LNES
LECTURE NOTES IN EARTH SCIENCES
106

Paul D. Bons
Daniel Koehn
Mark W. Jessell
Editors

Microdynamics Simulation

EXTRA
MATERIALS
extras.springer.com

Springer

Paul D. D. Bons · Daniel Koehn ·
Mark W. Jessell (Eds.)

Microdynamics Simulation

With 185 Figures

Springer

Paul D. D. Bons
Eberhard Karls Universität
Institut für Geowissenschaften
Fachbereich Mineralogie und
Geodynamik
10 Sigwartstrasse
Tübingen 72076
Germany
paul.bons@uni-tuebingen.de

Mark W. Jessell
IRD LMTG UMR 5563
14 avenue Edouard Belin
Toulouse, France 31400
mjessell@lmtg.obs-mip.fr

Daniel Koehn
Universität Mainz
FB Geowissenschaften
Geographisches Institut
21 Johann-Joachim-Becher-Weg
Mainz 55099
Germany
koehn@uni-mainz.de

"For all Lecture Notes in Earth Sciences published till now please see final pages of the book"

ISBN: 978-3-540-25522-2 e-ISBN: 978-3-540-44793-1

Lecture Notes in Earth Sciences ISSN: 0930-0317

Library of Congress Control Number: 2007937514

Cover design: WMXDesign GmbH

Printed on acid-free paper

9 8 7 6 5 4 3 2 1

springer.com

Preface

Paul Bons, Mark Jessell and Daniel Koehn

From the first observations of microstructures, geologists have tried to model and simulate their formation. Modelling and simulation are closely related terms, often used as synonyms. However, there is a difference. According to the Apple-Macintosh electronic dictionary, a *model* is *"a simplified description, especially a mathematical one, of a system or process, to assist calculations and predictions"*. The same dictionary defines *simulation* as the *"imitation of the appearance or character of [something]"*. As a special case it also lists *"product of a computer model"*. A *model* is therefore a theoretical abstraction, whereas the *simulation* is the actual application of the model to a specific case. For example, a rigid ellipsoid in a deforming homogeneous viscous matrix (Fig. 1b) can be a *model* for a porphyroblast or porphyroclast in a deforming rock (Fig. 1a) (Ghosh and Ramberg 1976). Applying and running the model for a specific case would be a *simulation* (Fig. 1c) (e.g. Bons et al. 1997, see also Ch. 4.7). We used the term *simulation* in the title of this book, because we are mainly concerned with the numerical implementation of models as a means to exploring their validity, not so much with the models themselves, which are amply dealt with in such works as Passchier and Trouw (2005).

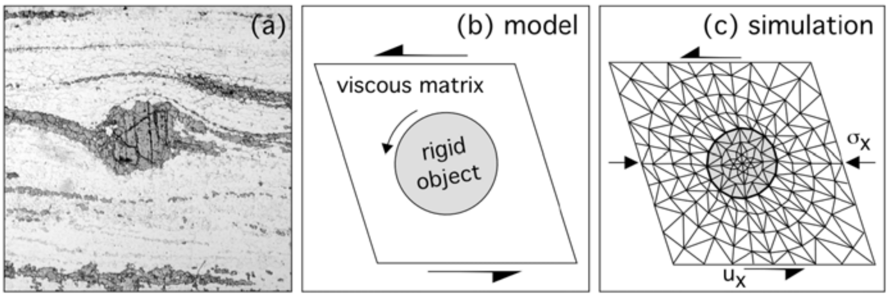


Fig. 1. (a) δ-shaped hornblende porphyroclast in a sinistral shear zone from Hidden Valley, South Australia. Width of view about 5 mm. **(b)** The model for the development of porphyroclast structures, such as the δ-clast, is that the porphyroclast rotates as a rigid object in a deforming ductile matrix. **(c)** To study the behaviour of porphyroclasts, the model can be implemented into, for example, a finite element program to run simulations

The purpose of this book is to provide an overview of the possibilities of using numerical techniques to study microstructures in rocks. Although wide-ranging, we cannot cover every technique that was ever applied to a microstructural problem. However we hope at least to demonstrate what is currently possible, and inspire users to investigate the field further if their particular interest is not currently met. In addition most of the numerical experiments we present may be repeated using the accompanying CD-ROM, so that the book can be used as a companion to an undergraduate or graduate level course on microstructures.

Chapter 1 gives a brief overview of the historical development of numerical simulation techniques as applied to grain scale phenomena in rocks, and discusses the principal constraints on applying numerical simulations to microstructural issues

There is no single way to simulate microstructural evolution. Some processes are best simulated using one method (e.g. Finite Elements), while other processes require other simulation methods, and many processes can in fact be adequately simulated by a whole range of techniques. Chapter 2 therefore gives an overview of the main numerical methods that are or can be used. This chapter is intended for those readers with no previous knowledge of numerical simulation methods, or only a subset of them.

A range of microstructural processes is presented in Chap. 3, which is process oriented. The theory of how each process may potentially affect a microstructure is briefly described followed by an overview of the way in which the process has been simulated numerically.

Based on Chap. 3, Chap. 4 gives some examples of the application of numerical simulation to particular problems. This chapter emphasises the need to combine different concurrent processes to simulate what happens in nature. Many highly sophisticated programs have been written to simulate single processes. However, these processes rarely operate in isolation in nature, and one therefore needs to be able to also simulate the interaction between different processes.

Many programs have been developed to simulate microdynamic processes. However, few of these are publicly available as these programs are usually developed for specific research purposes and remain with the authors. There is currently only one general package, called ***Elle***[1] that bundles a variety of simulation software for microdynamic modelling in earth sciences.

Elle is open source software that is developed by an informal consortium of earth scientists, many of whom have contributed to this book. The main advantage of *Elle* is that it is actually a growing collection of codes for individual processes, which allows the user to combine multiple processes in a single simulation. Where possible, *Elle* software is used for the examples presented in this book. “EXPERIMENT” at the end of a figure legend indicates that the figure can be replicated by running one of the example scripts on the CD-ROM (Appendix B). Thus, with the attached CD-ROM, the reader can run the examples exactly as they are shown in the book. As a next step, the reader can start to modify input parameters or data files to learn both how the simulation works and how particular processes act on a microstructure. The Appendices provide all the information needed to run *Elle*.

The open source *Elle* software is constantly being developed by researchers from all over the world. The software on the CD-ROM is represents the state of the code at the time of printing of this book. The latest version of *Elle* can be found on http://www.microstructure.info/elle. This web site also provides answers to frequently asked questions, fixes to the inevitable bugs and other information material.

[1] The *Elle* project started in the Department of Earth and Planetary Sciences of Monash University (Melbourne, Australia) during a series of weekly coffee-with-cakes meetings in the early nineties. The (over) ambitious idea was to develop a “super model” that would be able to model everything (metamorphism, deformation, the lot!). The “super model” was quickly baptised “Elle” after a then famous Australian super model by the name of “Elle”. We are indebted to Marlina Elburg to have come up with such a superb, easy to remember, but otherwise meaningless name for the project and software.

Contents

Contributing authors XVII
Acknowledgements XXI

1 Introduction 1
Paul D. Bons, Mark W. Jessell and Daniel Koehn
1.1 Historical background 4
1.1.1 Simulation of lattice-preferred orientations 5
1.1.2 Simulation of crystal growth in veins 8
1.1.3 Simulation of rigid objects 8
1.2 Why numerical simulation of microstructures? 9
1.3 Time and length scaling, resolution 11
1.4 Numerical opportunities and challenges 12

2 Numerical methods 15
Paul D. Bons, Daniel Koehn and Mark W. Jessell
2.1 Introduction 15
2.1.1 Data Structure 15
2.1.2 Variables and parameters 18
2.1.3 Equations 18
2.1.4 Boundary conditions 19
2.2 Lattice data structures: Monte Carlo, Ising, Potts, etc. 20
2.2.1 Introduction to Monte Carlo Principle 20
2.2.2 The Ising model 21
2.2.3 Potts models 25
2.2.4 Cellular Automata 27
2.2.5 Lattice-Gas and Lattice-Boltzmann method 28
2.3 Boundary models 33
2.4 Finite Difference method 37
2.4.1 Principles 37
2.4.2 Explicit Finite Difference method 39
2.4.3 Implicit Finite Difference method 45
2.4.4 The Crank-Nicholson scheme 47
2.4.5 Final remarks on the Finite Difference method 47
2.5 Finite Element method 50
2.6 Phase Field modelling 54
2.7 Molecular and Dislocation Dynamics 57
2.7.1 Molecular Dynamics 57
2.7.2 Modelling of dislocations 59
2.8 Lattice-Spring models 64
2.8.1 Normal Force model 65
2.8.2 Normal and Angular Force models 66
2.8.3 Elastic Beam models 68

2.8.4 Discrete Element models with contact forces 69
2.8.5 Fracturing or failure 71

3 Microprocess Simulations.. 75
Daniel Koehn, Mark W. Jessell and Paul D. Bons

3.1 Introduction to *Elle* 77
Lynn Evans, Mark W. Jessell, Paul D. Bons and Daniel Koehn
3.1.1 The *Elle* Project.. 77
3.1.2 The *Elle* Data Structure.. 78
3.1.3 Base Libraries.. 83
3.1.4 Precompiled Binaries 84
3.1.5 The *Elle* Data File 84
3.1.6 Experiment Scripts.. 84
3.1.7 Microprocess Simulation 85

3.2 Garnet - biotite cation exchange.. 87
Youngdo Park, Lynn Evans, Dal Park and Jin-Han Ree
3.2.1 Phenomenological observation... 87
3.2.2 Natural example 87
3.2.3 Inferred processes.. 89
3.2.4 Driving forces, controls and governing equations 89
3.2.5 Possible and actual simulation techniques.. 90
3.2.6 Implementation in *Elle*.. 92
3.2.7 Example *Elle* Run: garnet-biotite cation exchange............................ 94

3.3 Subgrain growth Potts model 97
Sandra Piazolo and Mark W. Jessell
3.3.1 Phenomenological observations... 97
3.3.2 Natural examples and experiments 97
3.3.3 Inferred processes.. 99
3.3.4 Driving forces, controls and governing equations 99
3.3.5 Possible and actual simulation techniques.. 100
3.3.6 Implementation in *Elle*.. 100
3.3.7 Example *Elle* Runs... 101

3.4 Nucleation and subgrain formation.. 105
Sandra Piazolo and Mark W. Jessell
3.4.1 Phenomenological observations... 105
3.4.2 Natural examples.. 106
3.4.3 Inferred processes.. 108
3.4.4 Driving forces, controls and governing equations 109
3.4.5 Possible and actual simulation techniques.. 110
3.4.6 Implementation in *Elle*.. 111
3.4.7 Example *Elle* Run.. 112

3.5 Grain boundary migration 115
Paul D. Bons, Mark W. Jessell and Jens Becker
3.5.1 Phenomenological observations 115
3.5.2 Natural examples 115
3.5.3 Inferred processes 116
3.5.4 Driving forces, controls and governing equations 117
3.5.5 Possible and actual simulation techniques 117
3.5.6 Implementation in *Elle* 118
3.5.7 Example *Elle* Run: Grain growth 123
3.5.8 Alternative implementations in *Elle* 125

3.6 Vein microstructures 127
Paul D. Bons and Daniel Koehn
3.6.1 Phenomenological Observations 127
3.6.2 Modeling of vein microstructures 128

3.7 Melt processes during grain growth 133
Jens Becker and Paul D. Bons
3.7.1 Phenomenological observations 133
3.7.2 Natural examples 133
3.7.3 Analogue modeling 134
3.7.4 Driving forces 134
3.7.5 Possible and actual techniques 135
3.7.6 Implementation in *Elle* 135
3.7.7 Example in *Elle* 136

3.8 *Basil*: stress and deformation in a viscous material 139
Greg Houseman, Terence Barr and Lynn Evans
3.8.1 Introduction 139
3.8.2 Formulation and assumptions 141
3.8.3 Numerical method 143
3.8.4 Porphyroblast example solution 147
3.8.5 Implementation in *Elle*: Growth and deformation of a porphyroblast 149
3.8.6 Conclusions 153

3.9 A hybrid FEM/Taylor Bishop Hill method 155
Mark W. Jessell
3.9.1 Introduction 155
3.9.2 Models of Lattice Rotation during Deformation 156
3.9.3 Implementation in *Elle* 158
3.9.4 Example *Elle* Run 159

3.10 Diffusion creep 161
John Wheeler and Judy Ford
3.10.1 Phenomenological observation 161
3.10.2 Natural examples 162
3.10.3 Inferred processes 162
3.10.4 Driving forces, controls and governing equations 162
3.10.5 Possible simulation techniques 166
3.10.6 Implementation in *DiffForm* 167
3.10.7 Simple example and description of input parameters 167

3.11 Fracturing 171
Daniel Koehn
3.11.1 Phenomenological observations 171
3.11.2 Natural examples 173
3.11.3 Inferred processes 173
3.11.4 Driving forces, controls and governing equations 173
3.11.5 Possible and actual simulation techniques 175
3.11.6 Implementation in *Elle* 176
3.11.7 Example *Elle* Runs 178

3.12 Fluid solid reactions 183
Daniel Koehn
3.12.1 Phenomenological observations 183
3.12.2 Natural examples 184
3.12.3 Inferred processes 185
3.12.4 Driving forces, controls and governing equations 185
3.12.5 Possible and actual simulation techniques 187
3.12.6 Implementation in *Elle* 187
3.12.7 Example *Elle* Runs 189

3.13 Solid-solid phase transitions and heat conduction 195
Till Sachau and Daniel Koehn
3.13.1 Phenomenological observations 195
3.13.2 Natural examples 196
3.13.3 Driving forces, equations describing the system 197
3.13.4 Possible simulation techniques 199
3.13.5 Implementation in *Elle* 199
3.13.6 Examples 204

4 Case studies and coupling of processes 207
Mark W. Jessell, Daniel Koehn and Paul D. Bons

4.1 Grain shapes and sizes during static grain coarsening in salt 209
Mark W. Jessell, Janos L. Urai and Oliver Schenk
4.1.1 Anisotropic grain growth 209
4.1.2 Implementation in *Elle* 210

4.1.3 Discussion 213
4.1.4 Conclusions 214

4.2 Dynamic recrystallisation and crystalline plasticity 215
Sandra Piazolo
4.2.1 Introduction 215
4.2.2 Theoretical background and implementation in *Elle* 215
4.2.3 Experimental Runs 220
4.2.4 Results 220
4.2.5 Discussion 223
4.2.6 Discussion of technique 223
4.2.7 Conclusions 224

4.3 Localisation of deformation 225
Mark W. Jessell, Esteban Siebert, Paul D. Bons, Lynn Evans and Sandra Piazolo
4.3.1 Introduction 225
4.3.2 Implementation in *Elle* 225
4.3.3 Experimental Results 227
4.3.4 Conclusions 230

4.4 Reactions and fracturing 231
Daniel Koehn
4.4.1 Overview of phenomenon 231
4.4.2 Implementation in *Elle* 232
4.4.3 Experimental runs 232

