discovery: phage display. Curr Opin Biotechnol 13:598–602. doi:10.1016/S0958-1669(02)00380-4
16. Dunbar J, Krawczyk K, Leem J et al (2013) SAbDab: the structural antibody database. Nucleic Acids Res 42(Database issue): D1140–D1146
17. Almagro JC, Teplyakov A, Luo J et al (2014) Second antibody modeling assessment (AMA-II). Proteins 82:1553–1562. doi:10.1002/prot.24567
18. Wu TT, Kabat EA (1970) An analysis of the sequences of the variable regions of Bence Jones proteins and myeloma light chains and their implications for antibody complementarity. J Exp Med 132:211–250
19. Al-Lazikani B, Lesk AM, Chothia C (1997) Standard conformations for the canonical structures of immunoglobulins. J Mol Biol 273:927–948. doi:10.1006/jmbi.1997.1354
20. Abhinandan KR, Martin ACR (2008) Analysis and improvements to Kabat and structurally

correct numbering of antibody variable domains. Mol Immunol 45:3832–3839

21. Lefranc MP (2011) IMGT unique numbering for the variable (V), constant (C), and groove (G) domains of IG, TR, MH, IgSF, and MhSF. Cold Spring Harb Protoc 6:633–642
22. Honegger A, Plückthun A (2001) Yet another numbering scheme for immunoglobulin variable domains: an automatic modeling and analysis tool. J Mol Biol 309:657–670. doi:10.1006/jmbi.2001.4662
23. Ehrenmann F, Kaas Q, Lefranc M (2010) IMGT/3Dstructure-DB and IMGT/DomainGapAlign: a database and a tool for immunoglobulins or antibodies, T cell receptors, MHC, IgSF and MhcSF. Nucleic Acids Res 38: D301–D307
24. Adolf-Bryfogle J, Xu Q, North B et al (2015) PyIgClassify: a database of antibody CDR structural classifications. Nucleic Acids Res 43:D432–D438
25. MacCallum RM, Martin AC, Thornton JM (1996) Antibody-antigen interactions: contact analysis and binding site topography. J Mol Biol 262:732–745. doi:10.1006/jmbi.1996.0548
26. Lefranc M, Pommié C, Ruiz M et al (2003) IMGT unique numbering for immunoglobulin and T cell receptor variable domains and Ig superfamily V-like domains. Dev Comp Immunol 27:55–77
27. North B, Lehmann A, Dunbrack RL Jr (2011) A new clustering of antibody CDR loop conformations. J Mol Biol 2:228–256
28. Berman HM, Westbrook J, Feng Z et al (2000) The protein data bank. Nucleic Acids Res 28:235–242
29. Chothia C, Lesk AM (1987) Canonical structures for the hypervariable regions of immunoglobulins. J Mol Biol 4:901–917
30. Chothia C, Lesk AM, Tramontano A et al (1989) Conformations of immunoglobulin hypervariable regions. Nature 342:877–883
31. Tramontano A, Chothia C, Lesk AM (1989) Structural determinants of the conformations of medium-sized loops in proteins. Proteins 6:382–394
32. Martin ACR (1996) Accessing the Kabat antibody sequence database by computer. Proteins 25:130–133
33. Oliva B, Bates PA, Querol E et al (1998) Automated classification of antibody complementarity determining region 3 of the heavy chain (H3) loops into canonical forms and its application to protein structure prediction. J Mol Biol 279:1193–1210
34. Nikoloudis D, Pitts JE, Street M, Ridgeway T (2014) A complete, multi-level conformational clustering of antibody complementarity-determining regions. PeerJ 2:e456
35. Choi Y, Deane CM (2010) FREAD revisited: accurate loop structure prediction using a database search algorithm. Proteins 78:1431–1440. doi:10.1002/prot.22658
36. Choi Y, Deane CM (2011) Predicting antibody complementarity determining region structures without classification. Mol Biosyst 7:3327–3334
37. Morea V, Tramontano A, Rustici M et al (1998) Conformations of the third hypervariable region in the VH domain of immunoglobulins. J Mol Biol 275:269–294
38. Kuroda D, Shirai H, Kobori M, Nakamura H (2008) Structural classification of CDR-H3 revisited: a lesson in antibody modeling. Proteins 73:608–620. doi:10.1002/prot.22087
39. Marcatili P, Rosi A, Tramontano A (2008) PIGS: automatic prediction of antibody structures. Bioinformatics 24:1953–1954
40. Shirai H, Ikeda K, Yamashita K et al (2014) High-resolution modeling of antibody structures by a combination of bioinformatics, expert knowledge, and molecular simulations. Proteins 82:1624–1635
41. Riechmann L, Clark M, Waldmann H et al (1988) Reshaping human antibodies for therapy. Nature 332:323–327
42. Foote J, Winter G (1992) Antibody framework residues affecting the conformation of the hypervariable loops. J Mol Biol 224:487–499
43. Chatellier J, Van Regenmortel MH, Vernet T, Altschuh D (1996) Functional mapping of conserved residues located at the VL and VH domain interface of a Fab. J Mol Biol 264:1–6. doi:10.1006/jmbi.1996.0618
44. Banfield MJ, King DJ, Mountain A, Brady RL (1997) VL:VH domain rotations in engineered antibodies: crystal structures of the Fab fragments from two murine antitumor antibodies and their engineered human constructs. Proteins 29:161–171
45. Khalifa MB, Weidenhaupt M, Choulier L et al (2000) Effects on interaction kinetics of mutations at the VH-VL interface of Fabs depend on the structural context. J Mol Recognit 13:127–139. doi:10.1002/1099-1352(200005/06)13:3<127::AID-JMR495>3.0.CO;2-9
46. Nakanishi T, Tsumoto K, Yokota A et al (2008) Critical contribution of VH–VL interaction to reshaping of an antibody: the case of humanization of anti-lysozyme antibody, HyHEL-10. Protein Sci 17:261–270. doi:10.1110/ps.073156708.Protein
47. Fera D, Schmidt AG, Haynes BF et al (2014) Affinity maturation in an HIV broadly

neutralizing B-cell lineage through reorientation of variable domains. Proc Natl Acad Sci U S A 111:10275–10280. doi:10.1073/pnas.1409954111

48. Whitelegg NR, Rees AR (2000) WAM: an improved algorithm for modelling antibodies on the WEB. Protein Eng 12:819–824
49. Ye J, Ma N, Madden TL, Ostell JM (2013) IgBLAST: an immunoglobulin variable domain sequence analysis tool. Nucleic Acids Res 41:W34–W40. doi:10.1093/nar/gkt382
50. Narayanan A, Sellers BD, Jacobson MP (2009) Energy-based analysis and prediction of the orientation between light-chain and heavy-chain antibody variable domains. J Mol Biol 388:941–953. doi:10.1016/j.jmb.2009.03.043
51. Sivasubramanian A, Sircar A, Chaudhury S, Gray JJ (2009) Toward high-resolution homology modeling of antibody Fv regions and application to antibody-antigen docking. Proteins 74:497–514
52. Dunbar J, Fuchs A, Shi J, Deane CM (2013) ABangle: characterising the VH-VL orientation in antibodies. Protein Eng Des Sel 26:611–620
53. Abhinandan KR, Martin ACR (2010) Analysis and prediction of VH/VL packing in antibodies. Protein Eng Des Sel 23:689–697
54. Bujotzek A, Dunbar J, Lipsmeier F et al (2015) Prediction of VH-VL domain orientation for antibody variable domain modeling. Proteins 83:681–695
55. Krawczyk K, Baker T, Shi J, Deane CM (2013) Antibody i-Patch prediction of the antibody binding site improves rigid local antibody-antigen docking. Protein Eng Des Sel 26:621–629. doi:10.1093/protein/gzt043
56. Kunik V, Peters B, Ofran Y (2012) Structural consensus among antibodies defines the antigen binding site. PLoS Comput Biol 8:e100238. doi:10.1371/journal.pcbi.1002388
57. Kunik V, Ashkenazi S, Ofran Y (2012) Paratome: an online tool for systematic identification of antigen-binding regions in antibodies based on sequence or structure. Nucleic Acids Res 40:W521–W524. doi:10.1093/nar/gks480
58. Olimpieri PP, Chailyan A, Tramontano A, Marcatili P (2013) Prediction of site-specific interactions in antibody-antigen complexes: the proABC method and server. Bioinformatics 29:2285–2291. doi:10.1093/bioinformatics/btt369
59. Kringelum JV, Lundegaard C, Lund O, Nielsen M (2012) Reliable B cell epitope predictions: impacts of method development and improved benchmarking. PLoS Comput Biol 8:e1002829
60. Ponomarenko JV, Bourne PE (2007) Antibody-protein interactions: benchmark datasets and prediction tools evaluation. BMC Struct Biol 7:64
61. Shirai H, Prades C, Vita R et al (2014) Antibody informatics for drug discovery. Biochim Biophys Acta 1844:2002–2015. doi:10.1016/j.bbapap.2014.07.006
62. Huang J, Honda W (2006) CED: a conformational epitope database. BMC Immunol 7:7
63. Kim Y, Ponomarenko J, Zhu Z et al (2012) Immune epitope database analysis resource. Nucleic Acids Res 40:W525–W530. doi:10.1093/nar/gks438
64. Kunik V, Ofran Y (2013) The indistinguishability of epitopes from protein surface is explained by the distinct binding preferences of each of the six antigen-binding loops. Protein Eng Des Sel 26:599–609
65. Sela-Culang I, Benhnia MREI, Matho MH et al (2014) Using a combined computational-experimental approach to predict antibody-specific B cell epitopes. Structure 22:646–657. doi:10.1016/j.str.2014.02.003
66. Krawczyk K, Liu X, Baker T et al (2014) Improving B-cell epitope prediction and its application to global antibody-antigen docking. Bioinformatics 30:2288–2294. doi:10.1093/bioinformatics/btu190
67. Brenke R, Hall DR, Chuang GY et al (2012) Application of asymmetric statistical potentials to antibody-protein docking. Bioinformatics 28:2608–2614. doi:10.1093/bioinformatics/bts493
68. Chen R, Li L, Weng Z (2003) ZDOCK: an initial-stage protein docking algorithm. Proteins 1:80–87
69. Sircar A, Gray JJ (2010) SnugDock: paratope structural optimization during antibody-antigen docking compensates for errors in antibody homology models. PLoS Comput Biol 6:e1000644. doi:10.1371/journal.pcbi.1000644
70. Sircar A, Kim ET, Gray JJ (2009) RosettaAntibody: antibody variable region homology modeling server. Nucleic Acids Res 37:W474–W479. doi:10.1093/nar/gkp387

Chapter 22

Computational Design of Membrane Curvature-Sensing Peptides

Armando Jerome de Jesus and Hang Yin

Abstract

Computer simulations have become an indispensable tool in studying molecular biological systems. The unmatched spatial and temporal resolution that it offers enables for microscopic-level views into the dynamics and mechanics of biological systems. Recent advances in hardware resources have also opened up to computer simulations the investigation of longer timescale biological processes and larger systems. The study of membrane proteins or peptides especially benefits from simulations due to difficulties related to crystallization of such proteins in a membrane environment. In this chapter, we outline the method of molecular dynamics and how it is applied to simulations that involve a peptide and lipid bilayers. In particular, the simulation of a membrane-curvature sensing peptide is examined, and ways of employing computational simulations to design such peptides are discussed.

Key words Molecular dynamics, Protein–lipid interactions, Membrane curvature, Curvature-sensing peptides, CHARMM

1 Introduction

Recently, numerous investigations have uncovered the active role of membrane curvature in controlling cellular organization and activity [1–6]. The picture that these studies paint is a departure from the traditional view of curvature as a passive geometric characteristic of the membrane. In cell signaling and trafficking, membrane shape has been revealed to play an important role [7–9]. Aside from this effect of curvature on protein function, it is also known that certain proteins generate membrane curvature or aggregate on curved membranes [9–15]. Examples of such proteins include: the C2B domain of synaptotagmin-I [16], the Golgi-associated ArfGAP1 lipid packing sensor (ALPS) [17], and the Bin-Amphiphysin-Rvs (BAR) domain of amphiphysin [18]. Cellular signaling functions ascribed to highly curved bilayer assemblies have also been implicated in health and disease [19–22]. Exosomes and microvesicles, assemblies that range in size from 30 to 1000 nm,

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_22,

have been implicated in numerous diseases such as cancer [23–28], HIV [29–31], and neurodegenerative diseases [32, 33]. Due to the important role that membrane curvature plays, curvature sensors such as a synaptotagmin-1-derived cyclic peptide and the effector domain of myristoylated alanine-rich protein kinase C (MARCKS-ED) [34, 35] have become important tools in targeting highly curved bilayer structures. The low molecular weight of these peptides makes it easier for their large scale production compared to large proteins. However, to design further improvements for these curvature sensors, the mechanics and kinetics of their action need to be studied and understood. Some proposals to explain curvature-sensing implicate the electrostatic interactions that occur between anionic lipid-enriched membranes and the concave surfaces of the protein or peptide [15, 36]. Another proposed mechanism involves the peptides that sense the surface defects arising from membrane curvature [17, 37–40], which usually entails certain residues of the peptide sensor to be inserted into the defects. Our previous work on MARCKS-ED [41] shows this possibility with the Phe residues acting, not only as the inserting residues but also to retain the peptide attachment to the bilayer.

Investigation of the mechanisms and dynamics of proteins is essential to studying how to generate new protein designs. For this purpose, a microscopic-level view into the movements and interactions of individual atoms is invaluable. In this regard, molecular dynamics (MD) simulations have become an indispensable method to explore the dynamics of biomolecular systems. Indeed, as a technique, MD has become a key tool in structural biology [42–46]. It enables an atomistic-level view of processes at a resolution, both spatially and temporally, that is currently inaccessible by experimental means. In our aforementioned study, MD simulations showed that the aromatic Phe residues, due to their hydrophobic character, were able to insert themselves into the bilayer interface within the first nanosecond of the simulation (temporal resolution). The simulations are also able to show that despite the presence of numerous Lys residues that can be visualized as interacting extensively with the bulk solvent, the Phe residues were able to stay buried to keep the MARCKS-ED peptide attached to the membrane (spatial resolution). This high-resolution level of understanding of the parameters influencing curvature sensing is essential to the design of peptides that are characterized by improved binding to highly curved structures.

In the context of using MARCKS-ED as a model membrane curvature sensor, the road to designing other similar curvature-sensing peptides can take the path of either replacing certain residues to increase the hold of the peptide on a curved bilayer or changing the chirality of the component residues to improve the resistance of the peptide to proteolytic processes [47–50].

2 Materials

To perform the simulations described in this chapter, software packages developed for molecular dynamics simulations are required. CHARMM [51], AMBER [52], GROMOS [53], and NAMD [54] are among the widely used programs that perform MD simulations. These programs also require topology and parameter files that are associated with the building of structures and the evaluation of the potential energy function.

The topology file contains the definitions for different atom types that comprise molecules (e.g., the atom type for a methyl carbon is different from a methylene carbon and both are different from an aromatic carbon). It also has the "structures" for different molecules typically used in biomolecular simulations such as those of individual amino acids and lipid molecules. These structures found in the topology files are essentially statements of atom types that comprise a molecule and statements of the connectivities of these atoms. The parameter file contains the force constants and equilibrium values needed to evaluate the different terms that comprise the force field equation (e.g., bond strengths, energies arising from dihedral angle configurations). The force field is the function used by MD programs to calculate the potential energy of the system. Further discussion of the force field is found below.

For solved protein structures, initial coordinates can be obtained from the Protein Data Bank. Lipid libraries offer initial structures for lipid molecules that will be needed in constructing a lipid bilayer. In addition, visualization programs are also essential. Examples of such programs are VMD [55], Rasmol [56], and Pymol [57]. In terms of hardware requirements, Unix machines are typically used.

3 Methods

3.1 The Method of Molecular Dynamics

The method of molecular dynamics treats each particle of the system as spheres which, based on their interactions with other particles in the system, move in accordance with Newtonian laws of motion [58–60]. These laws enable for a time-dependent picture of the system to be taken and thus, system kinetics to be observed. Knowledge of the forces acting on each particle of the system allow for the integration of the equations of motion which eventually results to a *trajectory* of the system to be computed and collected.

3.1.1 Overview of Running an MD Simulation

A typical MD run is summarized by the following steps [46, 59–61]:

1. Set up the system by initializing the positions and velocities of the atoms.

2. Calculate the forces on each atom.
3. Move each atom according to the calculated forces by integrating Newton's second law.
4. Advance simulation time.

Steps 2–4 are iterated until enough data has been accumulated. Each step is discussed in more detail below.

Initializing the Positions and Velocities of Atoms

For modeling proteins or peptides with solved structures, the initial configuration can be obtained from the Protein Data Bank. However, the absence of a structure deposited in the PDB does not necessarily mean that a particular sequence cannot be assigned initial coordinates. In some situations there are available means to fill in the missing information.

In cases where there is an absence of coordinates for a small number of atoms, internal coordinates can be used. Internal coordinates uses an atom's relation to other atoms, instead of using absolute Cartesian coordinates, to specify position [59].

Sometimes, knowledge of the secondary structure can also be used to initialize the atom positions. As an example, this method can be used to build Trp-Ala-Leu peptides (known as WALP [62] or WALP-like peptides [63]). These are model transmembrane peptides that have been widely used in studies of hydrophobic mismatch and are known to assume an alpha-helical secondary structure. Even in the absence of deposited structures for WALP or WALP-like peptides, initial positions can still be assigned to the component atoms by using a combination of data obtained from internal coordinates and the properties of an α-helix (such as the values for the backbone dihedral angles, Φ and ψ).

Initial atom positions are, however, not enough to begin an MD procedure. The initial velocities of the atoms need to be assigned as well. This is typically done by random assignment of velocities based on a distribution such as the Maxwell-Boltzmann distribution [59],

$$p(v_{ix}) = \left(\frac{m_i}{2\pi k_B T}\right)^{1/2} \exp\left[-\frac{1}{2}\frac{m_i v_{ix}^2}{k_B T}\right] \tag{1}$$

where the velocity of atom i, v_i, is assigned based on its mass, m_i, and the temperature of the system, T.

Calculating the Forces Acting on Each Particle

Once the position and velocity of each atom is known, the force acting on each one is calculated. The system configuration arising from the positions of the particles results to a potential energy, $\mathcal{V}$, for the system. The force acting on each particle can then be calculated as the gradient of the potential energy.

$$\mathcal{F} = -\nabla \mathcal{V}(\boldsymbol{R}) \tag{2}$$

In turn, the potential energy is calculated based on a set of functions that is known as a force field.

Moving the Particles

The positions of atoms are advanced at every time step by numerically integrating Newton's equations of motion. These algorithms are based on using a Taylor series expansion of atomic coordinates around time t. An example of a commonly used algorithm is called the velocity Verlet algorithm [64],

$$\begin{aligned} \boldsymbol{r}(t+\Delta t) &= \boldsymbol{r}(t) + \boldsymbol{v}(t)\Delta \mathrm{t} + \frac{1}{2m}\boldsymbol{F}(\boldsymbol{t})\Delta t^2 \\ \boldsymbol{v}(t+\Delta t) &= \boldsymbol{v}(t) + \frac{1}{2}\left[\frac{1}{m}\boldsymbol{F}(t) + \frac{1}{m}\boldsymbol{F}(t+\Delta t)\right]\Delta t \end{aligned} \tag{3}$$

which computes the atomic positions and velocities at the next time step ($\boldsymbol{r}(t+\Delta t)$ and $\boldsymbol{v}(t+\Delta t)$, respectively) in terms of the current positions, velocities and accelerations ($\boldsymbol{r}(t)$, $\boldsymbol{v}(t)$ and $\frac{1}{m}\boldsymbol{F}(t)$).

Advancing the Simulation Time

The errors in the positions and velocities obtained from the velocity Verlet algorithm (as well as from other similar, Taylor-series based algorithms) arise from the size of the time step, Δt. These errors are in the order $\mathcal{O}(\Delta t^4)$ for the positions and $\mathcal{O}(\Delta t^2)$ for the velocities [60]. Thus, a balancing act between lessening errors and achieving a reasonable sampling of phase space needs to be made. The size of Δt is limited by the fastest molecular motions which, in a typical MD simulation, involves the vibration of the bond connecting hydrogen to a heavier atom. The time step size sufficient for sampling these X–H bond vibrations is ~1 fs. To increase the time step, constraints can be applied to bonds to fix their lengths to equilibrium values. These constraints are typically applied to X–H bonds and their use enables the time step to be increased to 2 fs.

3.1.2 The Force Field

The force field is typically composed of terms that calculate the bonded and non-bonded interactions as shown in the CHARMM force field below [65]:

$$\begin{aligned} \mathcal{V}(\boldsymbol{R}) = & \sum_{\text{bonds}} K_b(b-b_0)^2 + \sum_{\text{angles}} K_\theta(\theta-\theta_0)^2 \\ & + \sum_{\text{dihedrals}} \sum_{n} K_\chi[1+\cos(n\chi-\delta)] + \sum_{\text{impropers}} K_{\text{imp}}(\omega-\omega_0)^2 \\ & + \sum_{\text{UB}} K_{\text{UB}}(S-S_0)^2 + \sum_{\text{nonbond}} \left\{ \epsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - 2\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right] + \frac{q_i q_j}{4\pi\epsilon_0 r_{ij}} \right\} \\ & + \sum_{\text{residues}} U_{\text{CMAP}}(\varphi,\psi) \end{aligned} \tag{4}$$

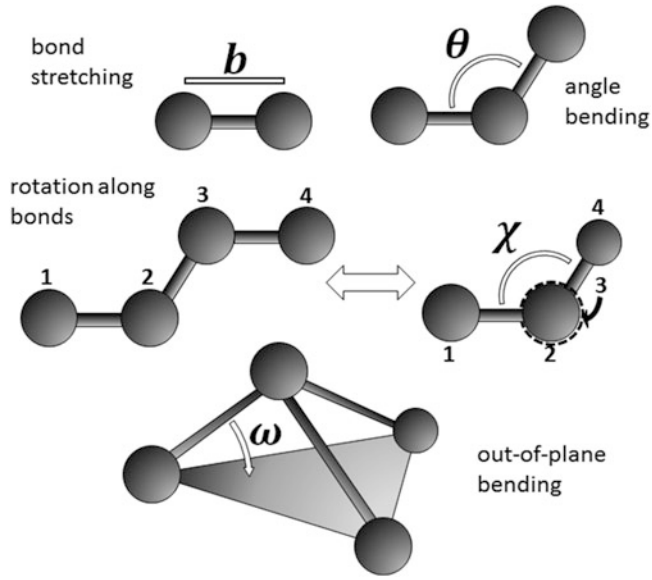


Fig. 1 Interactions among bonded atoms that are calculated by the bonded terms present in a typical force field. These bonded terms represent energy changes associated with stretching of bonds between two atoms, bending of angles defined by three atoms, rotation along a bond that connects a pair of bonded atoms and out-of-plane motions

In the CHARMM force field (Eq. 4), the bonded terms include bond, angle, dihedral, improper dihedral, and Urey–Bradley contributions. Most are harmonic in form with K_b, K_θ, K_{imp}, and K_{UB} representing the bond, angle, improper dihedral and Urey-Bradley force constants, respectively, and the zero-subscripted variables representing the equilibrium values. The Urey-Bradley term involves three bonded atoms A-B-C where S is the distance between A and B. The improper dihedral term controls the chirality of atoms A, B, and D connected to a central atom, C. The dihedral term is a sinusoidal expression where δ stands for the phase shift and n is the multiplicity that specifies the number of cosine terms that define the sinusoid. Figure 1 draws a schematic for these bonded terms. The non-bonded terms represent the Coulombic interactions between point charges q_i and q_j found at a distance r_{ij} from each other with ϵ_0 being the permittivity of free space. The Lennard-Jones term models the van der Waals interaction between two atoms i and j separated by a distance r_{ij}, where σ_{ij} equilibrium distance between the two atoms with an energy of ϵ_{ij}. The last term, the CMAP correction, corrects for small systematic errors for the dihedral backbone energy term.