4.5 Combined phase transition and heat diffusion 237
Till Sachau and Daniel Koehn
4.5.1 Introduction 237
4.5.2 Implementation 238
4.5.3 Example runs 238

4.6 Visco-elastic and brittle deformation 241
Daniel Koehn and Till Sachau
4.6.1 Overview of phenomenon 241
4.6.2 Implementation in *Elle* 242
4.6.3 Experimental runs 244

4.7 Strain localisation and rigid-object kinematics 247
Scott E. Johnson
4.7.1 Introduction 247
4.7.2 Numerical experiments 249
4.7.3 Results 251
4.7.4 Discussion 252

4.8 Transient strain-rate partitioning during porphyroblast growth 255
Wesley G. Groome and Scott E. Johnson
4.8.1 Overview 255
4.8.2 Theoretical background and natural example 255
4.8.3 Implementation in *Elle* 257
4.8.4 Experiments and discussion 259
4.8.5 Conclusions 264

References 267

Appendices 291

Appendix A Installing *Elle* 293
Mark W. Jessell
A.1 Windows 293
A.1.1. Installing *Basil* and *Sybil* 294
A.2 Linux 294
A.2.1. Installing *Basil* and *Sybil* 295

Appendix B Example experiments 297
Mark W. Jessell
The *Experiment Launcher* 297
Experiment 1 - Diffusion 299
Experiment 2 - Fluid flow in a porous medium 301
Experiment 3 - Crystal growth from melt 302
Experiment 4 - Fracturing in granular aggregates 304
Experiment 5 - Cation exchange reactions 307
Experiment 6 - Subgrain growth 309
Experiment 7 - Grain growth 311
Experiment 8 - Evolution of a partial melt 313
Experiment 9 - Rigid porphyroblast growing in a deforming matrix 314
Experiment 10 - Lattice rotations 317
Experiment 11 - Boudinage 321
Experiment 12 - Dissolution grooves 323
Experiment 13 - Stylolites 325
Experiment 14 - Strain-rate partitioning during porphyroblasts growth 327
Experiment 15 - Anisotropic grain growth 330
Experiment 16 - Dynamic recrystallisation 332
Experiment 17 - Deformation localisation 335
Experiment 18 - Expanding inclusions 338
Experiment 19 - Mud cracks 340
Experiment 20 - Visco-elastic deformation and fractures 342
Experiment 21 - Strain localization and rigid object kinematics 345

Appendix C The *Elle* graphical user interface 349
Jens Becker
C.1 Introduction 349
C.2 The status bar 350
C.3 The menu-bar 350
C.3.1 The File Menu 350
C.3.2 The Run Menu 351
C.3.3 The Graphics Menu 351
C.3.4 The Data Menu 351
C.3.5 The Help Menu 351
C.4 The Run Options dialog 351
C.5 The Preferences Window 353
C.5.1 Flynn Preferences 353
C.5.2 Boundaries Preferences 354
C.5.3 Bnode Preferences 354
C.5.4 Unode Preferences 355
C.5.5 The colour map 356
C.6 The zooming window 357
C.7 The data-windows 358

Appendix D Different ways to run an *Elle* experiment 361
Mark W. Jessell
D.1 Via the *Experiment Launcher* 361
D.2 Direct launching of experiments 362
D.3 Developer mode (Windows and Linux) 362

Appendix E How to create multiple-process experiments 365
Lynn Evans
E.1 Shelle scripts/batch scripts 365
E.2 Shelle: An *Elle* Shell Script Generator (v1.24) 366
E.2.1 How to use the Shelle Script Generator 366
E.2.2 Notes on the Shelle Script 369
E.2.3 Debugging Shell Scripts 370

Appendix F *Sybil* - The *Basil* post-processor 371
Lynn Evans, Terence Barr and Greg Houseman
F.1 Introduction 371
F.2 Menu Options 372
F.2.1 File 372
F.2.2 Record 372
F.2.3 XYPlot 373
F.2.4 Profile 373
F.2.5 Arrow 374
F.2.6 Contour 375
F.2.7 Locate 375
F.2.8 Options 376

Appendix G How to use *Elle_Latte* 379
Till Sachau and Daniel Koehn
G.1 Introduction 379
G.2 The experiment interface for Windows and Linux users 380
G.2.1 Introduction 380
G.2.2 Installation notes 380
G.2.3 The main-window 381
G.2.4 The 'Files' tab 381
G.2.5 The New file and the Edit file window 382
G.2.6 The Statistics window 384
G.2.7 Remaining tabs 385
G.3 The command line interface for Windows and Linux developers 386

Appendix H Miscellaneous processes and utilities 389
Mark Jessell
H.1 Introduction 389
H.2 *elle_gbdiff* 389
H.3 *elle_diff* 389
H.4 *elle_met* 389
H.5 *elle_recovery* 390
H.6 *elle_viscosity* 390
H.7 *elle_expand* & *elle_pblast* 390
H.8 *elle_manuel* 390
H.9 *reposition* 390
H.10 *tidy* 391
H.11 *elle2poly, basil2elle, elle2oof, goof2elle* 391
H.12 *ppm2elle* 391
H.13 *ebsd2elle, elle2ebsd* 392
H.14 *plotaxes* 392

Appendix I How to Create an *Elle* file 393
Mark W. Jessell

Appendix J The *Elle* file format 395
Mark W. Jessell
J.1 Example *Elle* file 395
J.2 *Elle* file blocks 397

Index 401

Contributing authors

Editors

Bons, Paul D.	Mineralogie und Geodynamik, Institut für Geo-wissenschaften, Eberhard Karls Universität Tübingen. Sigwartstraße 10, 72076 Tübingen, Germany E-mail: paul.bons@uni-tuebingen.de
Jessell, Mark W.	IRD LMTG UMR 5563, 14 avenue Edouard Belin 31400 Toulouse, France E-mail: mjessell@lmtg.obs-mip.fr
Koehn, Daniel	Tectonophysik, Institut für Geowissenschaften, Johannes-Gutenberg Universität Mainz, Becher-weg 21, 55099 Mainz, Germany Email: Koehn@mail.uni-mainz.de

Contributing authors

Barr, Terence	AFEX International, 952 Echo Lane, Suite 300, Houston, TX 77024, USA Email: tdbarr@afexintl.com
Becker, Jens K.	Mineralogie und Geodynamik, Institut für Geowissenschaften, Eberhard Karls Univer-sität Tübingen. Sigwartstr. 10, 72076 Tübingen, Germany E-mail: becker@jkbecker.de
Evans, Lynn	School of Earth Sciences, University of Melbourne, VIC3010, Australia E-mail: laevans@unimelb.edu.au
Ford, Judy	Dept of Earth and Ocean Sciences, Jane Herdman Building, University of Liverpool, Liverpool L69 3GP, UK

Groome, Wesley G.	Department of Earth Sciences, Bryand Global Sciences Center, University of Maine, Orono, ME, 04469, USA E-mail: wesley.groome@umit.maine.edu
Houseman, Gregory A.	School of Earth and Environment, University of Leeds, Leeds LS2 9JT, UK Email: greg@earth.leeds.ac.uk
Johnson, Scott E.	Department of Earth Sciences, Bryand Global Sciences Center, University of Maine, Orono, ME, 04469, USA E-mail: johnsons@maine.edu
Park, Youngdo	Department of Earth and Environmental Sciences, Korea University, Anam-dong, Seongbuk-gu, Seoul 136-701, South Korea E-mail: youngdo.park@gmail.com
Park, Dal	Department of Earth and Environmental Sciences, Korea University, Anam-dong, Seongbuk-gu, Seoul 136-701, South Korea
Piazolo, Sandra	Department of Geology and Geochemistry, Stockholm University, Svante Arrhenius väg 8C, 106 91 Stockholm, Sweden Email: sandra.piazolo@geo.su.se
Ree, Jin-Han	Department of Earth and Environmental Sciences, Korea University, Anam-dong, Seongbuk-gu, Seoul 136-701, South Korea E-mail: reejh@korea.ac.kr
Sachau, Till	Tectonophysics. Institut für Geowissenschaften, University of Mainz, Becherweg 21, 55099 Mainz, Germany Email: Sachau@mail.uni-mainz.de
Schenk, O.	Geologie-Endogene Dynamik, Fachgruppe Geowissenschaften, Lochnerstraße 4-20, D-52056 Aachen, Germany

Siebert, Esteban	Department of Geology, Faculty of Physical and Mathematical Sciences, Plaza Ercilla #803, Santiago, Chili
Urai, Janos L.	Geologie-Endogene Dynamik, Fachgruppe Geowissenschaften, Lochnerstrasse 4-20, D-52056 Aachen, Germany Email: j.urai@ged.rwth-aachen.de
Wheeler, John	Dept of Earth and Ocean Sciences, Jane Herdman Building, University of Liverpool, Liverpool L69 3GP, UK Email: johnwh@liverpool.ac.uk

Acknowledgements

We thank the various funding agencies that have (willingly or reluctantly) funded the work on microdynamic modelling on which this book is based. In particular we would like to mention the Australian Research Council (ARC), the Deutsche Forschungsgemeinschaft (DFG), the USA National Science Foundation (NSF), and the European Union, through its Marie Curie Fellowship scheme.

The development of the code has been the work of the editors and most of the authors in this book, but in particular of Lynn Evans and Jens Becker, without whose technical expertise, and patience, we would not be where we are today.

Over the past 10 years we have held training courses and developer workshops on the numerical simulations of microstructures in Aachen, Hürtgenwald and Mainz (Germany), Leeds (UK), Estremoz (Portugal), Miglos and Toulouse (France), Pisa (Italy), Särghaua (Estonia), Orono, Maine (USA), and Cadaqués (Spain) where the participants of these workshops acted as willing guinea pigs for various versions of the code, and have stimulated its further development.

P. England, K. Stüwe, P. Molnar and other colleagues and students have provided assistance and motivation in the development of the *Basil* software package.

We would also like to thank the proof readers and software testers of this book, Lenka Baratoux, Marlina Elburg, Enrique Gómez-Rivas, Anne Peschler, and Kristjan Urtson, who have considerably reduced the number of internal inconsistencies, mistakes and bugs.

Finally we would like to acknowledge the use of several Open Source and Public Domain codes within the *Elle* platform:

wxWidgets	http://www.wxwidgets.org
Lesstif	http://www.lesstif.org
gnu scientific library	http://www.gnu.org/software/gsl
triangle	http://www.cs.cmu.edu/~quake/triangle.html
Generic Polygon Clipper	http://www.cs.man.ac.uk/~toby/alan/software/gpc.html
Graphics Gems	http://www.acm.org/tog/GraphicsGems/

1 Introduction

Paul D. Bons, Mark W. Jessell and Daniel Koehn

The complete arrangement of µm-mm scale structures or elements in a rock is alternatively called *microstructure*, *fabric*, or *texture*. These structures involve grain sizes, grain shapes, grain boundary shapes, the subgrain structure, distribution of mineral phases, lattice orientations, etc. (Fig. 1.1). There is, unfortunately, no consensus of what term to use for the whole arrangement of all these elements. For example, some use *texture* to only denote the lattice orientation distribution, ignoring the other elements of the arrangement. Others use *texture* in a wider sense, as the term *foam texture* indicates: a microstructure where the grains look like the bubbles in a foam. Most geologists mean roughly the same when they use the term *microstructure*: this is what you see in a thin section with an optical microscope. This is also how the term is mostly used in this book. However, materials scientists usually think of a much smaller scale when they use *microstructure*. Microstructures in rocks are important to geologists for two reasons: firstly, to find out the history of a particular rock, and secondly, to understand processes in rocks and to predict rock properties.

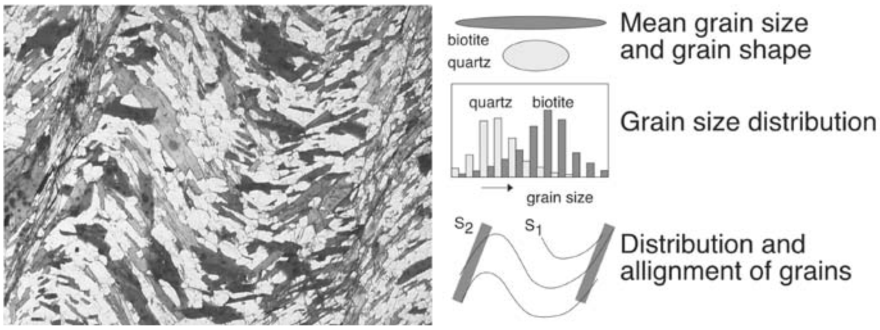


Fig. 1.1. Some microstructural features of a crenulated quartz-biotite schist, such as mean grain size and grain shape, grain size distribution and distribution and alignment of grains and mineral phases in foliations. Other possibly relevant features could be lattice-preferred orientation of quartz, chemical zonation patterns in the biotite, etc. (field of view circa 10 mm, PPL)

Geology started out as a mostly forensic science. For centuries, geologists have gone into the field to study outcrops and collected samples to look at them in more detail. For more than a century geologists have used petrographic microscopes to study the microstructure in thin sections. This microstructure is part of the "memory" of the rock, and it is our aim to read and decipher this memory in order to find out what happened to the rock millions or even billions of years ago. The trained geologist will use many indicators of past events, such as mineral parageneses to estimate metamorphic grade, overgrowth and intersection relationships to unravel sequences and order of events, and sense-of-shear indicators that reveal the kinetics of deformation in a deformed rock (Passchier and Trouw 2005). The method used is essentially an investigative or *deductive* one: "if one sees microstructure X it implies process or event Y happened".

The study of microstructures is also used in a predictive or *inductive* way. For example, microstructures in quartz indicate the operation of dislocation creep as the dominant deformation mechanism at amphibolite-facies metamorphic conditions. This can then be used to predict the rheological behaviour of the middle crust, which is input needed in large-scale geophysical simulations. Again we need to know the link between microstructure X and process or property Y.

In order to determine the link between X and Y, geologists rely on a body of knowledge and experience that has built up over the years. Because geological processes usually occur outside the scope of direct investigation, we often have to determine the link in an indirect experimental way, by analogy from other materials (e.g. metals and ceramics), and of course from theory. Increasingly, numerical simulations are used to investigate how microstructures form, and hence, to gain insight in what they mean.

The four key elements needed to explain how the earth deforms are (Fig 1.2):

1. The spatial (e.g. microstructures) and temporal (e.g. earthquake statistics) patterns seen in natural systems.
2. The boundary conditions acting on a system (e.g. temperature, stress, pressure, etc.)
3. Knowledge of the processes that occur in rocks and the mechanical response to the boundary conditions.
4. An understanding of how the three elements above couple.

It is widely accepted that to develop more complete theoretical models of rock deformation, we need to couple these elements more tightly, and numerical simulations provide a relatively new method for achieving that coupling. The numerical simulation of microstructures in geological materials has progressed in parallel with techniques developed in the wider materials science community (see Raabe 1998 for a comprehensive bibliography). In recent years there has been an upsurge in interest in this field due to the easy access to increasingly more powerful computers.

Equally important has been the introduction of new measurement techniques such as Orientation Imaging Microscopy (OIM) and Electron Backscatter Diffraction (EBSD) (Lloyd and Freeman 1991; Panozzo-Heilbronner and Pauli 1993; Pauli et al. 1996; Fueten 1997; Leiss *et al.* 2000; Fueten and Goodchild 2001; Bestman and Prior 2003).

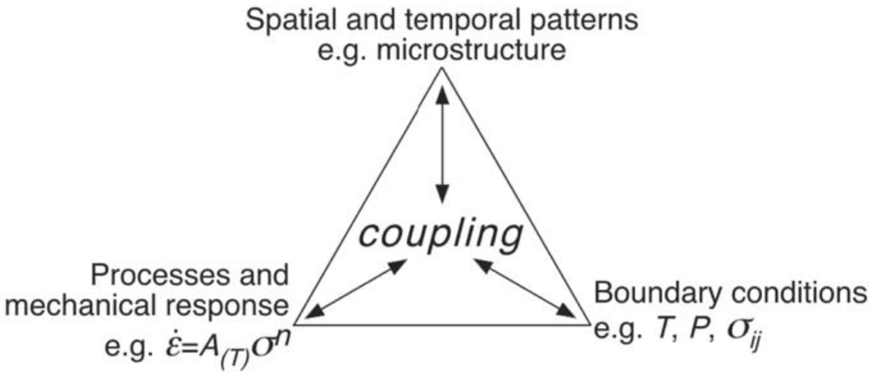

Spatial and temporal patterns
e.g. microstructure
coupling
Processes and mechanical response
e.g. $\dot{\varepsilon}=A_{(T)}\sigma^n$
Boundary conditions
e.g. T, P, σ_{ij}

Fig. 1.2. The fundamental controls on rock deformation

These techniques allow us to systematically characterize the grain boundary topologies and textures of rocks with relative ease and in much more detail (Trimby *et al.* 1998), thus provoking us to rethink what microstructural parameters we should use as indicators of specific processes. These techniques are in turn leading to the development of new in-situ experimental methods that allow us to characterise the sample microstructure in a variety of ways during the experiment, so that the coupling between processes and the mechanical response can be followed as a series of time lapse images of the sample (Offerman et al., 2002; Seward et al., 2004; Piazolo et al. 2004; Schenk and Urai, 2005). In this way the all-important temporal behaviour can be integrated into the interpretation, and together they provide much stronger constraints on our models.

The paradigm for understanding the mechanical behaviour of materials was for many years the Deformation Mechanism map (Frost and Ashby, 1983), which related the applied stress, the temperature of deformation and one or more microstructural parameters (such as average grain size) to predictions of the dominant deformation mechanism (in terms of accommodating the largest strain rate) for a given set of conditions. Unfortunately there are two fundamental limitations to this approach for geoscientists, who have to work with very long deformation histories:

1. these maps provide little insight into the non-equilibrium (non-steady state) behaviour, such as when the boundary conditions change, or when the material localizes strain, and
2. they effectively ignore the effects of other geological processes such as metamorphism, the role of fluids, or recrystallisation that modify the microstructure. These processes do not necessarily directly play a role in deforming the rock, but nevertheless modify the relative importance of different deformation mechanisms and the resulting mechanical response.

At the hand specimen scale, our understanding of how geological processes couple remains rudimentary, in no small part because of the inherent complexity of the system. If we want to develop flow laws applicable to a wide range of boundary conditions, we need to understand how grain-scale process couple with grain-scale patterns (i.e. microstructures) in order to modify mechanical behaviour. As a result of this complexity, the geological community understandably lags behind the wider materials science community in developing history dependant flow laws, where it is recognised that "Microstructure is the state variable of materials properties" (Gottstein *pers. comm.* 2005).

1.1 Historical background

Until recently, the simulation of microstructural development was quite limited. Physical simulations, or *experiments*, were of course carried out right from the beginning of investigations into microstructures (see the comprehensive review of the state of the art at the beginning of the 20th century in Paulcke, 1912). However, non-physical simulations were severely hampered by the limited calculation power before the advent of computers. The advent of computers made it possible to repeat the existing non-physical simulations, but more quickly, more accurately, with higher resolution and for larger data sets. This trend is illustrated with three examples of microstructures that have been simulated extensively for about a century: the development of crystallographic preferred orientations, constrained crystal growth in veins and the rotation of relatively rigid objects (e.g. porphyroclasts and porphyroblasts) during deformation.

1.1.1 Simulation of lattice-preferred orientations

The earliest microstructural numerical simulation studies in geology were focused on predicting the CPO patterns found in naturally deformed rocks. This work was based on metallurgical models that were first developed by Taylor (Taylor 1938), and which, with modifications, are still widely applied (Tóth *et al.* 1997). Lister and co-workers used a standard Taylor-Bishop-Hill (TBH) formulation to study the development of CPO in quartz and calcite rocks (Fig. 1.3a) (Lister *et al.* 1978; Lister and Paterson 1979; Lister & Hobbs 1980; Lister 1982). Other groups have since extended this work to a broader range of minerals and to allow refinements of the TBH scheme such as the relaxed-constraints (Ord 1988) and self-consistent approaches (Wenk *et al.* 1989a, 1989b; Canova 1994; Takeshita *et al.* 1999). Their work was able to demonstrate that, even though many minerals do not fulfil the requirement for the strict application of the Taylor model (due to the limited number of slip systems available) many of the key features of natural textures could be accounted for.

An alternative approach recognises that in some minerals, such as trigonal symmetry quartz at low temperatures, only one slip system may be readily activated. This necessitates incorporation of intra- and intergranular strain heterogeneity in simulations (Etchecopar 1977; Etchecopar and Vasseur 1987; Zhang et al. 1993, 1994a, 1996; Wilson & Zhang 1996; Wilson et al. 1996; Zhang and Wilson 1997a,b). These simulations can reproduce the behaviour of, for example, ice and low temperature quartz. Grain boundary sliding was shown to enhance texture development in model materials that allowed sliding interfaces between grains deforming with one slip system (Zhang et al. 1994b). The model of Ribe (1989) has been extended to simulate texture development in quartz when grain boundary sliding is the dominant deformation mechanism (Casey and McGrew 1999).

Under a wide range of crustal conditions, rocks deforming by crystalline plasticity are also modifying their microstructure as a result of dynamic recrystallisation processes (Urai *et al.* 1986; Hirth & Tullis 1992). Jessell drew upon experiments on cold worked copper (Kallend and Huang 1984) to support the assumption that the level of stored work is orientation dependent, and that the low crystal symmetry of the rock forming minerals would result in a larger anisotropy of stored work than is present in metals (Jessell 1988a,b; Jessell & Lister 1990). He developed a hybrid scheme that combined the Taylor code of Lister with a Monte Carlo simulation that simulated the evolution of textures in quartz polycrystals by iterating between small increments of lattice rotations and grain boundary migration and subgrain formation (Fig. 1.3a-b).

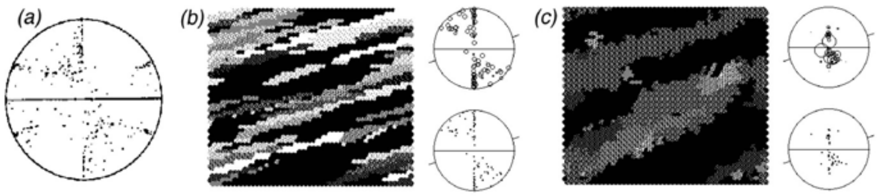


Fig. 1.3. Simulations of crystallographic preferred c-axes orientations in quartzite. **(a)** Classical Taylor calculation for simple shear up to a shear strain of three (Lister 1981). **(b)** Combination of a Taylor calculation for lattice reorientation and a Monte Carlo model for dynamic recrystallisation at relatively low temperature, after Jessell and Lister (1990). Shear strain (top to the right) is three. Stereoplots show orientations of c-axes. Dot size is proportional to grain size in upper plot. **(c)** Same as (b), but for a relatively high temperature with higher grain boundary mobility

The recrystallisation processes in these simulations were driven by a simple stored work term for each element in the Monte Carlo simulation, as well as the more normal boundary-energy derived neighbour relations (Anderson *et al.* 1984; Weaire & Rivier 1984). These simulations show that a good correspondence with natural textures could be explained by the coupling of lattice rotations and dynamic recrystallisation. Comparable hybrid lattice rotation/recrystallisation schemes have recently been developed that replace the Taylor-Bishop-Hill scheme entirely with Finite Element codes capable of heterogeneous intra-crystalline deformation (Radhakrishnan, et al. 1998; Raabe and Becker 2000; Bate 2001). Another approach to simulating the effects on texture of dynamic recrystallisation in rocks has been to build in a grain size weighting to simulate the changes in grain size that may result from dynamic recrystallisation driven by orientation dependent stored work terms (Takeshita *et al.* 1999).

This example highlights the importance of considering multiple and coupled processes in simulations. The coupled action of several processes is at the heart of the formation of microstructures, and therefore micro-gauges such as palaeo-stress indicators (Twiss 1977; Kohlstedt and Weathers 1980). However, the nature of coupling is often difficult to determine and numerical simulations that treat coupled processes are still in their infancy (Etchecopar & Vasseur 1987, Jessell & Lister 1990, Jessell et al. 2001; 2005).

Texture development is classically attributed to result from movement of dislocations (e.g. Wenk 1985). However, other deformation processes, such as dissolution-precipitation creep (Rutter 1976) may hypothetically also affect textures (Hippert 1994; Stallard & Shelley 1995; den Brok 1996). Recently, Bons & den Brok (2000) explored this possibility with a numerical simulation. They showed that reaction-controlled dissolution-precipitation creep, coupled with grain rotation can also lead to strong lattice-preferred orientation, similar to those described above. Their simulation used a constant stress approach, the opposite to the more commonly used constant strain rate or Taylor approach, and which may over-estimate the strain rate heterogeneity and rotation rates of grains.

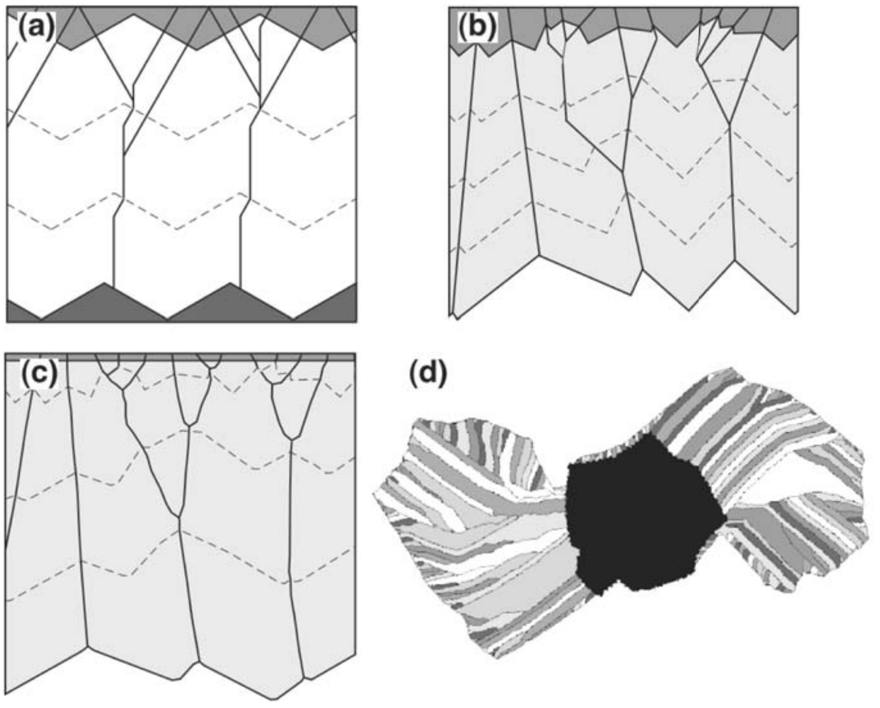


Fig. 1.4. Two-dimensional simulations of crystal growth. **(a)** Isotropic growth simulation of crystals in a sequentially opening crack as used by Urai et al. (1991). Grain boundaries are perpendicular to the growth front. **(b)** The *FACET* growth simulation of Zhang and Adams (2002) for growth of a number of seeds into an open fluid-filled crack. All grain surfaces in contact with the fluid are facets. **(c)** *Vein Growth* simulation of Bons (2001) for growth of a number of seeds into an open fluid-filled crack. Grain surfaces in contact with the fluid can have any crystallographic orientation, each with its own growth rate. Slow orientations become facets. **(d)** Fibrous fringes developed adjacent to a rigid object, simulated by Koehn et al. (2000) with *Fringe Growth*, an adapted version of *Vein Growth*

1.1.2 Simulation of crystal growth in veins

The second example is that of crystals growing side-by-side on the surface of a crack, which are constrained in their growth by their neighbours (Durney and Ramsay 1973; Bons 2000; Oliver and Bons 2001; Passchier and Trouw 2005; also see Ch. 3.6). This growth competition was already studied by Taber (1916) and Mügge (1928). They had to use pen, paper and ruler to model the progressive shape of crystals that impinge on each other. Over 80 years later, Urai et al. (1991) essentially used the same model and simulation technique, but using a computer drawing program instead of pen and paper (Fig. 1.4a). Zhang and Adams (2002) implemented the same basic model in the computer program *FACET*. This program makes it possible to quickly simulate competitive growth of large numbers of facetted crystals (Fig. 1.4b). However, a simulation by FACET could in principle be done by hand with pen and paper (although it would be an extremely cumbersome and time-consuming task!). Bons (2001) developed a slightly different model for competitive growth with the program *Vein Growth*. (Hilgers et al. 2001; Nollet 2005). Non-rational crystal surfaces are allowed in this model and a continuous growth function is assigned to a "mineral", describing the growth velocity as a function of crystal surface orientation relative to crystal lattice orientation of the crystal (Fig. 1.4c). Koehn et al. (2000, 2001) took the model of Bons (2001) a step further and used it to simulate the formation of fibrous fringes at pyrite crystals, involving a complex opening history of the available space for crystals to grow in (Fig. 1.4d). Such a simulation would be virtually impossible without a computer.

1.1.3 Simulation of rigid objects

The third example is that of rotating porphyroblasts and porphyroclasts. It is generally accepted that equant, relatively rigid porphyroblasts (e.g. garnets) or porphyroclasts (e.g. feldspar crystals) rotate relative to their matrix during non-coaxial deformation, leading to structures such as spiral inclusion trails and delta-clasts, respectively (Ghosh and Ramberg 1976; Schoneveld 1977; Ferguson 1979; Passchier and Simpson 1986; Hanmer and Passchier 1991). Since Ghosh and Ramberg (1976), the popular model for this rotation behaviour is that of Jeffery (1922) for a rigid elliptical inclusion in a viscous matrix. This model was widely accepted until recently, even though observations on natural microstructures often seem to contradict it (Pennacchioni 2000; ten Grotenhuis et al. 2002). The analytical model of Jeffery (1922) could be used to determine the rotation rate and finite rotation of an object during deformation or after a certain amount of deformation. With the advent of computer-based simulations, it also became possible to simulate the evolution of structures, like wings, around the rigid objects (Fig. 1.5). Using the full potential of computers, a number of authors have investigated the effects of various effects that may influence the process, such as slip between porphyroblast/-clast and matrix, localisation of deformation, non-linear behaviour of the matrix, etc. (Etchecopar and Malavieille 1987; Bjørnerud 1989; Bjørnerud and Zhang 1994, 1995; Bons et al. 1997; Kenkmann and Dresen 1998; Pennacchioni et al. 2000; Tenczer et al. 2001; Schmid and Podladchikov 2005; see also sections 3.8 and 4.7).