3.1.3 *Maintaining Temperature and Pressure*

Absent any additional modifications, the process of solving Newton's equations of motion for a system of N particles in a volume V will produce a trajectory of states that have the same average energy, E, that is, a *microcanonical ensemble* will be generated [60].

However, experiments are typically performed under conditions of constant temperature and pressure. Thus, a computational method to maintain constant temperature or pressure conditions should be used. This would enable sampling from the *canonical* (constant number of particles, volume and temperature or *NVT*) and the *isothermal-isobaric* (constant number of particles, pressure and temperature or *NPT* constant pressure) ensembles. Thermostats used to maintain constant temperature conditions generally employ some means of scaling the velocities of the particles comprising the system. The most common algorithms used are based on the Nosé–Hoover extended system method [66, 67]. As the name suggests, this approach calls for extending the real system by a heat bath where the temperature of the system is maintained by means of heat exchanges with the heat reservoir. The heat bath, which has a fictitious coordinate, s, and a fictitious mass, Q, is treated as an additional degree of freedom in the energy function. Q is the coupling parameter that controls the heat exchange between the heat bath and the real system. A small value of Q results in rapid temperature fluctuations in the system.

In instances where constant pressure conditions are required, the volume of the system is allowed to fluctuate. An extended system approach analogous to that used for thermostats can also be employed where a piston with a mass and coordinate, both of which are fictitious, is coupled to the real system. The motion of this piston follows Langevin dynamics and is coupled to the system with a collision frequency parameter [68].

3.1.4 Periodic Boundary Conditions

Despite the advances in the size of systems being simulated, the number of particles comprising the model system will always be significantly smaller than those found in real-life samples. This system size limitation can create artificial boundary effects. A common approach to deal with the artifacts arising from artificial boundary effects is to use periodic boundary conditions. In applying this method, the system being simulated (considered as the primary cell) is replicated infinitely in three dimensions such that each atom in the primary cell has an image in the other replicated boxes. When an atom leaves the primary cell along one direction, its image atom from the opposite side of the central box will enter. Figure 2 presents a schematic for how this process, called *image centering*, is performed.

3.1.5 Treatment of Long-Ranged Forces

During the performance of MD simulations, the bulk of the computational expense is allocated to the calculation of non-bonded forces. For a system with N atoms, the number of bonded terms that need to be calculated are on the order $O(N)$. However, for non-bonded interactions, the number of terms scales to the order $O(N^2)$ since these interactions are calculated for every pair of

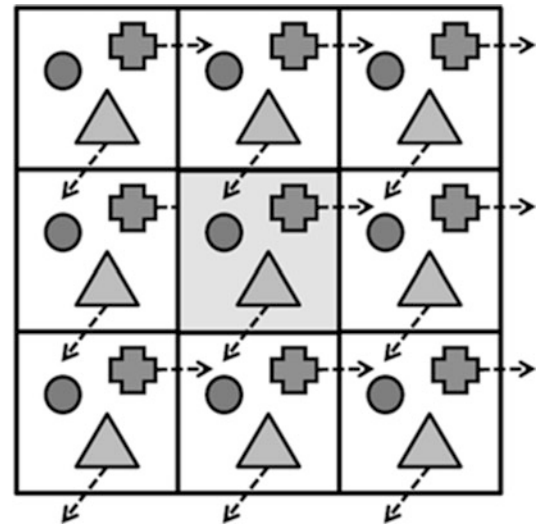


Fig. 2 A schematic of periodic boundary conditions in two dimensions. During the course of simulations, particles can move out of the central primary box. At regular intervals, periodic boundary conditions are applied and image centering is performed such that atoms that are found to have moved out of the central box in one direction have their images enter the box from the opposite direction

atoms in the system [59]. For vdW interactions, it can be seen from the Lennard-Jones potential that these interactions decrease rapidly with distance. One of the most popular methods for approximating these non-bonded interactions is through the use of cutoffs. Interactions between atoms that are beyond a cutoff distance are set to zero. The *minimum image convention* is also applied if PBCs are used. In this convention, an atom interacts with, at most, only one image of each atom in the system. This prevents the atom from interacting with its own image or with another atom twice. This method thus limits the cutoff distance to one half the length of the primary cell. Electrostatic interactions, however, do not fall off as rapidly as Lennard-Jones interactions and the use of the minimum image convention can lead to artifacts when dealing with electrostatics. To deal with this, lattice methods such as the Ewald summation are employed which enables electrostatics interactions to be calculated in full without distance cutoffs [43, 59, 60].

3.2 Constructing a Model Biomolecular System

In this section, the construction of a model biomolecular system will be discussed in the context of the Chemistry at HARvard Molecular Mechanics (CHARMM) program. A full discussion of the outline of a generic CHARMM project can be found in one of the CHARMM papers [65]. A good resource for familiarizing oneself with CHARMM can also be found online at http://www.charmmtutorial.org. A short discussion of the main parts of a typical CHARMM input file is presented below.

3.2.1 The Typical CHARMM Input File

1. *Loading the topology and parameter files.* These files, as mentioned above, are necessary for building structures and evaluating the potential energy function (**Note 1**). These are read in by the following statements:

```
open read card unit 10 name @TOPOLOGY_FILENAME
read rtf card unit 10
open read card unit 20 name @PARAMETER_FILENAME
read para card unit 20
```

   The symbol `@` is used to signify a variable name. Its use in this context means that the name of the file that will be accessed is contained in the variable **`TOPOLOGY_FILENAME`**. The `unit` designation simply assigns the file name to a Fortran logical unit number.

2. *Generating the protein structure file (PSF).* The PSF contains the list of all the atom types present in the model system together with a list of all bonds, angles, torsions and improper torsions. The PSF can be generated by reading in a PSF file:

```
open read card unit 10 name @PSF_FILENAME
read para card unit 10
```

   or by using the GENERATE statement of CHARMM after reading in the sequence of residues. This is especially true if the PSF file is not available. Note that the sequence refers not just to the amino acid residues but to all of the molecules that compose the system. For example, to generate the PSF for a system with two DPPC molecules, the following statements can be used if the PSF text file is not available.

```
read sequence card
* title
* ! End of title
2 ! number of residues
DPPC DPPC !residues specified
generate PC6 setup ! this is the command that actually sets up
the PSF
```

   `PC6` is the segment ID name that one assigns to this part of the PSF. Note that the "`!`" is a comment symbol in a CHARMM input file that goes through the end of the line. If a sequence of amino acid residues were specified, it is typical to see the `first` and `last` keywords as shown below

```
generate PEPT setup first @presn last @presc
```

   where **`@presn`** and **`@presc`** refer to patch residues, which are partial structures that can be added on to a stand-alone residue. An example of using a patch residue is one that connects two cysteine residues to create a disulfide bond. In the example above, the patches are used to cap the termini of the peptide sequence. An example of a standard N-terminus is the protonated amine while a standard C-terminus is a deprotonated carboxylic acid.

3. *Reading in the coordinates.* Coordinates can be read from a PDB file (**Note 2**)

```
open read card unit 10 name @PDB_FILENAME
read coor pdb unit 10
```

or from a CHARMM coordinate file via the following:

```
open read card unit 10 name @CRD_FILENAME
read coor card unit 10
```

4. *Set up the images for periodic boundary conditions.* If PBCs are employed for the simulation, the **crystal** and **image** modules of CHARMM provides for the interface to set up PBCs.

```
crystal define @crystaltype @A @B @C @alpha @beta @gamma
crystal build cutoff @cutoffval nope 0
image byresidue xcen 0.0 ycen 0.0 zcen 0.0 select water end
image bysegment xcen 0.0 ycen 0.0 zcen 0.0 select protein end
```

The first line specifies the crystal symmetry (**@crystaltype**) with the other variables providing the necessary information about the length of the sides of the crystal and the values of the corresponding angles. For a **cubic** crystal type, **@A**=**@B**=**@C** while **@alpha**=**@beta**=**@gamma**=**90**. The second line initiates the building of image atoms. The cutoff parameter value, **@cutoffval**, indicates the number of layers of image atoms that will be built.

The two lines with the **image** command controls how molecules are centered when the primary box is rebuilt after a certain number of simulation steps. The third line specifies that water molecules are shifted on a molecule by molecule basis when periodic boundary conditions are applied. Doing the same **byresidue** shifts to protein segments will essentially cause the protein to break, thus, proteins are shifted on a segment by segment basis.

5. *Specify the treatment of non-bonded interactions.* This section of a CHARMM input file details the type of switching that will be employed for non-bonded interactions and whether an Ewald summation will calculate electrostatic interactions. A sample statement is shown below.

```
nbonds atom vatom -
  ctonnb 10.0 ctofnb 12.0 -
  ewald pmew fftx @fftx ffty @ffty fftz @fftz kappa .34 spline
order 6
```

The first line contains directions for how cutoffs are handled, in this case atom-based cutoffs are used for electrostatics and van der Waals interactions (**vatom**). The second line specifies the cutoff distances within which interactions between atoms will be calculated. Lennard-Jones interactions are varied by a switching function between the **ctonnb** and **ctofnb** values. The third line provides the necessary information for doing the

Ewald summation, in particular via the particle-mesh Ewald algorithm (**pmew**), to calculate electrostatics without use of cutoffs. The keywords **fftx**, **fftx**, and **fftz** control the grid sizes for use in the particle mesh Ewald summation (**Note 3**).