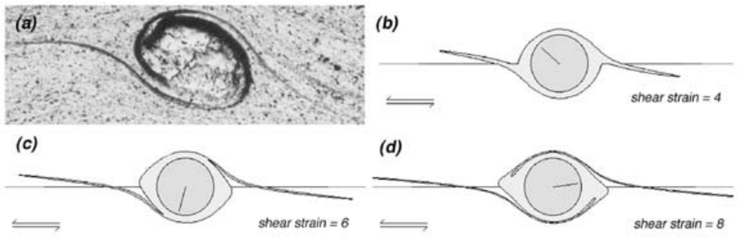


Fig. 1.5. **(a)** δ-clast (~1 mm in size) of camphor in an octachloropropane matrix created in an in-situ ring-shear experiment by ten Brink (1996). **(b-d)** Numerical simulation of the developement of a delta-clast using an analytical solution for simple-shear flow around a rigid cylinder, as was used by Passchier et al. (1993)

The recent date of publication in most of the examples cited above shows that the full implementation of computers in the simulation of microstructures is only really beginning and actually lagging behind some other disciplines in earth sciences, such as large-scale geophysical modelling and hydro-geology. However, this book will show that this field of geology is rapidly catching up and numerical simulations are being developed and implemented for a wide variety of microstructures and microstructural processes.

1.2 Why numerical simulation of microstructures?

For geologists there are three aspects of geological processes in nature that hamper their experimental investigation:

1. time scale,
2. length scale, and
3. boundary conditions (pressure, temperature).

Therefore, geologists can usually not reproduce the same conditions in the laboratory as those that apply to nature (Fig. 1.6). Geological processes can be extremely slow relative to human time scales. For example, a pressure fringe system (see Ch. 3.6) may take tens of millions of years to develop (Mueller et al., 2000). Ductile deformation strain rates in nature are in the order of 10^{-12} s^{-1} or slower. Experiments, on the other hand, cannot take longer than a few days to weeks at the most, meaning that strain rates have to be about 10^{-8} s^{-1} or faster. Therefore there is normally a discrepancy of four orders or more between natural and experimental rates for ductile deformation (Fig 1.6b). For brittle deformation, with time scales in the order of seconds this is of course less of an issue.

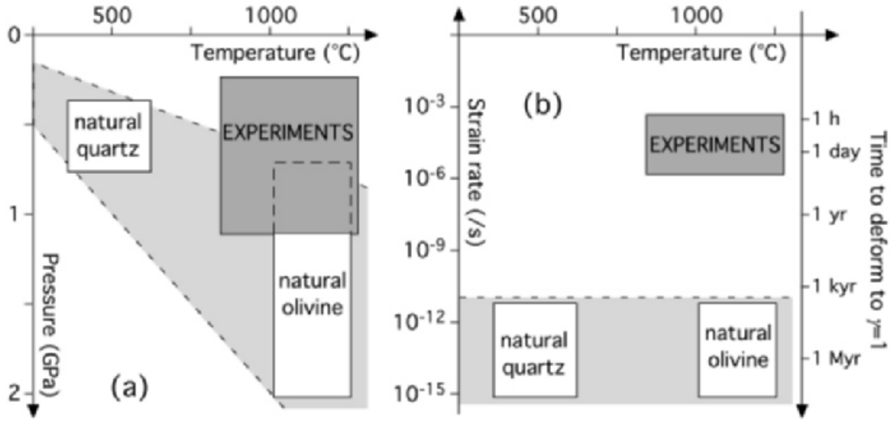


Fig. 1.6. Comparison between natural and experimental conditions (Paterson's/ Grigg's rig) of ductile rock deformation. **(a)** Temperature versus confining pressure and **(b)** temperature versus strain rate. Scale on the right shows time to achieve a shear strain of one, which is weeks at the most in the laboratory, but at least hundreds of thousands of years in nature. Grey band represents natural conditions

Length scales of geological systems reach up to the 100-1000 km scale (e.g. the San Andreas Fault or mantle convection), while laboratory samples are normally restricted to the 1-10 cm scale at the most. This often means that processes that involve a range of scales, especially those much larger than the grain scale, are difficult to simulate in laboratory experiments.

Crustal boundary conditions, such as pressure and temperature are more easily achieved in the laboratory, where pressures up to about one GPa and temperatures well over a 1000°C can be reached for deformation experiments in equipment like the Paterson gas apparatus or a Grigg's rig.

Even higher pressures can be reached in, for example, a diamond-anvil cell, but at the cost of accuracy in the application of deformation or the measurement of differential stresses.

To circumvent the problem of slow natural strain rates, many experimentalists increase both strain rate and temperature. For example, to simulate quartz deformation in the middle crust, one usually performs experiments on quartz at temperatures above 800°C. This may be permissible for pure quartz, but one runs into problems when one wants to study the behaviour of, for example, a quartz + mica rock at greenschist facies conditions. Increasing the temperature would change the metamorphic conditions and such a rock would melt above ~750°. This inherent problem in the study of rocks is one of the reasons to do numerical simulations. Numerical simulations are not constrained in time, temperature, pressure or any other parameter, and can just as easily be applied to experimental conditions as to natural conditions. If the simulation of deformation and metamorphic processes in real rocks under natural conditions cannot be done in the laboratory, one has to resort to numerical simulations.

1.3 Time and length scaling, resolution

In any numerical simulation, we need to start with a conceptual model of how a particular process behaves, translate that model into a series of mathematical equations, and then define how the model space is going to be divided up. For the simulation of microstructural evolution, this model space will typically consist of one, two or three spatial dimensions plus the dimension of time. The spatial and temporal discretisations need to be sufficiently fine to capture the detail of the process that is being simulated. Geological processes cover an enormous range of time and spatial scales, from 10^{-9} s and 10^{-9} m for atomic scale processes such as diffusion and glide of individual dislocations, up to 10^{15} s and 10^{6} m for global geodynamic processes such as mantle convection, thus it is unfortunately necessary to limit these scales to a particular range, in order to make the problem tractable. At any given scale, the boundary conditions for the experiment must be derived from the study of larger scale problems and the fundamental properties must be derived from the study of smaller scale processes (fig. 1.7). The limitations to this approach can be seen when the material properties that are used as inputs to the large-scale simulations vary as a function of time, and conversely when the boundary conditions that control the long term evolution of the small scale structures vary. One of the great challenges for numerical simulation over the next 10 years is to develop methodologies where we can account for not only the coupling of processes, but also of the coupling between scales, so that the truly dynamic nature of geological processes can be investigated. For example we cannot understand grain size evolution in rocks if we do not include subgrain evolution models, which in turn depend on our understanding of multiple dislocation interactions. Similarly we cannot understand lower crustal and mantle rheologies if we do not understand under what conditions (or if) a material will localise strain, and how quickly it responds to changes in boundary conditions. Multi-scale numerical simulations, while technically extremely challenging, provide a methodology for understanding these systems (Zbib and Diaz de la Rubia, 2002; Cordier, 2005).

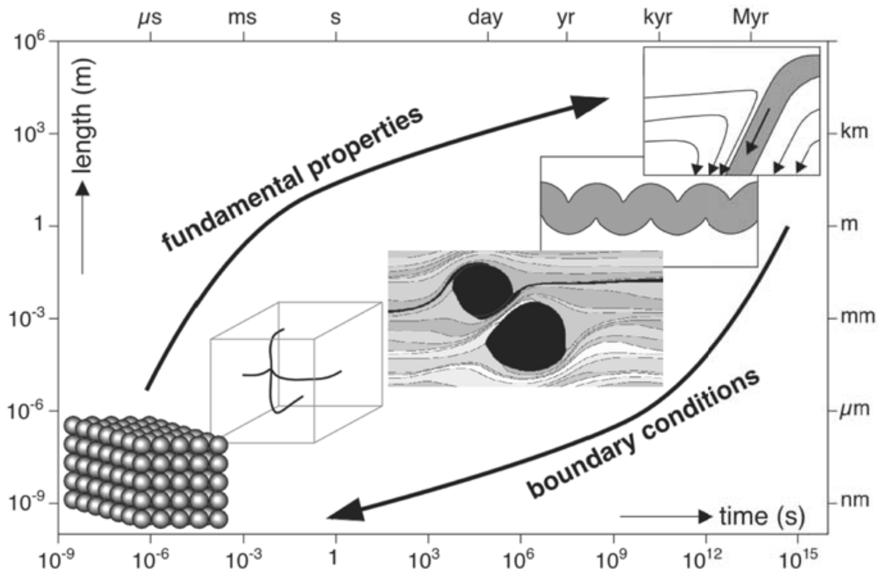


Fig. 1.7. Range of time and spatial scales of geological numerical simulations, from the atom and dislocation scale, via microstructures (the principal subject of this book) all the way up to the scale of the crust and mantle

1.4 Numerical opportunities and challenges

We should not underestimate the technical difficulties in performing numerical simulations, or properly validating them. Apart from having analytical solutions for a few cases with simple boundary conditions and geometries, for single processes such as diffusion, validation of numerical simulations remains problematic. In large scale geodynamic simulations this is being addressed by performing controlled experiments using a variety of different codes (Buiter et al., 2006) and we look forward to a future where 'competitor' codes exist to *Elle*, so similar comparisons can be made. These will however only test the numerical correctness of the code, not the assumptions in the underlying conceptual models, and for that the most promising avenue for research is the performance of integrated numerical and in-situ experiments. For example, Bons et al. (2001) performed in-situ experiments of grain growth in octachloropropane and compared the results with numerical simulations of anisotropic grain growth. In-situ experiments can now also be carried out inside an electron microscope and first steps have been made towards in-situ experiments in synchrotrons, providing challenging new opportunities to callibrate and test numerical simulations (e.g. Piazolo et al. 2004; Schmidt et al. 2004)

In-situ experiments are still limited in their ability to run under a wide range of boundary conditions, and this is one of the reasons why numerical simulations of rock deformation offers an important parallel methodology: although numerical simulations have other limitations, they can often be performed over a wide range of boundary conditions, including non-equilibrium behaviour (Aharonov and Sparks, 1999; Raabe and Becker, 2000; Jessell et al., 2005). If properly calibrated, simulations, in particular those that investigate coupled processes and which incorporate a geometric and topological description of the microstructure (Ford et al., 2002), allow a more complete scaling to natural conditions, since each individual process can be individually scaled, and the coupling that controls mechanical behaviour is a direct outcome of the simulations.

One of the fundamental couplings in the microdynamic evolution in rocks is that between mechanical and mineralogical-chemical processes. Whereas many simulations exist of the microstructural evolution as a result of mechanical processes (Ch. 1.1), few have so far undertaken to simulate the evolution of a "thin section picture" by metamorphic processes (Foster 1999; Park et al. 2003). This is despite the huge progress that has been made in thermodynamic modelling of mineral reactions, with programs such as THERMOCALC (Powell and Holland 1985; Powell et al. 1998) and PERPLEX (Connolly 1990). One of the main challenges ahead is to combine such thermodynamic and mineralogical/crystallographic models with mechanical models to be finally able to simulate the evolution of rocks during tectonic-metamorphic events.

First approaches in this direction include mechanical-chemical coupling during pressure-solution processes (Ghoussoub and Leroy, 2001; Ford et al., 2002; Koehn et al., 2003; 2006, Ch. 3.12), reactions and fracturing (Jamtveit et al., 2000; Malthe-Sørenssen et al., 2006) and mineral phase transitions that include large volume changes (Ch. 3.13). One of the main challenges in these systems is their non-linearity in time. Different processes act on very different time scales so that the program itself has to choose model steps for the coupling. For example we may have to deform a potential rock for a long time to build up stresses that are high enough for a phase transition to start, but once the transition proceeds it may happen very quickly. Therefore the initial time steps should be relatively large until the transition starts and become very small during the transition. The differences in time can easily span several orders of magnitude.

As our understanding of the relationship between the processes of rock deformation and spatial and temporal patterns improves, so does our ability to interpret these patterns in terms of the prior evolution of the system. Structural geologists have traditionally relied heavily on micro-gauges to interpret the strain, and sometimes the stress conditions at the time of definition (Passchier and Trouw, 1996). As in the engineering sciences, we are now able to apply our understanding to investigations of other characteristics of minerals and rocks such as surface energy and palaeo-rheology, so that we can test our laboratory-based determinations against natural examples (Kenis et al., 2005). At the same time as developing these new tools, we need to continue to apply all of our currently available tools to the study of major tectonic structures such as active faults and volcanoes, so that we continue to improve our ability to predict the behaviour of specific natural hazards.

2 Numerical methods

Paul D. Bons, Daniel Koehn and Mark W. Jessell

2.1 Introduction

A model is a way to describe and usually simplify some process(es) or phenomenon one tries to understand. This can be constructed in words, as a physical object (e.g. a model of the Alps made of plaster), but usually in a mathematical way (by a set of equations). Sometimes, the equations and the system we want to apply them on (together the "model") are relatively simple and the equations can be solved on the back of an envelope. In that case there is an analytical solution. In many cases, however, the model is too complex to solve all the equations on a piece of paper, and we have to resort to computers to solve the equations.

Microstructures are typically complex structures and it is almost always impossible to solve the equations analytically when the goal is to obtain local information, rather than bulk statistics. Microdynamic simulation therefore employs a whole variety of numerical methods, which are methods that use a computer to solve equations. In this chapter, the main numerical methods that are used for microdynamic simulation are presented. These methods are not specific to microdynamic simulation, but are employed in all fields of science. Here, we focus specifically on those methods that are used in the next chapters for the simulation of a variety of microdynamic processes. Those readers already acquainted with numerical methods can skip this chapter, and go straight to the next chapters.

All (numerical) models share some basic ingredients that can be grouped into four categories:

2.1.1 Data Structure

We use the term Data Structure here to mean the way the system to be simulated is described. For microdynamic models, which typically aim to simulate a granular aggregate, the two basic frameworks are:

Lattice data structures, where the system is mapped on a lattice, like the pixels in a digital image. A lattice can be square, which is numerically the simplest case and is easily displayed on a screen, which has square pixels (Fig. 2.1a). Other lattices are also possible, such as triangular or hexagonal lattices (Fig. 2.1b). The resolution of the model is directly related to the density of the lattice. Potts models, Cellular Automata, Lattice-Spring models and Phase Field models are all based on lattice frameworks.

Element data structures, where the system is described by elements. Such elements can be points, line segments, or polygons (Fig. 2.1c). The advantage of using elements is that one usually needs far fewer elements to describe a system accurately than lattice nodes in a lattice framework. Finite Element models and boundary models typically use elements to describe a system.

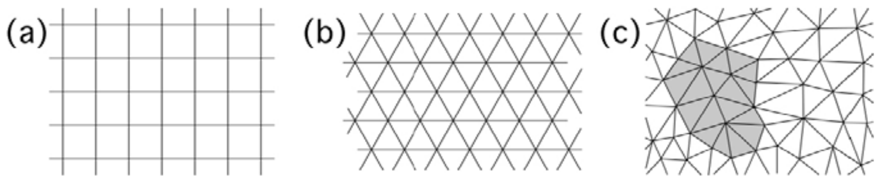


Fig. 2.1. Examples of different lattice types. **(a)** Regular square grid. **(b)** Regular triangular grid where each triangle has the same size and shape. **(c)** Grid with triangular elements, where each triangle has a different size and shape. With such a grid one can also define polygonal elements, like the one shaded grey. Such grids are commonly used for Finite Element models

There are many ways in which a system can be described by elements, even if one uses the same basic element shape, such as triangles only (Fig. 2.1c). The requirement for the shape of elements for practically all numerical methods are:

- The elements should be as equidimensional as possible. Elements with a very large length/width ration are to be avoided.
- The elements are preferably convex.
- Sharp points and corners should be avoided.

Figure 2.2 illustrates two classical ways of dividing a system into elements. The first is the Delaunay triangulation, which divides a system into triangles. The system is first described by nodes (points in space) that define the boundaries of the system and the shape of objects or regions (e.g. grains), or that are (randomly) distributed through the system (Fig. 2.2a). The Delaunay triangulation connects all these nodes by triangular elements. This means that the more nodes there are, the smaller the elements will be. The first step is to define circles that go through three nodes only and which *contain no other nodes*. The three nodes through which such a circle goes define one triangular element. This is repeated until the whole system is covered by triangles (Fig. 2.2b). The advantage of this technique is that it favours equidimensional triangles over pointed ones, although some pointed triangles may still occur, depending on the distribution of the nodes. One would rarely have to write code to do such a Delaunay triangulation, as free software for this is available in the Public Domain. Such software usually allows the user to define the minimum angle of corners and the minimum/maximum size of triangles. These requirements are then met by adding or deleting nodes.

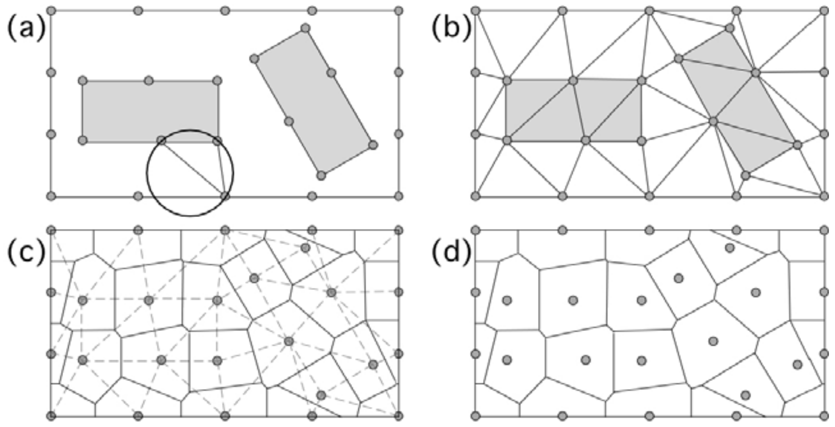


Fig. 2.2. **(a)** A system, with two rectangular regions shown in grey, which is to be divided into elements. The regions and the boundaries are defined by nodes (small circles). For a Delaunay triangulation, one defines triangles that go through 3 nodes and that contain no other nodes (large circle, for example). The three nodes then define one triangle. **(b)** Result of a Delaunay triangulation. The edges of the triangles coincide with the boundaries of the grey regions. **(c)** A Voronoi network is created by drawing line segments perpendicular to the Delaunay segments (dashed lines), exactly halfway between nodes. **(d)** The resulting Voronoi network

Delaunay triangles are geometrically linked to Voronoi networks. One can draw lines perpendicular to the Delaunay segments, exactly halfway between nodes (Fig. 2.2c). These segments then define a network of polygons, which is called a Voronoi network (Fig. 2.2d). The Voronoi polygons have some special properties:

- Each Voronoi polygon contains only one single node.
- Any point within a Voronoi polygon is closer to the one node in that polygon than to any other node in the system.
- Voronoi polygons are always convex.

If the nodes by which the Voronoi network is defined have a regular distribution, then the Voronoi polygons will be approximately equidimensional and similar in size.

2.1.2 Variables and parameters

Within a given framework, variables describe the system. Independent variables define the base of the system. Such variables are for example the length scale (size) of the system, or variables such as temperature or pressure if these are given as constant values. Dependent variables or state variables describe the state the system is in. As the model evolves, the state variables change. If the model describes the deformation of a grain, the state variables that describe the shape of the grain and the stress state evolve over time. Other state variables that may be involved in such a model could be the lattice orientation and dislocation density.

Almost any model will also use certain parameters that do not change as the model evolves. Such parameters are, for example, the Boltzmann constant, or material properties such as the Young's modulus or lattice parameters.

2.1.3 Equations

The variables and parameters define the system in the chosen framework. For something to happen in the model, equations are needed. The equations can change the state variables. There are two main types of equations:

- Path-independent equations change state variables independently of the current state of those variables. For example, a flow law that calculates strain rate as a function of differential stress always gives the same strain rate for a given stress, irrespective of the current strain rate.
- Path-dependent equations or evolution equations change state variables based on the current state of those variables. If evolution of a state variable over time (t) is modelled, such evolution equations typically have the form $State_{(t+\Delta t)} = State_{(t)} + function \cdot \Delta t$. Kinematic equations are a special type of evolution equations that change the position of something in the system, like the position of a particle or a boundary.

2.1.4 Boundary conditions

Normally, no model can do without boundary conditions. Typically, these are the values of some state variables at the physical boundaries of the model. For example, ductile deformation of a block of material requires the stresses or velocities at at least part of the surface of the block to be specified. Choosing the boundary conditions is by no means trivial, because they can play a significant role in the outcome of a simulation. An example was given by Bons et al. (1997) for the simulation of the rotation of a rigid clast in a viscous matrix that undergoes simple shear. The simple shear boundary conditions can be applied by letting the top of the model move to the right and the bottom to the left by a fixed rate. These are velocity boundary conditions, which then determine the stress needed to achieve the shearing. Alternatively, one could set the stresses at the boundaries and then calculate the resulting velocities. Although both simulate identical bulk simple shear, the effect on the object in the middle of the model turned out to be slightly different.

Boundary conditions must be carefully chosen and must be clearly stated in any reporting of modelling. This may seem obvious, but unfortunately this is not always the case. To really understand the boundary conditions, it is necessary to be thoroughly acquainted with the model that one uses. Using ready-made models as a "black box" can easily lead to erroneous results, if the boundary conditions and underlying assumptions are unknown to the user.

2.2 Lattice data structures: Monte Carlo, Ising, Potts, etc.

2.2.1 Introduction to Monte Carlo Principle

Monte Carlo models or stochastic models are models that involve probabilities. Deterministic equations of the type $a=F(b)$ will always give the same value of a for a given value of b. If chance comes into play: $a=P(b)$, the value of a can vary, depending on the probability function $P(b)$. Such equations would be suitable to simulate, for example, vacancy diffusion in a lattice. Each vacancy has a certain probability per time unit to jump to a neighbouring lattice site. If there are six such neighbouring sites, one could role a dice each time the vacancy jumps to determine to which of the six positions the vacancy will jump. Brownian motion of a small particle can be simulated in a similar way. Figure 2.3 shows the path of a particle that jumps a unit distance in a random direction each time step.

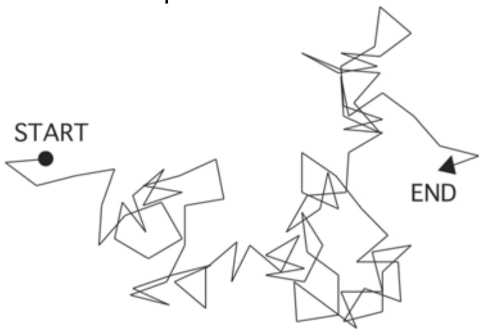


Fig. 2.3. Example of a Monte Carlo simulation for 100 steps of Brownian motion of a particle. Each time step, the particle jumped a unit distance in a random direction

Some classical Monte Carlo models are presented in the following sections. The ones presented here have in common the fact that the system to be modelled is divided up into small particles, typically on a lattice. Each lattice site has one or more state variables, such as magnetic spin or lattice orientation. The state variables can have a limited range of values. Rules are set up that determine the probability that a state variable changes or “flips” from one value to another. The “Monte Carlo” principle comes in at the stage where the dice are rolled to determine whether this change actually occurs for that lattice site. This step is repeated over and over again for all lattice sites and the system slowly evolves.

2.2.2 The Ising model

The first Monte Carlo model we will introduce here is the Ising model. It is one of the simplest Monte Carlo models, but already shows the rich behaviour that such models can exhibit. Potts models and Cellular Automata that are discussed next are essentially more complex variations of the Ising model. The Ising model is therefore a good example of the principles that all these models share.

The Ising model was developed to model magnetisation of materials. The magnetisation state of a particle within a volume of material depends on the spin of electrons within that particle. In a fully magnetised magnet, all spins will be the same. The magnet can have its positive pole either on one side or the other. This spin can therefore be described as having two states, say either "-1" or "+1". In a homogeneously magnetised material, all particles will have spin state -1 or all have spin state +1. If the magnet is heated, magnetisation is lost at a certain temperature, the Curie temperature. At that temperature some particles will have variable spin states between -1 and +1. The Ising model was developed to simulate what happens around T_{Curie}.

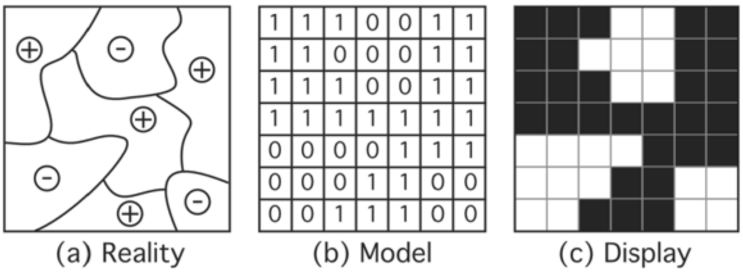


Fig. 2.4. **(a)** Regions in the magnetised material have either a positive or negative spin. **(b)** The regions are mapped as zeros and ones onto a square lattice, where zero stands for a negative spin, and one for a positive spin. **(c)** The map is easily displayed on the computer screen, where the spin states are represented by either black or white

The Ising model is a lattice model, as the material that is modelled is mapped onto a lattice (Fig. 2.4). In its simplest form this is a square lattice or grid. Each node in the lattice represents one small particle of material and all these particles have the same size. An advantage of such a square grid is that is easily displayed on a computer screen, where each lattice site can be represented by one screen pixel. As each lattice site can have only one of two states (-1 or +1), each node in the lattice can also only have one of two states. In a numerical simulation, one would probably choose "0" (representing -1) and "1" (representing +1), in which case only one bit would be needed to store the spin state of one lattice site.

The spin state of lattice sites can vary over time. Since only two states are possible, this means that lattice sites may switch their state from -1 to +1, or vice versa. Evolution equations are therefore needed to model the evolution of magnetisation over time. In the Ising model these evolution equations are based on the state of the local neighbourhood of a lattice site, not on long-range effects or on its history. The spin (S) of a lattice site, at time $t+\Delta t$, is a function of its own spin and those of the neighbouring lattice sites at time t. In the Ising model, the function is a probability function, which provides the probability that a lattice site flips its state from one to the other in a certain time step. This probability (P_{flip}) is a function of the difference (ΔE) between the local energy state before and after a switch in spin.

The local energy state (E_h) around lattice site h is a function of all energy states of J pairs of lattice sites:

$$E_h = \sum_{j=1}^{J} \zeta_{ij} e, \tag{2.1}$$

where e is the energy state of two neighbouring lattice sites and ζ_{ij} is the so-called "Hamiltonian" defined as:

$$\zeta_{ij} = 0 \text{ when } S_i = S_j, \text{ and} \tag{2.2a}$$

$$\zeta_{ij} = e \text{ when } S_i \neq S_j. \tag{2.2b}$$

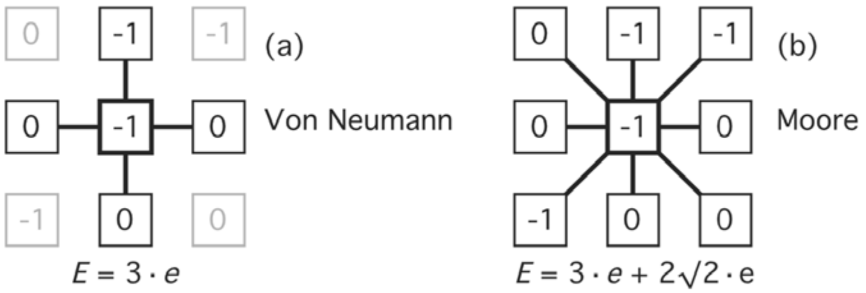


Fig. 2.5 Types of neighbouring schemes to calculate the probability that the lattice point in the middle will flip to the opposite state. **(a)** The Von Neumann scheme only takes the four nearest neighbours into account (6 in 3D). **(b)** The Moore scheme also takes the diagonal neighbours into account, with a different weighting to account for their distance (26 in 3D)

This means that pairs of lattice sites with the same spin have zero energy and pairs with opposite spins have energy *e*. Which neighbours are taken into account depends on the lattice type. In a 2-dimensional square lattice, two basic types of neighbouring are normally used. The simplest is “Von Neumann neighbouring” that only takes into account the four immediate neighbours (Fig 2.5a). “Moore neighbouring” takes into account 8 neighbours (Fig 2.5b), whereby a geometric factor needs to be introduced for the difference in distance between direct neighbours and oblique neighbours:

$$E_h = \sum_{j=1}^{4} \left(\zeta_{ij} e\right)_{direct} + \frac{1}{\sqrt{2}} \sum_{j=1}^{4} \left(\zeta_{ij} e\right)_{oblique}. \tag{2.3}$$

Various schemes can, of course, be designed to also include lattice sites further away than the immediate neighbours.