6. *Have the system undergo molecular dynamics simulation*. This section spells out the commands and parameters that control the dynamics run of the system.

```
DYNA CPT leap restart nstep @nstep timestep 0.002 -
     iunrea 11 iunwri 12 iuncrd 13 -
     nsavc 100 -
     PCONS pint pref 1.0 pmxx 0. pmyy 0. pmzz @Pmass pgamma 20.0 -
     HOOVER reft @temp tmass 2000.0 tbath @temp firstt @temp
```

The **DYNA** keyword directs CHARMM to start a dynamics simulation of the system. The first line contains information on which ensemble to maintain (**CPT** or constant pressure and temperature), the integration algorithm to be used (the leap-frog algorithm), whether to use a previous simulation as a starting point (**restart**) (**Note 4**), the number of time steps (**nstep**) (**Note 5**), and the size of the time step (**timestep**). If a previous simulation is used as a starting point, the restart file written at the end of that previous simulation needs to be accessed. The second line points to the unit number of files that will be read if the simulation is restarted from a previous one (**iunrea**), where to write the restart file information from the current simulation (**iunwri**) and the destination for the trajectory file (**iuncrd**). The trajectory file contains the coordinates of all the atoms collected at regular intervals. The frequency of saving coordinates to the trajectory file is given by the **nsavc** keyword. In this case, coordinates will be collected at every 100 time steps and saved (**Note 6**). The fourth line contains the parameters that will be used for pressure control of the system while the last line contains the same information for temperature control.

3.2.2 Constructing a Lipid Bilayer or Protein–Lipid Bilayer System

Constructing a lipid bilayer system with or without a protein or peptide follows the same general procedure. A general method is outlined below for building the model systems. This method was developed by Benoit Roux (http://thallium.bsd.uchicago.edu/RouxLab/membrane.html) and is also discussed in more detail elsewhere [69]. A Web-based method with a graphical user interface can also be utilized (http://www.charmm-gui.org/) [70, 71]. This latter method was developed by the group of Wonpil Im and follows the same general outline as Roux's technique. The method, in summary, involves a bilayer that is constructed around the protein or peptide that is being simulated. The component lipids are randomly selected from a library of individual lipid molecules that

was extracted from an equilibrated and hydrated bilayer. The water layer surrounding the lipid bilayer is built from smaller boxes of water molecules (**Note 7**). If an equilibrated lipid bilayer is available, another method of incorporating the protein or peptide involves the deletion of a certain number of lipids whose total cross-sectional area would match that of the protein or simply to remove lipid molecules whose atoms overlap with an inserted protein [44, 72].

1. *Determining the size of the primary simulation box.* The lateral size (size along the *xy*-plane) of the primary simulation box is determined by two components: the number of lipids on one leaflet and the cross-sectional area of the protein. The lipid component of the box size depends on the number and type of lipid molecules that comprises the lipid bilayer. Each lipid type has a characteristic head group cross sectional area. Suggested values for lipid head group areas can be found in http://www.charmm-gui.org/?doc=input/membrane_only&step=1. In the presence of protein or peptide, the cross-sectional area of this inclusion also needs to be calculated. This is done by using a probe of a certain size (typically, the size of a methylene group for a peptide that is embedded in a bilayer) to measure the solvent-accessible surface area of the protein. This latter calculation is performed by the CHARMM input file, sys1.inp, in Roux's method.
2. *Building the bilayer structure.* After the lateral size of the primary box has been determined, the bilayer system can be constructed. The protein is placed in the center of the primary box. Dummy atoms are then introduced into the system; each will be replaced by a lipid molecule further on. These are placed randomly around the protein and in the top and bottom leaflet of the bilayer. The vertical placement of these dummy atoms is dependent on the length of the acyl chain of the lipids comprising the bilayer. This procedure is performed by the input file sys2.inp. After the placement of the dummy atoms, a minimization procedure is then performed (via sys3.inp) to find the spatial distribution of dummy atoms around the protein with the lowest energy. After the spatial distribution of the dummy atoms in the bilayer has been determined, each dummy atom is then consecutively replaced by a lipid randomly selected from a library of lipid molecules. Afterwards, systematic rotations and translations of the lipid molecules are done in order to remove bad contacts. These latter steps are performed using the input file sys4.inp. At this time, the system is composed of the protein/peptide and the lipid bilayer. The input file sys5.inp then performs energy minimization procedures on the system.

3. *Adding the water layer*. The assembly of the water layer is done by input files sys6.inp, sys7.inp, and sys8.inp. Starting from a small box of water molecules, a sheet of water boxes that would cover the lateral area of the primary box is first assembled (sys6.inp). This sheet of water is then replicated vertically until the height of the stacked water layers is at least equal to the desired height of the primary simulation box (sys7.inp). Finally, the water boxes are fitted on each side of the lipid bilayer (sys8.inp).
4. *Minimizing the energy of the full system and addition of ions*. A series of energy minimization procedures is then initialized (sys9.inp to sys18.inp). If the total charge of the system is not equal to zero, ions are added in order to neutralize the charge. This is done in order to make use of the particle mesh Ewald summation algorithm for calculating electrostatic interactions.

3.2.3 Generating a Curved Bilayer

The generation of curved bilayers in silico is essential for computational investigations into the interaction of curvature-sensing peptides with curved membranes. One method involves simulating a bilayer with a known curvature-generating protein such as BAR [73] and using the resulting curved bilayer with the protein removed [74]. Another method introduces an asymmetry across the bilayer, in particular by using a heterogeneous distribution of single-tailed and double-tailed lipids in the two bilayer leaflets [75]. In our laboratory, the method that was used involves increasing the lipid density by the gradual compression of a flat lipid bilayer along the *x*- and *y*-axes [76]. This technique is summarized below.

1. *Construct a flat lipid bilayer*. The method outlined above is used for this step.
2. *Scaling of the system along the xy plane*. The scaling of coordinates entails multiplying the *x*- and *y*-coordinates of each particle by the same factor. Using a factor greater than one has the effect of moving each particle outwards, that is, away from the *z*-axis while using a factor less than one moves each particle closer to the *z*-axis. It is the latter that causes a compression of the system along the *xy* plane, and thus, an increase in the lateral lipid density. To maintain the compression of particles, the scaling of coordinates needs an accompanying scaling of the primary box dimensions in order to maintain both the overall density of the system and the volume of the primary box size. The *x*-*y* dimensions are each scaled by the same factor used for the particle coordinates and the *z*-dimension is multiplied by the square of the reciprocal of the scaling factor. To avoid sudden changes in the bilayer structure, the compression is performed gradually by using small changes in the scaling factor. In our work, the coordinates are scaled in increments of 2 % as outlined in the steps below.

3. Multiply the *x*- and *y*-coordinates of each particle by 0.98 (i.e., a 2 % compression).
4. Multiply the *x*- and *y*-dimensions of the primary box by 0.98 and multiply the *z*-dimension by $1/(0.98)^2$.
5. Perform an energy minimization.
6. Let the system undergo a dynamics procedure for 5 ps.
7. Repeat **steps 3–6** where during each iteration, decrease the scaling factor by another 2 %.
8. A compression to 76 % of the original lateral area of the simulation box typically translates to a 4- to 6-Å decrease in the radius of curvature of the bilayer (where the ideal flat bilayer is assumed to have an infinite radius of curvature).

Figure 3a shows a diagram of the coordinate scaling procedure and Fig. 3b shows the curved membranes resulting from an increase in the lateral density of the lipids brought about by coordinate scaling.

For this section, the work done on a peptide segment of the effector domain of myristoylated alanine-rich C kinase (MARCKS-ED) will

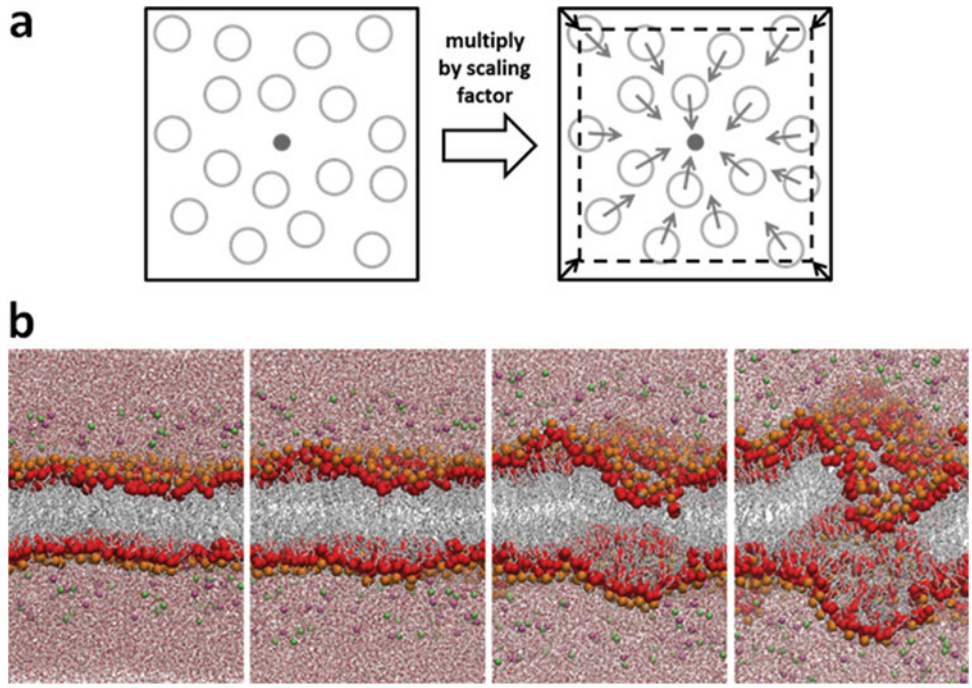


Fig. 3 Panel (**a**) shows a diagram of how the coordinate scaling procedure is performed leading to an increase in the lateral density of particles. Panel (**b**) shows the increase in membrane curvature caused by the increase in lateral density of lipids. The *large balls* represent the interfacial region of the lipid bilayer, with the *red-colored ones* representing carbonyl oxygens in the lipid molecule and *orange ones* representing phosphorus atoms

3.2.4 Simulating and Designing Curvature-Sensing Peptides

be described. The MARCKS-ED peptide has 25 residues with the following sequence: *KKKKKRFSFKKSFKLSGFSFKKNKK*. As mentioned beforehand, the typical starting point for protein structures is the Protein Data Bank. However, in the MARCKS-ED peptide, predominant secondary structures are absent [35]. Due to this and the relative shortness of the sequence, building the peptide from scratch is sufficient. To build the peptide, the sequence of residues is specified in the CHARMM input file. Based on internal coordinates, the CHARMM program will construct a straight chain of the residues as shown in Fig. 4 where the backbone atoms are red. To create the initial structure for MARCKS-ED, the text below can be used in a CHARMM input file. This will give a segment ID (segid) of MRX to the peptide and will cap the termini with the standard N- and C-terminus.

```
read sequence card
* MARCKS-ED
*
25
lys lys lys ... ! specify the rest of the residue sequence
generate MRX setup first NTER last CTER
```

To construct the coordinates, the internal coordinate (`IC`) module of CHARMM is accessed

```
ic para
ic seed 1 N 1 CA 1 C
ic fill
ic build
```

The `ic para` command fills in missing information from the IC tables in the topology files with values from the parameter file. The `ic seed` command specifies the positions of the three atoms specified (i.e., the N, CA and C atoms of residue 1, in this instance). The first atom is placed at the origin, the second on the *x*-axis and the third, on the *xy*-plane. Since the atomic coordinates of three atoms are now known, the other missing IC values can be filled with the command `ic fill` and `ic build` (**Note 8**).