The probability, P_{flip}, that a lattice site switches its state is now given by the following evolution equation:

$$P_{flip} = P_0 \qquad \text{if } \Delta E \leq 0, \tag{2.4a}$$

$$P_{flip} = P_0 \exp^{\frac{-c\Delta E}{kT}} \qquad \text{if} \Delta E > 0. \tag{2.4b}$$

Here k is Boltzmann’s constant, T is the absolute temperate and c a rate constant (Fig. 2.6). This means that P_0 is the probability that a switch in state occurs if that switch achieves a reduction in local energy. P_0 can be seen as the frequency at which a lattice site attempts a switch in state within a certain period of time, the time step in the model. If the switch increases the energy, a switch may still occur, but the probability reduces with the gain in energy. The probability is also temperature dependent: a switch that increases the local energy state is more likely to occur at high temperature than at low temperature. The rate constant c bundles several parameters that scale the model in space and time. The complexity of the actual scaling of the Ising model is unfortunately beyond the scope of this chapter.

The Ising model is a Monte Carlo type model, because of the stochastic nature of the evolution equation (Eq. 2.4). Equation 2.4 defines the probability that a switch in spin of a lattice site would occur, but it does not tell whether it will occur. A random number generator is used to obtain a value Q between zero and one. If $Q>P_{flip}$, no switch in spin occurs, and the spin of the lattice site is switched to the opposite state if $Q \leq P_{flip}$.

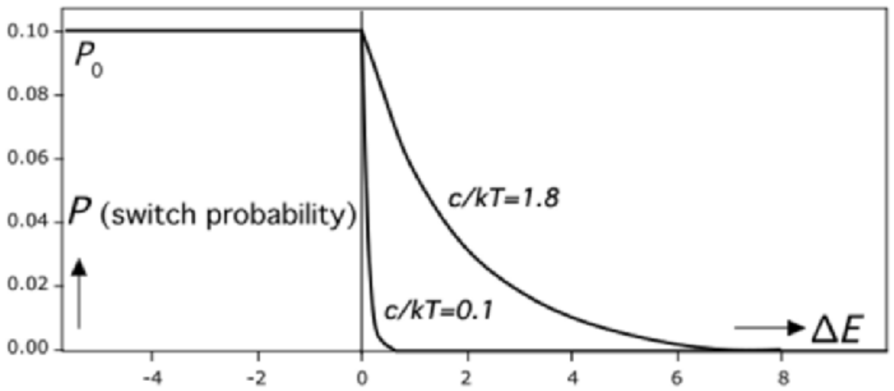


Fig. 2.6 Graph of the probability (P) that a flip will occur as a function of the energy change (ΔE, with e=1) such a flip would produce

An example of a 2-dimensional implementation of the Ising model is shown in Fig. 2.7, using a 200 x 200 square lattice with fully wrapping boundaries. Pixels in the image are coloured black and white to represent the two possible spin states. Lattice sites are randomly chosen and their ΔE is calculated using Moore neighbouring with e=1 (Eq. 2.3). This means that the maximum energy gain or loss is ±(4 + 4/ 2). A second random number is then used to determine whether the lattice site will change its spin (Eq. 2.4). P_0 is set to 0.1 and c/kT in Eq. 2.4 are lumped together in one unscaled "temperature" parameter, $1/T^*$. On average, each lattice site is selected once every time step.

Figure 2.7 shows the evolution, starting with randomised spins, of the spin distribution in 3 stages for different values of T^*, as well as the percentage of lattice sites in one spin state (black). Clusters of same-spin lattice sites quickly form below the Curie temperature (T^*=0.1). These clusters grow, until eventually all lattice sites have the same spin: the magnetisation is complete and frozen-in. At or above the Curie temperature (close to T^*=1.8), clusters form as well, but keep on evolving. No spin state ever wins over the other, as their fractions fluctuate around 50%: the material does not get magnetised. Clusters have highly irregular fractal shapes around the Curie temperature.

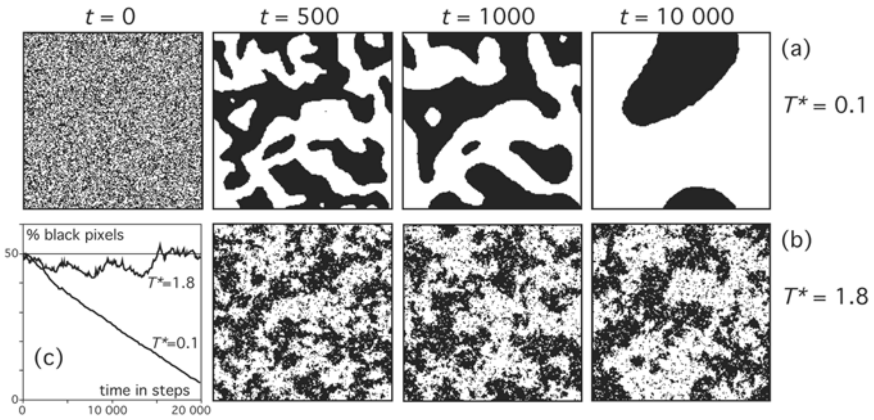


Fig. 2.7 Example of an Ising model, using a 200x200 square grid with fully wrapping boundaries. **(a)** Simulation at a temperature of $T^*=0.1$, well below the Curie temperature. In the end, all lattice points will have the same value. **(b)** Simulation at a temperature of $T^*=1.8$, well above the Curie temperature. Lattice points keep on flipping from black to white and vice versa, and the material never becomes fully magnetised. **(c)** Graph of the percentage of black pixels in the model as a function of time. Each lattice point is treated on average once every time step

2.2.3 Potts models

The Ising model described above allows only two possible states (spin is -1 or +1) for each lattice site. Clearly, the number (q) of possible states is not limited to two. Models where the number of possible states is larger than two (q>2) are called "q-state Potts models", or simply "Potts models". The basic Ising model is therefore a special q=2 case of a Potts model, designed to model magnetisation. Because of their simplicity, Potts models are very popular and useful models. They are particularly well suited to model granular aggregates and have therefore been extensively used for the simulation of static grain growth in crystalline aggregates.

A q>2 state Potts model works essentially the same way as the 2-state Ising model, with the only exception that additional rules must be defined to what other state one lattice site can switch its state. The typical choice would be that one lattice site can only switch to the state of one of its neighbours. This still leaves open two options for the switch:

- the lattice site may switch to the state of a randomly selected neighbour (an additional Monte Carlo step),
- or it may switch only to the energetically most favourable other state.

In either case, the decision to actually switch the state of the lattice site is still dependent on some energy-change dependent probability P_{flip}.

In static grain growth, individual grains grow or shrink to reduce the total surface energy of the grain boundaries. If surface energy per area of grain boundary is a constant, the surface energy can only be reduced by reducing the total surface of grain boundaries. Figure 2.8 shows an example of a 255-state Potts model for grain growth. The model is 2-dimensional and uses a square lattice. Grains are represented by continuous clusters of lattice sites that have the same integer value between 0 and 255. The model uses Moore-neighbouring to calculate the energy state before and after a possible switch. All grain boundaries have the same energy, represented by the parameter $e = 1$ in Eq. 2.1. At each time step, all lattice sites are selected randomly and on average once. A selected site may switch to the state of one of its four direct neighbours, according to Eqs. 2.3 and 2.4, using P_0=0.1 and T^*=0.5. A random number is used to select which of the four neighbour states is considered for a possible switch.

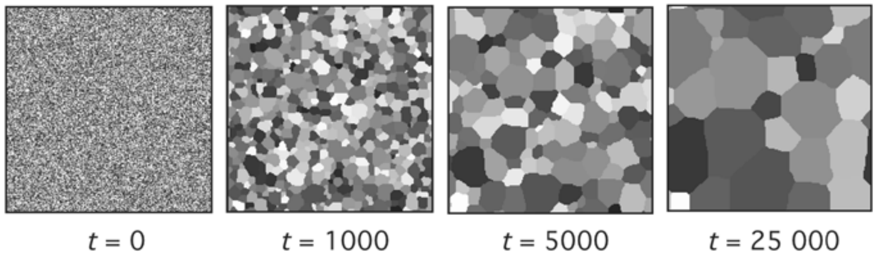


Fig. 2.8 A 255-state Potts model of static grain growth in a grain aggregate with fully wrapping boundaries. Notice that the square grid produces a prevalence of vertical, horizontal and diagonal grain boundaries

The main advantage of Potts models is their relative numerical simplicity. They are very easy to code and the rules to determine the probabilities that a switch of state may occur are separate from the rest of the model. This means that one can easily take the source code of a working Potts model and essentially only modify those routines that define the evolution equation (determining P_{flip}, and the possible state switches) to change the model for any desired purpose. Another advantage is that a 3-dimensional model is in essence no more complicated than a 2-dimensional one. The main difference between the two lies in the number of neighbours that need to be taken into account in the evolution equation.

Potts models are, however, not universally used, as they also have some serious drawbacks:

- The biggest problem with Potts models is that they are based on regular lattices or grids. Grids normally have equal sized square/cubic or triangular/tetrahedral elements. The example of grain growth (Fig. 2.8) shows that a square grid tends to favour horizontal, vertical and diagonal grain boundaries. Grain boundaries in this model can only move by discrete steps, defined by the grid type. This problem can, but only to some extent, be circumvented by employing more elaborate evolution equations that involve neighbours further away than the nearest ones.
- Potts models are not very efficient when high resolution is required. To get a decent resolution, one may want to use a 1000 x 1000 grid in two dimensions (10^6 lattice sites). In three dimensions this is equivalent to 10^9 lattice sites, which is still a large number for current computers to deal with. In case of grain growth simulations, most lattice sites lie inside grains, not neighbouring a boundary. These lattice sites do nothing, except using up memory space and computing time, although this can be avoided by storing and updating the nature of each lattice site. This problem increases with higher resolution, where the ratio of grain boundary lattice sites to grain interior lattice sites increases.
- State variables in a Potts model can only have q discrete values. If q is large this may not be a problem, but for some applications a continuous range of possible states is imperative.

Summarising, Potts models can be very useful to simulate certain processes in crystalline aggregates, because of their numerical simplicity and ease of extension to the third dimension. They are, however, limited in their applicability and more demanding scientific questions and problems may often require more powerful methods, such as Finite Element modelling or Front-Tracking.

2.2.4 Cellular Automata

Cellular Automata are a class of numerical models that include Potts models. In Cellular Automata the system is again divided up into particles, which typically lie on a regular grid. Each lattice site would have one or more state variables that change according to certain evolution equations, which typically involve a stochastic (Monte Carlo) component. The difference with Potts models is that evolution equations in Cellular Automata are not as restricted. In a Potts model, one lattice site can only switch its state to another one according to an equation that takes into account the local neighbourhood. This restriction is dropped in the case of Cellular Automata. One change in the model may involve many, if not all lattice sites in the model. Potts models thus form a subclass of Cellular Automata.

One example of a Cellular Automaton is the classical forest fire model (Fig. 2.9) (Bak et al. 1990; Malamud and Turcotte 1999). The model is designed to simulate the dynamics of forest fires. It is a 2-state model, where nodes on a grid can have only two states: being a tree or not. Trees are "planted" by randomly selecting a grid node and then deciding, by some stochastic rule, whether the state of the node should become "tree". Forests are connected clusters of "tree-nodes". This evolution rule alone would lead to the whole grid becoming covered with trees in time. However, a second rule is employed to burn trees. By another stochastic rule, matches are occasionally thrown on the grid. If a match happens to land on a grid node with a tree, not only the tree burns, but also the whole forest to which this tree belongs. Burning trees is implemented as reverting the state of a node from "tree" to "not tree". Although the rules are extremely simple, the behaviour of the system is extremely complex and the resulting statistics bear strong similarities with the Ising model near the critical Curie temperature.

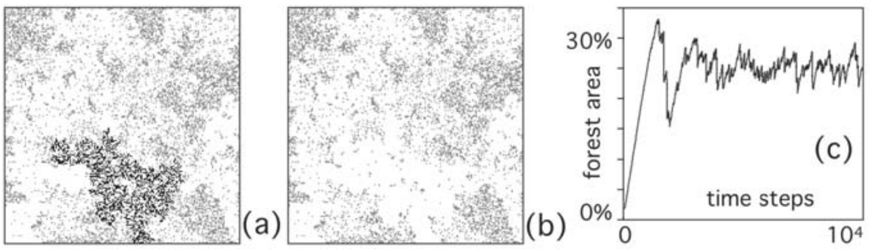


Fig. 2.9 Example of the forest fire model at the moment of a fire **(a)** and one time step afterwards **(b)**. Grey pixels represent trees, black pixels burning trees and white areas with no trees. All trees connected (Moore neighbouring) in one cluster burn if a match lands on one of the trees. In the large fire in (a) 2038 trees burn. Model size is 200x200. Each time step one tree is planted on a randomly selected lattice node. Each time step, the probability of throwing a match on a random lattice points is 1%. **(c)** Forest density (percentage of sites covered by trees) as a function of time

2.2.5 Lattice-Gas and Lattice-Boltzmann method

One of the most important and widely used classes of Cellular Automata is the Lattice-Gas method. A common example is the modelling of a fluid (which is traditionally called a Lattice-Gas model) in which, as with the Potts models and forest fire model, a discrete grid is used. Each site on the grid is characterized by a number of states that are changed by a given update rule each "time step", where the rule determines how the state of a site changes based on its neighbours. This method is, like all Cellular Automata, very efficient for computers since one or a small number of bits are enough to store the state of a single site in the lattice. In the case of fluids each lattice site represents a small part of the fluid. In order to be consistent the models normally conserve momentum and mass. Particles in the fluid are now either present or absent on lattice sites. Initially particles will be given a velocity in a desired direction or initial velocities are chosen randomly. How particles travel through the lattice depends on their velocities and on collisions with other particles or boundaries (for example rough walls of a crack).

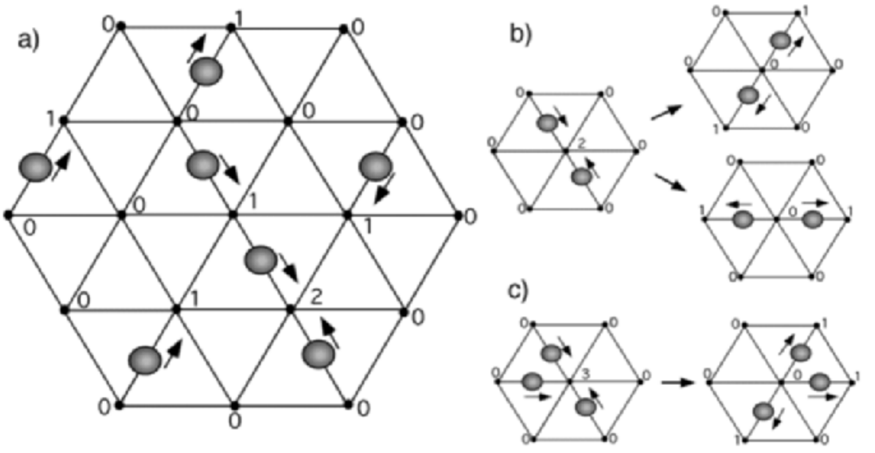


Fig. 2.10 The figure shows an example of a Lattice-Gas model with a triangular lattice **(a)**. Lattice sites are given as small black circles whereas fluid particles are large grey circles. Unit vectors of particle movements are given as arrows. Numbers on lattice sites mark the number of approaching particles. If these numbers are greater than one, a collision occurs. **(b)** In case of a two-particle collision, two choices of scattering of particles are possible on a triangular lattice. **(c)** In case of a three-particle collision, particles are simply scattered to the free sites

As an example we will describe the so-called FHP (Frisch, Hasslacher and Pomeau) rule for a Lattice-Gas (Frisch et al. 1986). The FHP Lattice-Gas model uses a triangular 2-dimensional grid as basis (Fig. 2.10). Each particle has six possible states that describe its six possible neighbours, which are either there or absent. The update rules for states are performed in two steps that handle collisions and transport. If three particles collide at a given site they scatter in opposite directions. If two particles collide they scatter sideways in two possible directions. These two directions are chosen randomly. After collisions are handled, particles that have not collided move one step further depending on their given velocities (Fig. 2.10).