Further modifications to the initial structure can be made if other characteristics are known. For example, if this stretch of peptide were to be simulated as an alpha helix, the dihedral angles of the backbone atoms would be modified to have values

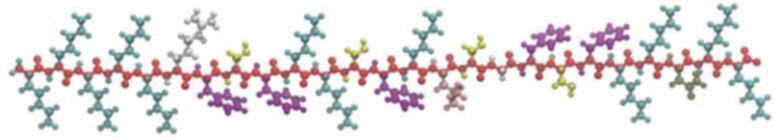


Fig. 4 CPK representation of the MARCKS-ED peptide after being built using internal coordinates

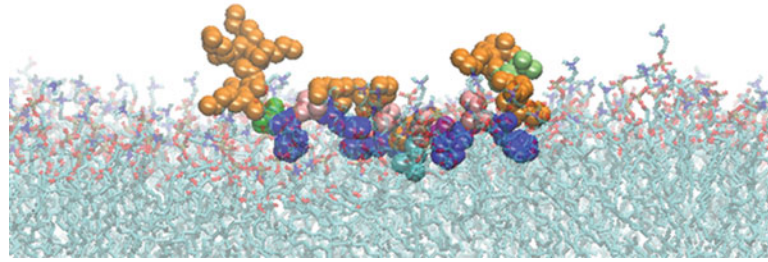


Fig. 5 The MARCKS-ED peptide adopting a "boat" conformation after a period of dynamics. The hydrophilic Lys residues (*orange balls*) are pointed towards the bulk solvent while the Phe residues (*blue balls*) remain buried in the interfacial region of the lipid bilayer

characteristic of an alpha helix. The MARCKS-ED peptide is placed around the depth of the interfacial region of the bilayer due to the fact that the peptide is known to respond and interact with the bilayer interface [41]. MD simulations of this peptide confirm a "boat conformation" where the hydrophilic Lys residues at the ends of the peptide are stretched towards the bulk solvent while the central stretch containing the Phe residues are buried into the bilayer [41, 77]. A snapshot of this conformation is shown in Fig. 5.

One modification to the design of the MARCKS-ED peptide can come in the form of using the enantiomeric D-form of the peptides. The use of the D-isomer is desirable in some cases since they are less prone to proteolytic processes in vivo. In studies done in our laboratory, D-MARCKS-ED has been shown to also sense membrane curvature [41]. The CHARMM topology files contains the internal coordinates for the L-form of the different amino acids, thus, modifications to the topology files are necessary to simulate D-amino acids. Recall that the chirality is controlled by the improper dihedral. To change the chirality of an L-amino acid to the D-form, two improper dihedral entries in the internal coordinate (IC) table are changed. These IC table entries pertain to two sets of four atoms: the backbone atoms N, C, C_α and side chain atom C_β and the backbone atoms N, C, C_α and the attached H_α. In particular, the modification is to switch the sign of the improper dihedral angle for these two sets of atoms. An example is shown for alanine. The relevant IC table entry for L-ala is

```
IC N C *CA CB 1.4592 114.4400 123.2300 111.0900 1.5461
IC N C *CA HA 1.4592 114.4400 -120.4500 106.3900 1.0840
```

The corresponding entry for D-ala is

```
IC N C *CA CB 1.4592 114.4400 -123.2300 111.0900 1.5461
IC N C *CA HA 1.4592 114.4400 120.4500 106.3900 1.0840
```

It is also typical in modifying peptides to change certain residues to effect some change in function. For the MARCKS-ED

peptide, it was shown that Phe residues play a major role in the attachment of the peptide to the lipid bilayer, in particular, through its interactions with the interfacial region of the bilayer. A change that could be explored in the design of this peptide is to change the Phe residues to other residues. Examples of such replacement residues are Tyr and Trp, which are known to also interact with the bilayer interface [78, 79].

4 Notes

1. A typical CHARMM installation will contain topology and parameter files that contain the required information for amino acids, lipids, water or sugars. In cases where small organic molecules need to be included the simulation system, the CHARMM General Force Field for organic molecules (CGENFF program: http://cgenff.paramchem.org/) can be utilized to generate initial topologies and parameters for these small molecules [80, 81].
2. Most PDB files do not contain coordinates for hydrogen atoms due to the resolution limits of X-ray crystallography. In these cases, CHARMM facilities such as HBUILD can be used to add coordinates for the hydrogen atoms.
3. The grid size value should be greater than or equal to the corresponding cell dimension and should preferably be composed of prime factors 2 or 3.
4. For simulations that do not start from a previous one, the ISEED keyword is typically added followed by a number (e.g., `iseed 31514495`). The number serves as the seed used by CHARMM in assigning velocities.
5. Simulations can be performed in two ways. For example, a 1-ns simulation can be obtained from a single simulation run of 500,000 time steps (assuming a 0.002-fs time step size). The same total simulation time can also be obtained from ten successive simulations (each of which, except the first one, started from the preceding simulation) of 50,000 time steps. Assuming identical initial conditions, both options will give similar results. However, it is good practice to use the second option and break down long simulations into shorter ones. This lessens the impact of possible problems in computing resources. For instance, when the nodes running the simulation in a supercomputer cluster break down, data from the middle of a long single simulation might not get saved and might entail restarting the simulation from an earlier time step.
6. The trajectory file is typically the largest that will be generated by an MD run as it contains multiple sets or frames of the

coordinates of the whole system. Thus, as simulation systems get larger and time scales become longer, a decision has to be made on the frequency of writing the coordinates to the trajectory file in order to manage disk storage more efficiently. This would depend on what information is desired from the simulation. The initial runs can be made with a lower **nsavc** number and could be checked if lessening the frequency of saving coordinates will not lead to a meaningful loss of detail in the desired information. After this check, a higher value for **nsavc** may be used if appropriate.

7. Both of these methods for constructing protein–lipid bilayer systems involve separate steps that generate intermediate structures. It is always good to visually check these intermediate structures (via VMD or Rasmol) to confirm that the system is being built correctly.
8. Though the `ic build` statement will provide coordinates for missing hydrogen atoms, it is often the case that an additional statement (`hbuild select all end`) is typically added after `ic build`.

Acknowledgment

We thank the National Institutes of Health (R01GM103843) and the National Science Foundation (CHE0954819) for financial support for this work. This work utilized the Janus supercomputer, which is supported by the National Science Foundation (award number CNS-0821794), the University of Colorado-Boulder, the University of Colorado-Denver, and the National Center for Atmospheric Research. The Janus supercomputer is operated by the University of Colorado Boulder. We also acknowledge the use of the Vieques cluster at the BioFrontiers Institute of the University of Colorado-Boulder.

References

1. Baumgart T, Capraro BR, Zhu C, Das SL (2011) Thermodynamics and mechanics of membrane curvature generation and sensing by proteins and lipids. Annu Rev Phys Chem 62:483–506
2. Callan-Jones A, Bassereau P (2013) Curvature-driven membrane lipid and protein distribution. Curr Opin Solid State Mater Sci 17:143–150
3. Aimon S, Callan-Jones A, Berthaud A, Pinot M, Toombes GES, Bassereau P (2014) Membrane shape modulates transmembrane protein distribution. Dev Cell 28:212–218
4. Koller D, Lohner K (2014) The role of spontaneous lipid curvature in the interaction of interfacially active peptides with membranes. Biochim Biophys Acta 1838:2250–2259
5. Bigay J, Antonny B (2012) Curvature, lipid packing, and electrostatics of membrane organelles: defining cellular territories in determining specificity. Dev Cell 23:886–895
6. Janmey PA, Kinnunen PKJ (2006) Biophysical properties of lipids and dynamic membranes. Trends Cell Biol 16:538–546
7. McMahon HT, Gallop JL (2005) Membrane curvature and mechanisms of dynamic cell membrane remodelling. Nature 438:590–596

8. Zimmerberg J, Kozlov MM (2006) How proteins produce cellular membrane curvature. Nat Rev Mol Cell Biol 7:9–19
9. Haney EF, Nathoo S, Vogel HJ, Prenner EJ (2010) Induction of non-lamellar lipid phases by antimicrobial peptides: a potential link to mode of action. Chem Phys Lipids 163:82–93
10. Antonny B (2011) Mechanisms of membrane curvature sensing. Annu Rev Biochem 80:101–123
11. Hatzakis NS, Bhatia VK, Larsen J, Madsen KL, Bolinger PY, Kunding AH, Castillo J, Gether U, Hedegard P, Stamou D (2009) How curved membranes recruit amphipathic helices and protein anchoring motifs. Nat Chem Biol 5:835–841
12. Graham TR, Kozlov MM (2010) Interplay of proteins and lipids in generating membrane curvature. Curr Opin Cell Biol 22:430–436
13. Jao CC, Hegde BG, Gallop JL, Hegde PB, McMahon HT, Haworth IS, Langen R (2010) Roles of Amphipathic Helices and the Bin/Amphiphysin/Rvs (BAR) domain of endophilin in membrane curvature generation. J Biol Chem 285:20164–20170
14. Farsad K, Ringstad N, Takei K, Floyd SR, Rose K, De Camilli P (2001) Generation of high curvature membranes mediated by direct endophilin bilayer interactions. J Cell Biol 155:193–200
15. Peter BJ, Kent HM, Mills IG, Vallis Y, Butler PJ, Evans PR, McMahon HT (2004) BAR domains as sensors of membrane curvature: the amphiphysin BAR structure. Science 303:495–499
16. Hui E, Johnson CP, Yao J, Dunning FM, Chapman ER (2009) Synaptotagmin-mediated bending of the target membrane is a critical step in Ca(2+)-regulated fusion. Cell 138:709–721
17. Drin G, Casella JF, Gautier R, Boehmer T, Schwartz TU, Antonny B (2007) A general amphipathic alpha-helical motif for sensing membrane curvature. Nat Struct Mol Biol 14:138–146
18. Zimmerberg J, McLaughlin S (2004) Membrane curvature: how BAR domains bend bilayers. Curr Biol 14:R250–R252
19. Fleming A, Sampey G, Chung MC, Bailey C, van Hoek ML, Kashanchi F, Hakami RM (2014) The carrying pigeons of the cell: exosomes and their role in infectious diseases caused by human pathogens. Pathog Dis 71:107–118
20. Azmi AS, Bao B, Sarkar FH (2013) Exosomes in cancer development, metastasis, and drug resistance: a comprehensive review. Cancer Metastasis Rev 32:623–642
21. Belting M, Wittrup A (2008) Nanotubes, exosomes, and nucleic acid-binding peptides provide novel mechanisms of intercellular communication in eukaryotic cells: implications in health and disease. J Cell Biol 183:1187–1191
22. van Niel G, Porto-Carreiro I, Simoes S, Raposo G (2006) Exosomes: a common pathway for a specialized function. J Biochem 140:13–21
23. Luga V, Zhang L, Viloria-Petit AM, Ogunjimi AA, Inanlou MR, Chiu E, Buchanan M, Hosein AN, Basik M, Wrana JL (2012) Exosomes mediate stromal mobilization of autocrine Wnt-PCP signaling in breast cancer cell migration. Cell 151:1542–1556
24. Ono M, Kosaka N, Tominaga N, Yoshioka Y, Takeshita F, Takahashi RU, Yoshida M, Tsuda H, Tamura K, Ochiya T (2014) Exosomes from bone marrow mesenchymal stem cells contain a microRNA that promotes dormancy in metastatic breast cancer cells. Sci Signal 7:1–10
25. Taylor DD, Gercel-Taylor C (2008) MicroRNA signatures of tumor-derived exosomes as diagnostic biomarkers of ovarian cancer. Gynecol Oncol 110:13–21
26. Rabinowits G, Gercel-Taylor C, Day JM, Taylor DD, Kloecker GH (2009) Exosomal MicroRNA: a diagnostic marker for lung cancer. Clin Lung Cancer 10:42–46
27. Taylor DD, Gercel-Taylor C (2011) Exosomes/microvesicles: mediators of cancer-associated immunosuppressive microenvironments. Semin Immunopathol 33:441–454
28. Peinado H, Aleckovic M, Lavotshkin S, Matei I, Costa-Silva B, Moreno-Bueno G, Hergueta-Redondo M, Williams C, Garcia-Santos G, Ghajar CM, Nitadori-Hoshino A, Hoffman C, Badal K, Garcia BA, Callahan MK, Yuan JD, Martins VR, Skog J, Kaplan RN, Brady MS, Wolchok JD, Chapman PB, Kang YB, Bromberg J, Lyden D (2012) Melanoma exosomes educate bone marrow progenitor cells toward a pro-metastatic phenotype through MET. Nat Med 18:883–891
29. Izquierdo-Useros N, Naranjo-Gomez M, Erkizia I, Puertas MC, Borras FE, Blanco J, Martinez-Picado J (2010) HIV and mature dendritic cells: trojan exosomes riding the trojan horse? PLoS Pathog 6:1–9
30. Lenassi M, Cagney G, Liao MF, Vaupotic T, Bartholomeeusen K, Cheng YF, Krogan NJ, Plemenitas A, Peterlin BM (2010) HIV Nef is secreted in exosomes and triggers apoptosis in bystander CD4(+) T cells. Traffic 11:110–122
31. Kadiu I, Narayanasamy P, Dash PK, Zhang W, Gendelman HE (2012) Biochemical and biologic characterization of exosomes and

microvesicles as facilitators of HIV-1 Infection in macrophages. J Immunol 189:744–754
32. Vella LJ, Sharples RA, Nisbet RM, Cappai R, Hill AF (2008) The role of exosomes in the processing of proteins associated with neurodegenerative diseases. Eur Biophys J 37:323–332
33. Vella LJ, Sharples RA, Lawson VA, Masters CL, Cappai R, Hill AF (2007) Packaging of prions into exosomes is associated with a novel pathway of PrP processing. J Pathol 211:582–590
34. Saludes JP, Morton LA, Ghosh N, Beninson LA, Chapman ER, Fleshner M, Yin H (2012) Detection of highly curved membrane surfaces using a cyclic peptide derived from Synaptotagmin-I. ACS Chem Biol 7:1629–1635
35. Morton LA, Yang HW, Saludes JP, Fiorini Z, Beninson L, Chapman ER, Fleshner M, Xue D, Yin H (2013) MARCKS-ED peptide as a curvature and lipid sensor. ACS Chem Biol 8:218–225
36. Wang J, Gambhir A, Hangyas-Mihalyne G, Murray D, Golebiewska U, McLaughlin S (2002) Lateral sequestration of phosphatidylinositol 4,5-bisphosphate by the basic effector domain of myristoylated alanine-rich C kinase substrate is due to nonspecific electrostatic interactions. J Biol Chem 277:34401–34412
37. Ellena JF, Burnitz MC, Cafiso DS (2003) Location of the myristoylated alanine-rich C-kinase substrate (MARCKS) effector domain in negatively charged phospholipid bicelles. Biophys J 85:2442–2448
38. Zhang W, Crocker E, McLaughlin S, Smith SO (2003) Binding of peptides with basic and aromatic residues to bilayer membranes: phenylalanine in the myristoylated alanine-rich C kinase substrate effector domain penetrates into the hydrophobic core of the bilayer. J Biol Chem 278:21459–21466
39. Bigay J, Gounon P, Robineau S, Antonny B (2003) Lipid packing sensed by ArfGAP1 couples COPI coat disassembly to membrane bilayer curvature. Nature 426:563–566
40. Bigay J, Casella JF, Drin G, Mesmin B, Antonny B (2005) ArfGAP1 responds to membrane curvature through the folding of a lipid packing sensor motif. EMBO J 24:2244–2253
41. Morton LA, Tamura R, de Jesus AJ, Espinoza A, Yin H (2014) Biophysical investigations with MARCKS-ED: dissecting the molecular mechanism of its curvature sensing behaviors. Biochim Biophys Acta 1838:3137–3144
42. Dror RO, Dirks RM, Grossman JP, Xu HF, Shaw DE (2012) Biomolecular simulation: a computational microscope for molecular biology. Annu Rev Biophys 41:429–452
43. Vorobyov I, Allen TW (2009) Molecular dynamics computations for proteins: a case study in membrane ion permeation. In: Bohr HG (ed) Handbook in molecular biophysics. Wiley, Weinheim
44. Biggin PC, Bond PJ (2015) Molecular dynamics simulations of membrane proteins. Methods Mol Biol 1215:91–108
45. Harvey MJ, De Fabritiis G (2012) High-throughput molecular dynamics: the powerful new tool for drug discovery. Drug Discov Today 17:1059–1062
46. Lindahl E (2015) Molecular dynamics simulations. Methods Mol Biol 1215:3–26
47. McGregor DP (2008) Discovering and improving novel peptide therapeutics. Curr Opin Pharmacol 8:616–619
48. Croft NP, Purcell AW (2011) Peptidomimetics: modifying peptides in the pursuit of better vaccines. Expert Rev Vaccines 10:211–226
49. Lien S, Lowman HB (2003) Therapeutic peptides. Trends Biotechnol 21:556–562
50. Pujals S, Sabido E, Tarrago T, Giralt E (2007) all-D proline-rich cell-penetrating peptides: a preliminary in vivo internalization study. Biochem Soc Trans 35:794–796
51. Brooks BR, Bruccoleri RE, Olafson BD, States DJ, Swaminathan S, Karplus M (1983) Charmm – a program for macromolecular energy, minimization, and dynamics calculations. J Comput Chem 4:187–217
52. Weiner PK, Kollman PA (1981) Amber – Assisted Model-Building with Energy Refinement – a general program for modeling molecules and their interactions. J Comput Chem 2:287–303
53. Scott WRP, Hunenberger PH, Tironi IG, Mark AE, Billeter SR, Fennen J, Torda AE, Huber T, Kruger P, van Gunsteren WF (1999) The GROMOS biomolecular simulation program package. J Phys Chem A 103:3596–3607
54. Phillips JC, Braun R, Wang W, Gumbart J, Tajkhorshid E, Villa E, Chipot C, Skeel RD, Kale L, Schulten K (2005) Scalable molecular dynamics with NAMD. J Comput Chem 26:1781–1802
55. Humphrey W, Dalke A, Schulten K (1996) VMD: visual molecular dynamics. J Mol Graph Model 14:33–38
56. Sayle RA, Milnerwhite EJ (1995) Rasmol – biomolecular graphics for all. Trends Biochem Sci 20:374–376
57. The PyMOL Molecular Graphics System, Version 1.8 Schrödinger, LLC

58. Allen P, Tildesley DJ (1989) Computer simulation of liquids. Clarendon, Oxford
59. Leach AR (2001) Molecular modelling: principles and applications. Prentice Hall, Harlow, England
60. Frenkel D, Smit B (2001) Understanding molecular simulation: from algorithms to applications. Academic, San Diego
61. Durrant JD, McCammon JA (2011) Molecular dynamics simulations and drug discovery. BMC Biol 9:1–9
62. Killian JA, Salemink I, dePlanque MRR, Lindblom G, Koeppe RE, Greathouse DV (1996) Induction of nonbilayer structures in diacylphosphatidylcholine model membranes by transmembrane alpha-helical peptides: importance of hydrophobic mismatch and proposed role of tryptophans. Biochemistry 35:1037–1045
63. de Jesus AJ, Allen TW (2013) The determinants of hydrophobic mismatch response for transmembrane helices. Biochim Biophys Acta 1828:851–863
64. Verlet L (1967) Computer experiments on classical fluids. I. Thermodynamical properties of Lennard-Jones molecules. Phys Rev 159:98–103
65. Brooks BR, Brooks CL, Mackerell AD, Nilsson L, Petrella RJ, Roux B, Won Y, Archontis G, Bartels C, Boresch S, Caflisch A, Caves L, Cui Q, Dinner AR, Feig M, Fischer S, Gao J, Hodoscek M, Im W, Kuczera K, Lazaridis T, Ma J, Ovchinnikov V, Paci E, Pastor RW, Post CB, Pu JZ, Schaefer M, Tidor B, Venable RM, Woodcock HL, Wu X, Yang W, York DM, Karplus M (2009) CHARMM: the biomolecular simulation program. J Comput Chem 30:1545–1614
66. Nosé S (1984) A molecular dynamics method for simulations in the canonical ensemble. Mol Phys 52:255–268
67. Hoover WG (1985) Canonical dynamics: equilibrium phase-space distributions. Phys Rev A 31:1695–1697
68. Feller SE, Zhang YH, Pastor RW, Brooks BR (1995) Constant-pressure molecular-dynamics simulation – the Langevin piston method. J Chem Phys 103:4613–4621
69. Langham A, Kaznessis YN (2010) Molecular simulations of antimicrobial peptides. Methods Mol Biol 618:267–285
70. Jo S, Kim T, Iyer VG, Im W (2008) CHARMM-GUI: a web-based graphical user interface for CHARMM. J Comput Chem 29:1859–1865
71. Wu EL, Cheng X, Jo S, Rui H, Song KC, Davila-Contreras EM, Qi Y, Lee J, Monje-Galvan V, Venable RM, Klauda JB, Im W (2014) CHARMMGUI membrane builder toward realistic biological membrane simulations. J Comput Chem 35:1997–2004
72. Wolf MG, Hoefling M, Aponte-Santamaria C, Grubmuller H, Groenhof G (2010) g_membed: efficient insertion of a membrane protein into an equilibrated lipid bilayer with minimal perturbation. J Comput Chem 31:2169–2174
73. Blood PD, Voth GA (2006) Direct observation of Bin/amphiphysin/Rvs (BAR) domain-induced membrane curvature by means of molecular dynamics simulations. Proc Natl Acad Sci U S A 103:15068–15072
74. Cui HS, Lyman E, Voth GA (2011) Mechanism of membrane curvature sensing by amphipathic helix containing proteins. Biophys J 100:1271–1279
75. Meyer GR, Gullingsrud J, Schulten K, Martinac B (2006) Molecular dynamics study of MscL interactions with a curved lipid bilayer. Biophys J 91:1630–1637
76. de Jesus AJ, Kastelowitz N, Yin H (2013) Changes in lipid density induce membrane curvature. RSC Adv 3:13622–13625
77. Qin ZH, Cafiso DS (1996) Membrane structure of protein kinase C and calmodulin binding domain of myristoylated alanine rich C kinase substrate determined by site-directed spin labeling. Biochemistry 35:2917–2925
78. MacCallum JL, Bennett WFD, Tieleman DP (2008) Distribution of amino acids in a lipid bilayer from computer simulations. Biophys J 94:3393–3404
79. de Jesus AJ, Allen TW (2013) The role of tryptophan side chains in membrane protein anchoring and hydrophobic mismatch. Biochim Biophys Acta 1828:864–876
80. Vanommeslaeghe K, MacKerell AD (2012) Automation of the CHARMM General Force Field (CGenFF) I: bond perception and atom typing. J Chem Inf Model 52:3144–3154
81. Vanommeslaeghe K, Raman EP, MacKerell AD (2012) Automation of the CHARMM General Force Field (CGenFF) II: assignment of bonded parameters and partial atomic charges. J Chem Inf Model 52:3155–3168

Chapter 23

Computational Tools for Allosteric Drug Discovery: Site Identification and Focus Library Design

Wenkang Huang, Ruth Nussinov, and Jian Zhang

Abstract

Allostery is an intrinsic phenomenon of biological macromolecules involving regulation and/or signal transduction induced by a ligand binding to an allosteric site distinct from a molecule's active site. Allosteric drugs are currently receiving increased attention in drug discovery because drugs that target allosteric sites can provide important advantages over the corresponding orthosteric drugs including specific subtype selectivity within receptor families. Consequently, targeting allosteric sites, instead of orthosteric sites, can reduce drug-related side effects and toxicity. On the down side, allosteric drug discovery can be more challenging than traditional orthosteric drug discovery due to difficulties associated with determining the locations of allosteric sites and designing drugs based on these sites and the need for the allosteric effects to propagate through the structure, reach the ligand binding site and elicit a conformational change. In this study, we present computational tools ranging from the identification of potential allosteric sites to the design of "allosteric-like" modulator libraries. These tools may be particularly useful for allosteric drug discovery.