Even though the applied rules for Lattice-Gas models seem very simple, it can be shown that they can lead to the full Navier-Stokes equation where the particular rule of a Lattice-Gas model determines the viscosity of the fluid (for a full derivation see Gershenfeld 1999). In Fig. 2.11 we present two examples of a simple FHP Lattice-Gas model. It is included in *Elle* so that the reader can reproduce the examples (see Appendix B). A large variety of numerical experiments can be performed using Lattice-Gas models. These include two- and three-dimensional flow problems with complex geometries, as well as multiphase flow problems, problems of diffusion and even coupled reaction-diffusion problems (Rothman and Zaleski 1994).

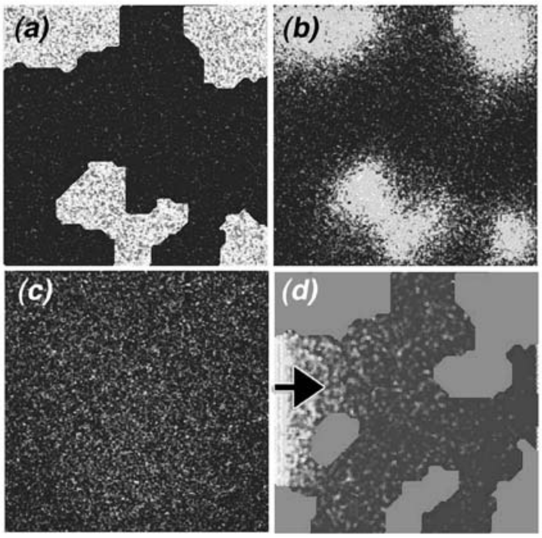


Fig. 2.11 Example of a Lattice-Gas model using a triangular lattice and simple FHP rules (Frisch et al. 1986). Example of diffusion after **(a)** 1, **(b)** 100 and **(c)** 1000 time steps and a triangular lattice with 46000 nodes. An initial low (dark) density Gaussian distribution of fluid particles is set in the background whereas a number of regions (which could be grains) have a high density of particles (bright). Diffusion of the particles eventually leads to an equal distribution of the particles over the whole model (EXPERIMENT 1). **(d)** Example of fluid flow through a porous medium (in 2-dimensions) where fluid is introduced on the left hand side and can leave on the right hand side. The upper and lower walls are fixed and repel fluid particles. Grey regions are solid grains. In the pore space in between brightness reflects particle density (pressure) from high on the left to low on the right. This simulation used 11500 nodes (EXPERIMENT 2)

An alternative numerical method for the solution of the Navier-Stokes equation is the Lattice-Boltzmann method. It is related to Lattice-Gas methods but uses a solution scheme that is similar to Finite Differences (see Sect. 2.4). As with the Lattice-Gas approach, space is represented by a discrete lattice. Variables in the approach are called population variables that evolve in discrete time steps. The basic Lattice-Boltzmann equation can be written as

$$N_i(x+c_i,t+1)=N_i(x,t)+\Delta_i\left[N(x,t)\right], \tag{2.5}$$

where N_i is a population on a lattice site, x the lattice site, x+c_i the new lattice site, t the time, Δ_i the Boltzmann collision operator and N the population b-vector (Rothman and Zaleski 1994). Equation 2.5 and the equation to derive the Boltzmann collision operator are the basis for the so-called Lattice-Boltzmann method. They are explained in detailed in Rothman and Zaleski (1994, page 1432 and appendix C). A simple version of the Lattice-Boltzmann method uses a linearised collision operator where the simplest form is

$$N_i(x+c_i,t+1)=N_i(x,t)+\sum_j \omega\delta_{ij}\left(N_j(x,t)-N_j^{(0)}(x,t)\right), \tag{2.6}$$

where ω is a parameter that determines viscosity of the fluid, δ_{ij} is the Kronecker delta (which together define the collision operator) and $N_j^{(0)}(x,t)$ pseudo-equilibrium populations (Rothman and Zaleski 1994, Appendix C). Note the similarity to conventional explicit Finite Difference methods (Sect. 2.4).

Symbols used in Chap. 2.2

a, b	Arbitrary variables
c	Rate constant
E	Energy state of lattice site
e	Energy contribution arising from the difference in state between two neighbouring lattice sites in Ising and Potts models
$F()$	Arbitrary function
h	Index of lattice site
J	Number of neighbours of a lattice site
k	Boltzmann's constant = $1.3806503 \cdot 10^{-23}$ m^2 kg s^{-2} K^{-1}
N	Population *b*-vector in a Lattice-Boltzmann model
N_i	Population on site *i* in a Lattice-Boltzmann model
$P()$	Probability function
P_{flip}	Probability to switch from one state to another
P_0	Probability that a switch in state occurs if that switch achieves a reduction in local energy
Q	Random number ranging from zero to one
q	Number of possible states of a lattice site in a Potts model
S	Spin
T_{Curie}	Curie temperature [K]
$1/T^*$	Unscaled temperature parameter ($=c/kT$)
Δ_i	Boltzmann collision operator
δ_{ij}	Kronecker delta
ζ_{ij}	Hamiltonian in Ising and Potts models
ω	Parameter determining the viscosity of a fluid in a Lattice-Boltzmann model

2.3 Boundary models

In the previous sections, the system to be simulated was mapped onto a lattice or grid. Each site on the lattice represented a small part of the system. For many models, this is a suitable way to describe the system, but there are some drawbacks. The first is the resolution, which is determined by the number of lattice sites. To make an accurate map of a granular aggregate with complex grain shapes, one needs a very large lattice. This can use up too much memory space and/or the calculation time becomes inhibitive. A second problem is that boundaries between phase regions (e.g. grains) themselves do not really exist. In the Potts model, a grain boundary is only implicitly defined as locations where lattice sites with different values are found next to each other. The position of the boundary is inferred, but it is not really there in the model. This is a problem when the boundaries themselves are important for the process that is simulated.

One well-known example of a system where the boundaries between regions are more important than the regions themselves is a foam (such as in the head of a beer). The regions themselves, the bubbles, are filled with gas and have no strength at all. The whole evolution of the foam is determined by the thin liquid walls between the bubbles. The surface tension strives to straighten the walls, which leads to a gradual increase of the average bubble size. Some bubbles shrink and eventually disappear, while others grow. The growth or shrinkage of a bubble is possible by the diffusion of gas through the bubble walls, driven by pressure differences between the bubbles as a result of the surface tension of the walls. A lattice model can and has been used to simulate this process, but since everything happens at the bubble walls, it is often considered to be more precise to describe the foam by the boundaries. The bubbles are then inferred as the regions enclosed by these boundaries, but do not play a major role in the process themselves. The evolution of the foam can be followed by tracking what happens at the boundaries or “fronts”, which is why such models are called “boundary-tracking models”, “Front-Tracking models”, or simply “boundary models”.

Boundary models typically describe the boundaries by nodes (points in space), connected by boundary segments (Fig. 2.12). The simplest segments are just straight lines connecting two nodes, but the segments can also be more complex curved lines (e.g. using Bézier curves). The advantage of curves is that fewer nodes are needed for an accurate description of a complex boundary. A disadvantage is that the mathematics become more complex. All boundary models in this book use straight boundary segments between nodes. An accurate description of the shape of a region can be achieved with relatively few boundary nodes, compared to the number of lattice sites in a lattice (Fig. 2.12).

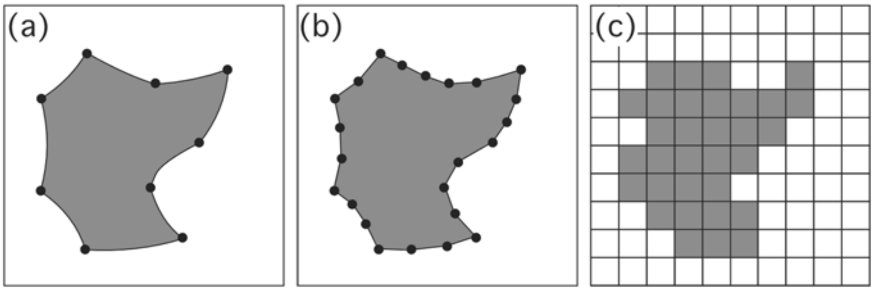


Fig. 2.12 **(a)** A region, like a grain, can be described accurately with only a few nodes that are linked by curved boundary segments, for example with Bézier curves. **(b)** More nodes, here 23, are needed to accurately describe the same region when the segments are simply straight lines. **(c)** A lattice with 10x10 points cannot accurately describe the shape of the region at all. A lattice with at least about 100x100=10000 points would be needed to describe the shape of the region reasonably well

The progressive evolution of a system is modelled by applying evolution equations to either the nodes or segments. These equations will:

- change the position of the boundary by changing the positions of nodes and segments. This is typically done by an equation that calculates the velocity of the boundary nodes or segments.
- calculate the rate of change of the state variables. These can be the properties of the boundaries, such as the film width in a foam, or grain boundary fluid composition in a granular aggregate. These can also be related to the regions enclosed by the boundaries.

Once the rate of change of position (velocity) and of the state variables are calculated, they are typically integrated over a small time increment and then applied to the model.

Many schemes can be used to calculate and apply the evolution equations, such as Finite Difference and Finite Element schemes that are introduced below, or schemes that involve some Monte-Carlo step. Here we give one example of the latter, since it illustrates well how Front-Tracking is usually modelled in *Elle* (Ch. 3.5).

Static grain growth is a process in granular aggregates where a reduction in surface energy drives an increase in average grain size. It is similar to coarsening of a foam or froth, and the resulting texture looks the same: a foam texture. Since the process is completely driven by the boundaries, it makes sense to use a boundary model. Here we take a model where the boundaries are defined by nodes that link straight segments. In the two-dimensional example model, each node is linked to two or three other nodes. A curved boundary will straighten to reduce the total surface energy, which is proportional to the total boundary length (area in 3D). The velocity (v) of a point on a circular boundary is determined by the mobility (m) and the surface energy per unit length (γ) of the boundary and the radius of curvature (R):

$$v = \frac{\partial R}{\partial t} = \frac{-m\gamma}{R} \Leftrightarrow \Delta R \approx \left(\frac{-m\gamma}{R}\right)\Delta t \tag{2.7}$$

(when Δt is very small). The velocity is directed towards the centre of curvature and always reduces R.

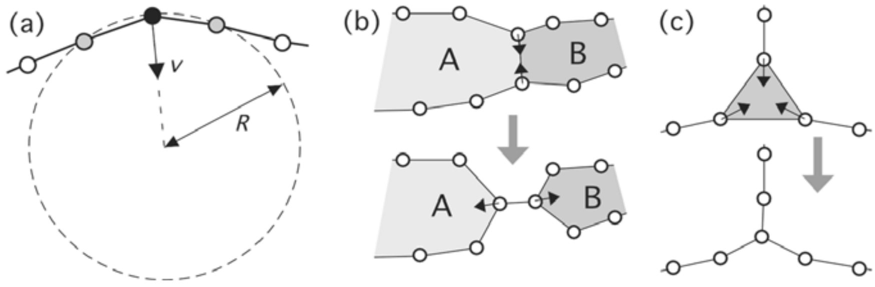


Fig. 2.13 **(a)** The velocity (v) of one node (the black one) is a function of the radius (R) of curvature. R can be determined by constructing a circle through the node and its two neighbours. Two additional topological routines are needed for the simulation of grain growth. **(b)** When two nodes approach each other, a neighbour switch needs to be carried out. Grains A and B were originally neighbours, but not any more after the switch. **(c)** When three nodes of a small three-sided grain approach each other, they merge into one, which means the small grain has disappeared

One step in the simulation involves randomly picking one of the nodes in the model. The local radius of curvature is calculated by fitting a circle through the node and its two neighbours (Fig. 2.13a). This gives both R and the direction towards the local centre of curvature. The position of the node is then updated by applying Eq. 2.7 for a small time increment. If the node is a triple junction where three boundaries meet, three movement vectors are calculated and simply summed. Three additional routines must be added to the model:

- Inserting or removing nodes to keep the average distance between nodes the same. If this is not done, one may change the "effective" time increment;
- Carry out neighbour switches when two triple junction nodes converge (Fig. 2.13b);
- Converting three converging triple junction nodes into one, which effectively removes a small three-sided grain (Fig. 2.13c).

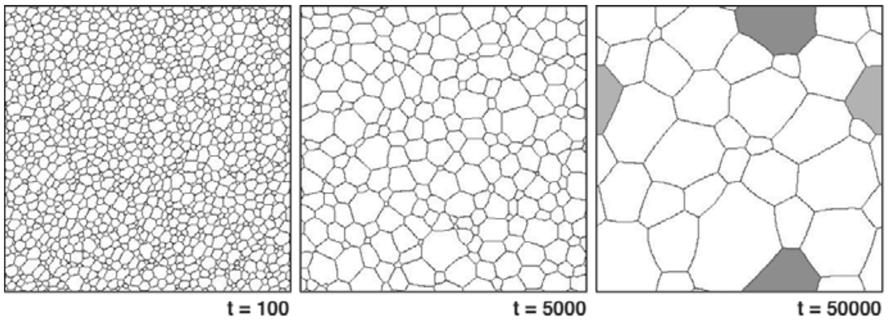


Fig. 2.14 Simulation of grain growth in an isotropic granular aggregate after 100, 5000 and at 50000 time steps. Node displacements were calculated with Eq. 2.7. Distance between the nodes was kept between $0.5 \cdot 10^{-3}$ and $1.1 \cdot 10^{-3}$ times the model size. Model boundaries are wrapping, so that a grain touching the top or left boundary continue on the other side, as illustrated with the two shaded grains

This simple model (Fig. 2.14) efficiently and accurately simulates ideal grain growth in isotropic granular aggregates, where the surface energy is not a function of the orientation of a boundary relative to the crystallographic lattice orientation of the grains on either side of the boundary. A more complex boundary model for anisotropic surface energy and other driving forces for grain-boundary movement is described in Chaps. 3.5-7.