Key words Allosteric site, Allosteric modulator, Allosteric drug discovery, Allostery, Allosteric drug design

1 Introduction

Allostery, which is also known as allosteric regulation, is an essential biological phenomenon that plays significant roles in signal transduction pathways, metabolic processes, and genomic transcription [1, 2]. Perturbation at an allosteric site can rapidly shift the equilibrium of a protein conformational ensemble towards another state, thereby inducing local conformation change at an active site [3–5]. Potential perturbations include the binding of small molecules/ions and local chemical modifications [6–8]. Thus, allostery is the most direct mechanism for regulating the function of biological macromolecules. Insight into allostery can lead to new ideas for method development in allosteric drug discovery [9, 10].

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0_23,

Unlike orthosteric drugs, which compete with the substrates of target proteins at the active sites, allosteric drugs bind at a location other than an active site and influence the affinity or catalytic efficiency of biological macromolecules through the propagation of a perturbation signal [11–14]. Allosteric drugs have several advantages relative to orthosteric drugs. First, according to sequence conservation analyses [15, 16], allosteric sites are significantly less conserved than orthosteric sites; this phenomenon allows allosteric modulators to selectively target specific subtypes within receptor families [17, 18], resulting in higher selectivity and fewer side effects than orthosteric drugs. Second, allosteric drugs do not block substrate-protein interactions, and there is an upper bound to allosteric regulation. In addition, allosteric modulators can enhance the efficiency of orthosteric drugs [19]. For instance, the allosteric modulator GNF-2 binds to the myristate-binding site of T315I human Bcr-Abl. GNF-2 and the substrate-competitive inhibitor imatinib exhibit additive inhibitory activity against this mutated Bcr-Abl; as a result, a combination of these two drugs can be used to overcome drug resistance in cases of chronic myelogenous leukemia (CML) [20]. Thus, the identification of modulators targeting allosteric sites receives increasing attention in the field of drug discovery, and several allosteric drugs have been approved by the US FDA [21, 22]. For example, Genzyme's plerixafor is an allosteric antagonist of the C-X-C chemokine receptor type 4 (CXCR4) that enhances the mobilization of hematopoietic stem cells (HSCs).

Allosteric drug discovery also presents new challenges relative to traditional drug discovery approaches. The identification and characterization of drug binding sites is the first step of structure-based drug discovery. However, the locations of allosteric sites remain unclear for most drug targets [23]. Moreover, the discovery of allosteric modulators is hampered by several obstacles, such as the low affinities and unknown structural features of potential small allosteric molecules. In our prior work, we summarized the properties of allosteric sites [24] and allosteric modulators [25] and developed an allosteric site identification method named Allosite [26]. In addition, a preliminary filter for allosteric modulator discovery was also established. In this protocol, we introduce practical guidelines describing how to obtain predictions for allosteric sites and build a focused library of "allosteric-like" molecules.

1.1 Theory

The fundamental strategy for Allosite is to use the topology and physiochemical properties of protein pockets to build a classification model relating allosteric sites to other sites. We have extracted 90 nonredundant allosteric protein–allosteric modulator co-crystals from the AlloSteric Database [27, 28]. After feature selection, 21 pocket descriptors were characterized for each pocket identified by FPocket [29]. The classification model for allosteric site

identification was then trained and tested using a support vector machine [30]. In a cross-validation test, the success metrics for the Support Vector Machine (SVM) model were a sensitivity of ~83 % and a specificity of ~96 %. We have made the final model available on the Allosite Web server.

To reveal the structural specificity of allosteric modulators, 3916 known structurally diverse allosteric modulators in the Allosteric Database were compared with compounds from other databases (the Accelrys Available Chemicals Directory, the Accelrys Comprehensive Medicinal Chemistry database, the Chinese Natural Product Database, DrugBank, the MDDR database, and the NCI Open Database). Interestingly, relative to other modulators, allosteric modulators exhibit higher structure rigidity, with less rotatable bonds and more rings from ring systems. In addition, higher hydrophobicity is also observed for allosteric modulators; this finding is consistent with the hydrophobic characteristics of allosteric sites [25]. In summary, we established the following rule for differentiating allosteric modulators from other modulators: (1) molecular weight (MW) ≤ 600; (2) number of rotatable bonds (nRB) ≤ 6; (3) $2 \leq$ number of rings (nR) ≤ 5; (4) number of rings in the largest ring Systems (nRIS) $= 1$ or 2; and (5) $3 \leq \text{SlogP} \leq 7$.

2 Materials

2.1 Software for Visualizing Protein Structures

The PyMOL molecular graphics system is required for visualizing PDB files and allosteric sites. This system, which is a open-source software, is available at http://www.pymol.org.

2.2 Browser

The Allosite server requires a Web browser with JavaScript and cookies enabled. A recommendation to ensure that protein structures can be visualized correctly is to use the latest version of Firefox or Chrome to access Allosite.

3 Methods

In the following subsections, we first describe the individual steps that the Allosite server uses to identify allosteric sites and then describe how to construct focused libraries for the screening of allosteric modulators with a preliminary filter.

3.1 Input File Preparation

Allosite utilizes a method based on the proteins' three-dimensional structures, which can be obtained from the Protein Data Bank database (*see* **Note 1**). If there is no crystal structure for the query protein, homology modeling methods will be helpful for building the protein's 3D structure. The following considerations should be

taken into account to ensure the quality of the prediction. (1) We recommend using an X-ray structure with a resolution <2.5 Å. (2) There should be no missing loops in the main chain of the protein. (3) Small molecules, ions and solvents within the PDB structure will automatically be removed.

3.2 Job Submission

The Allosite Web server is freely available for use at http://mdl.shsmu.edu.cn/AST. Jobs can be submitted either by "PDB ID" or by "PDB File" (*see* Fig. 1). In "PDB ID" mode, users can specify their input by simply entering the 4-character PDB ID of their query protein. Users with their own experimental/model-based structures can choose the "PDB File" mode to browse their local hard drives and provide a protein structure file. A submitted query protein structure should be in standard PDB format. Another parameter, "Job Name", must be set before running the job; this parameter can then be used to check the status of the job and retrieve calculated results for the job at any time. After "Job Name" has been specified, users can click "Run" and select PDB chain(s) to submit the job (*see* **Note 2**).

3.3 Retrieving the Results

Once the job has been submitted, detailed job information, including a unique Job ID, will appear on the "Select PDB chains" page. Users can also track the progress of a job or access the results page from the "Job Queue" page by searching for their Job ID. The status of a job is refreshed in the "Job List" every 10 s until the "Finished" button appears. The "Finished" button indicates that the job has finished, and results can be viewed by clicking this button. The Allosite approach features rapid calculation times that depend on the size of the query protein. A typical Allosite job for a 400-residue protein will require ~15 s.

3.4 Analyzing the Results

The job will redirect to the calculation result page after the "Finished" button has been clicked. The GLmol applet will load automatically and provide a default color-coded representation of the query protein. The predicted allosteric site can be viewed in the GLmol applet by clicking the "Show Pocket" button. The predicted allosteric site is displayed as white spheres, and allosteric site residues are represented using a stick model. The result page also contains the following pocket properties for the predicted allosteric site: "Pocket Volume", "Pocket Total SASA", "Pocket Polar SASA", and "Pocket Druggability Score". A representative run of an Allosite job provides 0–4 potential allosteric sites.

3.5 Analyzing the Results Using PyMOL

Result files can be downloaded for offline analysis by clicking the "Download Report" link. After tar archives have been extracted, three files are obtained: a structure file for the query protein, site information for the predicted results, and a .pml PyMOL script for visualization. Users can then analyze the predicted allosteric site in PyMOL (*see* Fig. 2).

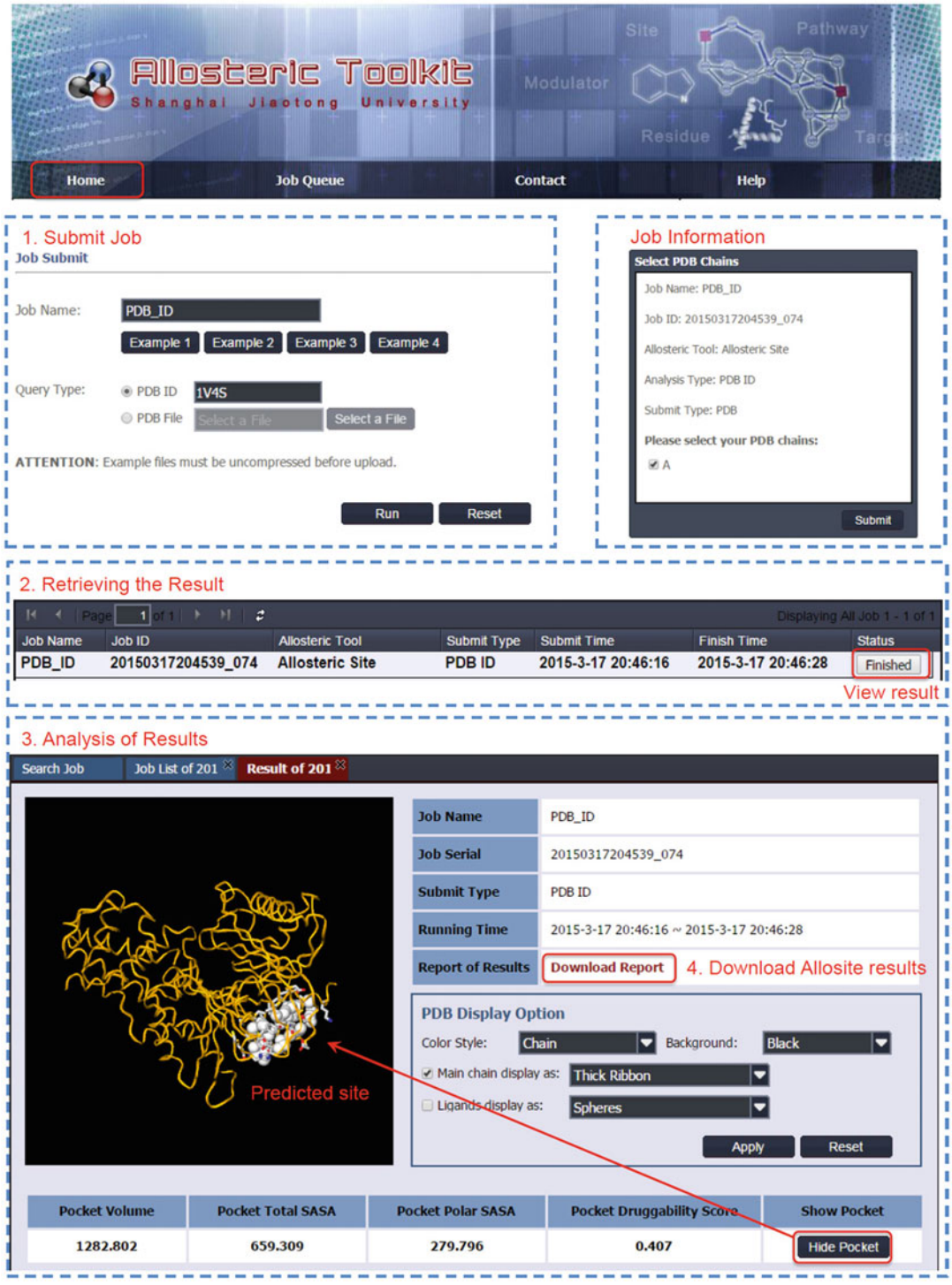


Fig. 1 The Web interface and workflow of Allosite

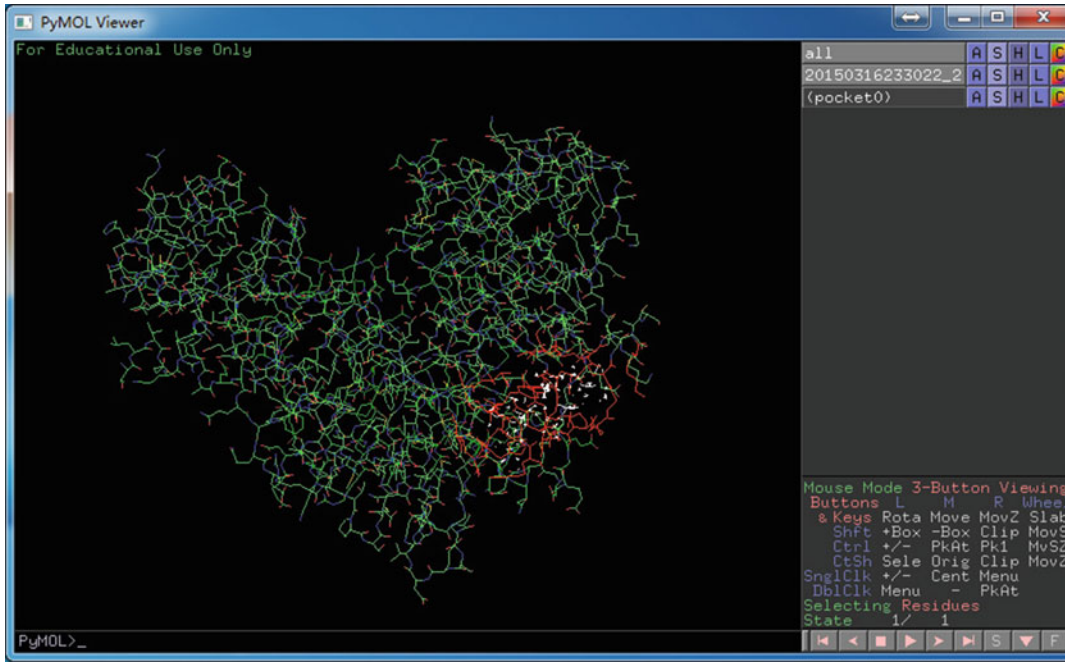


Fig. 2 An analysis of Allosite results using PyMOL. The allosteric pocket is represented by *white points. Red lines* are used to highlight residues in this site

3.6 Designing Focus Libraries of Allosteric Modulators

Based on our prior work, molecules that satisfy the following criteria are more likely to be allosteric modulators: (1) MW $\leq$ 600; (2) RBN $\leq$ 6; (3) 2 $\leq$ nR $\leq$ 5; (4) nRIS = 1 or 2; and (5) 3 $\leq$ SlogP $\leq$ 7. To fetch potential allosteric modulators from a database of chemical molecules, we developed a Web server that can be accessed at http://mdl.shsmu.edu.cn/ASD/. For each job, users can upload their molecular database of interest with either 2D or 3D structures. Three file types are acceptable for uploading: MOL, SDF, and SMILES. When our Web server completes a job, users are redirected to a results page where they can click "Download" to download molecules that have passed the "allosteric-like" filter. Histograms indicate the distribution of five calculated molecular properties. The filter's local script runs quickly and is recommended for users who intend to filter large molecular databases.

4 Notes

1. Many proteins have multiple structures in the PDB database that have been generated in different crystal environments. If a protein has multiple conformation states (such as a protein kinase with active and inactive states [13]), diverse conformations can

be separately submitted to the Allosite server to obtain robust results. Moreover, these crystal conformations only represent small proportions of the conformational ensembles of allosteric proteins. Therefore, it is difficult to identify cryptic allosteric sites in proteins because these sites are transient during conformational changes and invisible to conventional X-ray crystal structures [31]. Molecular dynamics (MD) simulations are widely used for conformational ensemble sampling and for generating representative structures from diverse conformations [32]. Thus, users can predict cryptic allosteric sites with Allosite if representative conformations are chosen as inputs.

2. One class of allosteric modulators binds to a pocket emerging from multimerization or protein–protein interactions, but these modulators do not directly inhibit protein interaction [33]. To identify interfacial allosteric sites, multiple chains should be chosen at the "PDB chain selection" step.

Acknowledgments

This project has been funded in whole or in part with Federal funds from the Frederick National Laboratory for Cancer Research, National Institutes of Health, under contract HHSN261200800001E. This research was supported [in part] by the Intramural Research Program of NIH, Frederick National Lab, Center for Cancer Research. The content of this publication does not necessarily reflect the views or policies of the Department of Health and Human Services, nor does mention of trade names, commercial products, or organizations imply endorsement by the US Government. This research was supported in part by Natural Science Foundation of China (81322046, 81302698, 81473137) and Shanghai Rising-Star Program (13QA1402300).

References

1. Changeux JP (2013) The concept of allosteric modulation: an overview. Drug Discov Today Technol 10:e223–e228
2. Eisenberg D, Marcotte EM, Xenarios I, Yeates TO (2000) Protein function in the post-genomic era. Nature 405:823–826
3. Kar G, Keskin O, Gursoy A, Nussinov R (2010) Allostery and population shift in drug discovery. Curr Opin Pharmacol 10:715–722
4. Boehr DD, Nussinov R, Wright PE (2009) The role of dynamic conformational ensembles in biomolecular recognition. Nat Chem Biol 5:789–796
5. Motlagh HN, Wrabl JO, Li J, Hilser VJ (2014) The ensemble nature of allostery. Nature 508:331–339
6. Csermely P, Palotai R, Nussinov R (2010) Induced fit, conformational selection and independent dynamic segments: an extended view of binding events. Trends Biochem Sci 35:539–546
7. Sinha N, Nussinov R (2001) Point mutations and sequence variability in proteins: redistributions of preexisting populations. Proc Natl Acad Sci U S A 98:3139–3144

8. Goodey NM, Benkovic SJ (2008) Allosteric regulation and catalysis emerge via a common route. Nat Chem Biol 4:474–482
9. Nussinov R, Tsai CJ (2013) Allostery in disease and in drug discovery. Cell 153:293–305
10. Nussinov R, Tsai CJ (2014) Unraveling structural mechanisms of allosteric drug action. Trends Pharmacol Sci 35:256–264
11. Christopoulos A (2002) Allosteric binding sites on cell-surface receptors: novel targets for drug discovery. Nat Rev Drug Discov 1:198–210
12. Szilágyi A, Nussinov R, Csermely P (2013) Allo-network drugs: extension of the allosteric drug concept to protein- protein interaction and signaling networks. Curr Top Med Chem 13:64–77
13. Cowan-Jacob SW, Jahnke W, Knapp S (2014) Novel approaches for targeting kinases: allosteric inhibition, allosteric activation and pseudokinases. Future Med Chem 6:541–561
14. Nussinov R, Tsai C-J (2014) The design of covalent allosteric drugs. Annu Rev Pharmacol Toxicol 55:249–267
15. Yang J-S, Seo SW, Jang S et al (2012) Rational engineering of enzyme allosteric regulation through sequence evolution analysis. PLoS Comput Biol 8, e1002612
16. Ma B, Elkayam T, Wolfson H, Nussinov R (2003) Protein-protein interactions: structurally conserved residues distinguish between binding sites and exposed protein surfaces. Proc Natl Acad Sci U S A 100:5772–5777
17. Fang Z, Grütter C, Rauh D (2013) Strategies for the selective regulation of kinases with allosteric modulators: exploiting exclusive structural features. ACS Chem Biol 8:58–70
18. Kenakin T, Miller LJ (2010) Seven transmembrane receptors as shapeshifting proteins: the impact of allosteric modulation and functional selectivity on new drug discovery. Pharmacol Rev 62:265–304
19. Nussinov R, Tsai C (2012) The different ways through which specificity works in orthosteric and allosteric drugs. Curr Pharm Des 18:1311–1316
20. Zhang J, Adrián FJ, Jahnke W et al (2010) Targeting Bcr-Abl by combining allosteric with ATP-binding-site inhibitors. Nature 463:501–506
21. Müller CE, Schiedel AC, Baqi Y (2012) Allosteric modulators of rhodopsin-like G protein-coupled receptors: opportunities in drug development. Pharmacol Ther 135:292–315
22. Lu S, Li S, Zhang J (2014) Harnessing allostery: a novel approach to drug discovery. Med Res Rev 34:1242–1285
23. Lu S, Huang W, Zhang J (2014) Recent computational advances in the identification of allosteric sites in proteins. Drug Discov Today 19:1595–1600
24. Li X, Chen Y, Lu S et al (2013) Toward an understanding of the sequence and structural basis of allosteric proteins. J Mol Graph Model 40:30–39
25. Wang Q, Zheng M, Huang Z et al (2012) Toward understanding the molecular basis for chemical allosteric modulator design. J Mol Graph Model 38:324–333
26. Huang W, Lu S, Huang Z et al (2013) Allosite: a method for predicting allosteric sites. Bioinformatics 29:2357–2359
27. Huang Z, Zhu L, Cao Y et al (2011) ASD: a comprehensive database of allosteric proteins and modulators. Nucleic Acids Res 39: D663–D669
28. Huang Z, Mou L, Shen Q et al (2014) ASD v2. 0: updated content and novel features focusing on allosteric regulation. Nucleic Acids Res 42: D510–D516
29. Le Guilloux V, Schmidtke P, Tuffery P (2009) Fpocket: an open source platform for ligand pocket detection. BMC Bioinformatics 10:168
30. Chang C-C, Lin C-J (2011) LIBSVM: a library for support vector machines. ACM Trans Intell Syst Technol 2:27
31. Bowman GR, Geissler PL (2012) Equilibrium fluctuations of a single folded protein reveal a multitude of potential cryptic allosteric sites. Proc Natl Acad Sci U S A 109:11681–11686
32. Jacobs DJ, Livesay DR, Mottonen JM et al (2012) Ensemble properties of network rigidity reveal allosteric mechanisms. Methods Mol Biol 796:279–304
33. Villoutreix BO, Kuenemann MA, Poyet J-L et al (2014) Drug-like protein-protein interaction modulators: challenges and opportunities for drug discovery and chemical biology. Mol Inform 33:414–437

INDEX

Ilan Samish (ed.), *Computational Protein Design*, Methods in Molecular Biology, vol. 1529,
DOI 10.1007/978-1-4939-6637-0, © Springer Science+Business Media New York 2017

E

F

G

H

I

K

L

M

N

P

Q

R

S

T

W

X