Symbols used in Chap. 2.3

m	Mobility [m s^{-1} Pa^{-1}]
R	Radius of curvature [m]
v	Velocity [m s^{-1}]
γ	Surface energy per unit area of grain boundary [J m^{-2}]

2.4 Finite Difference method

2.4.1 Principles

The Finite Difference (FD) method is one of the most powerful and popular numerical modelling methods (e.g. Press et al. 1992; Gershenfeld 1999). It is ideal for problems such as large-scale flow through a porous medium or aquifer, or cooling of a dyke that intruded into cooler rocks. In both systems the models themselves do not change their geometry, but variables such as fluid fluxes, chemical composition of fluid and temperature vary in space and time. A fixed grid can be used to describe the system. These values are mapped onto this grid, and the FD method can be used to calculate how these values evolve over time and in space.

The FD method is based on the principle that any complex function ($f_{(x)}$) can be approximated with a simple linear function for a small increment of the independent variable (x), which could for example be space or time. Mathematically this can be described as:

$$f_{(x+\Delta x)} \approx f_{(x)} + \left(\frac{df}{dx}\right)\Delta x \quad \text{(when } \Delta x \downarrow 0\text{)}. \tag{2.8}$$

An example could be: $\cos(\alpha+\Delta\alpha) \approx \cos(\alpha) - \sin(\alpha)\cdot\Delta\alpha$. If $\alpha=45°$, the error is only 0.0001 for $\Delta\alpha = 1°$, and 0.0026 for $\Delta\alpha = 5°$ (Fig 2.15). This linear approximation is extremely useful, because linear functions are very easy to solve.

In the FD method, a system to be modelled is mapped onto some grid or lattice. Values of the state variables (e.g. temperature) are only recorded at the grid nodes. The values are assumed to vary linearly between these nodes. It is clear that the spacing between these nodes must be small enough to describe the real system accurately enough: Eq. 2.8 must be satisfied. Time-dependent changes to the system, like cooling of a dyke, are calculated in small increments, again assuming that changes are linear for small enough increments of time. Ideally, a system would be described by an extremely large number of nodes and all time increments should approach zero. However, limited computer memory and calculation time constraints mean that the user must find a compromise between accuracy on the one hand, and computing time and memory use on the other hand.

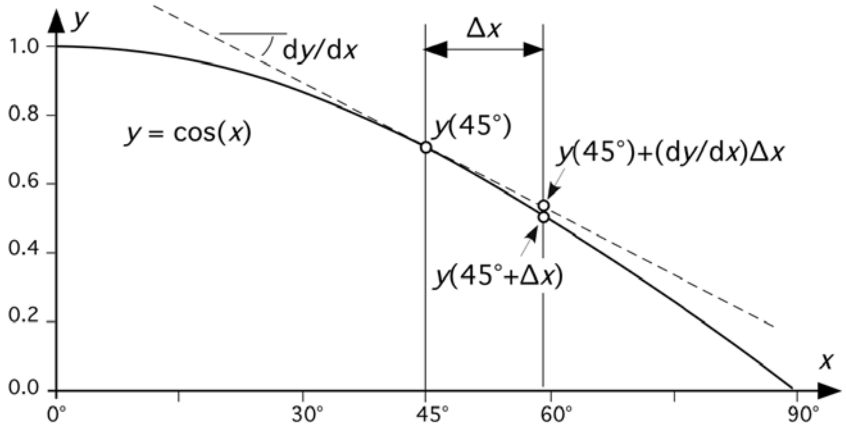


Fig. 2.15 Example of a smooth curve, in this case $y(x) = \cos(x)$. The value of $y(x+\Delta x)$ can be approximated from the value of $y(x)$ and the local gradient

In Eq. 2.8 we used the derivative of $f_{(x)}$ to approximately predict the value of $f_{(x+\Delta x)}$ after adding a small increment of Δx. In many cases, however, both $f_{(x)}$ and $f_{(x+\Delta x)}$ are known, but not the derivative of the function. The derivative can easily be obtained from Eq. 2.8:

$$\frac{df}{dx} \approx \frac{f_{(x+\Delta x)} - f_{(x)}}{\Delta x}. \tag{2.9}$$

This is called the "forward" approximation of the derivative (Fig. 2.16). Two other possibilities are the "backward" and "central" approximations, respectively (Fig. 2.16):

$$\frac{df}{dx} \approx \frac{f_{(x)} - f_{(x-\Delta x)}}{\Delta x} \text{ (backward)}, \tag{2.10a}$$

$$\frac{df}{dx} \approx \frac{f_{(x+\Delta x)} - f_{(x-\Delta x)}}{2\Delta x} \text{ (central)}. \tag{2.10b}$$

Often one is also interested in a second derivative of the function of interest (e.g. when Fick's second law is applied for diffusion). The second derivative of a function, of which the values are know at discrete increments Δx, can be approximated in a similar way as the first derivative:

$$\frac{\partial^2 f}{\partial x^2}=\frac{\partial\left(\frac{\partial f}{\partial x}\right)}{\partial x}\approx\frac{\left(\frac{f_{(x+\Delta x)}-f_{(x)}}{\Delta x}\right)-\left(\frac{f_{(x)}-f_{(x-\Delta x)}}{\Delta x}\right)}{\Delta x}$$
$$=\frac{f_{(x+\Delta x)}-2f_{(x)}+f_{(x-\Delta x)}}{(\Delta x)^2}. \tag{2.11}$$

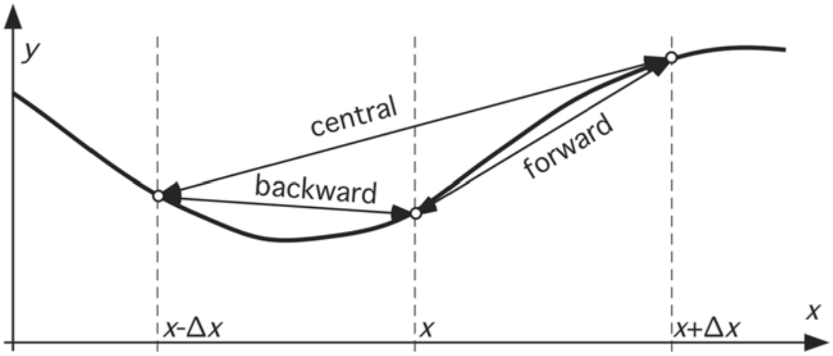


Fig. 2.16 To determine the local gradient at one point (x) on a lattice, one can use the difference with the next point (x+Δx; forward method), the previous point (x-Δx; backward method), or the points on both sides (x-Δx and x+Δx; central method). Note that the latter is closest to the true gradient at point x

2.4.2 Explicit Finite Difference method

One big advantage of the FD method is that it can deal very well with partial differential equations, such as for example:

$$\frac{\partial y}{\partial t}=c\frac{\partial^2 y}{\partial x^2}. \tag{2.12}$$

Here y is some state variable that can vary over time (e.g. temperature), c is a rate constant (a physical parameter, for example heat conductivity) and x an independent variable to describe space in one dimension. Equation 2.12 could describe heat flow next to a cooling dyke (y is temperature), or changes in trace element concentration during diffusion (y is concentration).

We will use a model for a cooling magmatic dyke as an example of how the FD method can be employed for a process (cooling) that can be described with a partial differential equation like Eq. 2.12. As usual with models, we will start with some assumptions (Fig. 2.17):

- The dyke intrudes as an infinite plate of width $2w$. This has the advantage that the model can be one-dimensional, with temperature only varying perpendicular to the dyke. This direction we will call the x-axis. Because of the symmetry of the system, we only need to model half of the system, putting $x = 0$ m in the middle of the dyke.
- The dyke intrudes instantaneously, so it has a constant temperature (T_{dyke}) at the time of intrusion ($t = 0$ s), at which we start the simulation.
- The wall rock into which the dyke intrudes has a constant temperature (T_{wall}) at the time of intrusion.
- For simplicity, we will assume that heat conductivity is a constant and is the same for the dyke as for the wall rock. We will also neglect factors such as latent heat of crystallisation, etc.. This means that we only need one rate constant in Eq. 2.12, namely the heat conductivity (κ).

In the FD model, the system at $t = 0$ s can now be described by N nodes, lying on the x-axis with a spacing Δx. All nodes that lie within the dyke ($x_i \leq w$) have T_{dyke}, and all others have T_{wall}. Here we use i as an index for the i-th node.

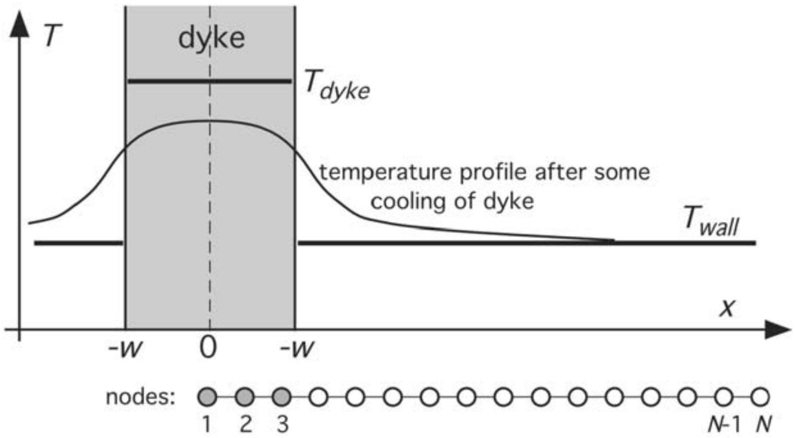


Fig. 2.17 Set-up for the one-dimensional Finite Difference model to simulate the cooling of a dyke that intrudes at a temperature T_{dyke}

To calculate the temperature of a node after a small time increment Δt, we must use Eq. (2.12) for heat conductivity:

$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial x^2}. \tag{2.13}$$

We use the assumption that temperature varies linearly with time (Eq. 2.8), if our time increment Δt is small enough:

$$T_i^{t+\Delta t} \approx T_i^t + \left(\frac{\partial T_i}{\partial t}\right) \cdot \Delta t = T_i^t + \left(\kappa \frac{\partial^2 T_i}{\partial x^2}\right) \cdot \Delta t. \quad (2.14)$$

The way the second derivative of temperature to x can be calculated was already provided in Eq. 2.11. Inserting Eq. 2.11 into 2.14 gives:

$$T_i^{t+\Delta t} \approx T_i^t + \left(\frac{\kappa \Delta t}{(\Delta x)^2}\right)\left(T_{i+1}^t - 2T_i^t + T_{i-1}^t\right). \quad (2.15)$$

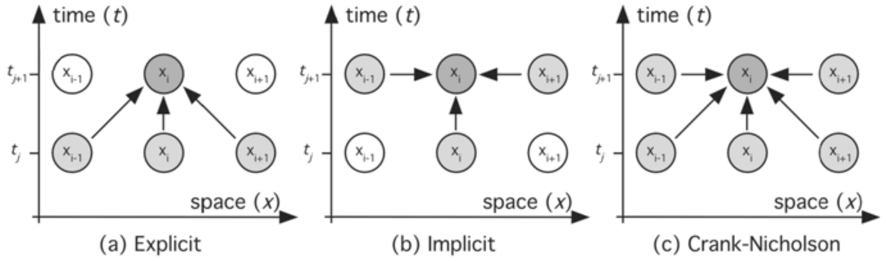


Fig. 2.18 Scheme showing which nodes contribute to the calculation of the state of node i at position x_i and time step t_{j+1}, for the explicit method **(a)**, the implicit method **(b)**, and the combined Crank-Nicholson scheme **(c)**

The great advantage of Eq. 2.15 is that we can calculate the temperature of node i at time $t+\Delta t$, using only known temperatures at time t. For one time increment we can thus update all the temperatures of the nodes, and then continue onto the next time increment. This method is called the “explicit” method. One big advantage of this method is that it is simple. To calculate the new value of a state variable, like T in our example, we only need to take into account the current values of a limited number of neighbouring nodes in our FD-grid (Fig. 2.18a). Of course there are also disadvantages. Remember that Eq. 2.15 is only an approximation. This means that a small error is made every time increment. These errors add up, which is all right if they balance on average. If, however, these small errors are always on one side of the correct solution, the model will progressively deviate further and further from the true solution. This is illustrated in fig. (2.19) where an explicit method is used to calculate the path of a particle that should move in a circle. If the particle moves clockwise with a velocity of v_0 on a circle with radius 1, its velocity (v) is described by:

$$v_{(x)} = \frac{\partial x}{\partial t} = v_0 y \text{ and } v_{(y)} = \frac{\partial y}{\partial t} = -v_0 x . \tag{2.16}$$

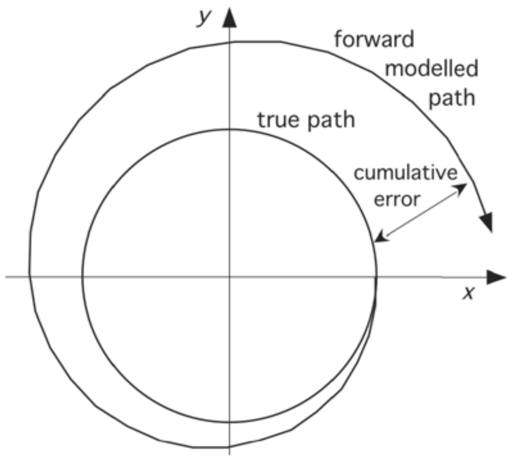


Fig. 2.19 A circular orbit of a particle simulated with the explicit forward Finite Difference method. Each time step a small error is made, which adds up in a systematic way

Using the explicit FD method, we can use Eq. 2.8 to approximate the position ($x_{t+\Delta t}$,$y_{t+\Delta t}$) of the particle after a small time increment Δt, using the current position of the particle (x_t,y_t):

$$x_{t+\Delta t} \approx x_t + \left(\frac{\partial x}{\partial t} \right) \Delta t = x_t + v_0 y_t \Delta t \tag{2.17}$$

$$\text{and } y_{t+\Delta t} \approx y_t + \left(\frac{\partial y}{\partial t} \right) \Delta t = y_t - v_0 x_t \Delta t . \tag{2.17}$$

Using this scheme, the particle will slowly move away from the centre of rotation, no matter how small we make Δt. This can be seen if we calculate the distance of the particle before (r_t) and after a time increment Δt ($r_{t+\Delta t}$):

$$r_t = \left\{ x_t^2 + y_t^2 \right\}^{\frac{1}{2}}, \text{ and} \tag{2.18}$$

$$r_{t+\Delta t} = \left\{ \left(x_{t+\Delta t} \right)^2 + \left(y_{t+\Delta t} \right)^2 \right\}^{\frac{1}{2}} = \left\{ \left(x_t + v_0 y_t \Delta t \right)^2 + \left(y_t - v_0 x_t \Delta t \right)^2 \right\}^{\frac{1}{2}} . \tag{2.19}$$

Rearranging Eq. 2.19 and inserting 2.18 then gives:

$$r_{t+\Delta t} = \left\{\left(x_t^2 + y_t^2\right) + \left(v_0 \Delta t\right)^2 \left(x_t^2 + y_t^2\right)\right\}^{\frac{1}{2}} = \left\{r_t^2\left(1 + \left(v_0 \Delta t\right)^2\right)\right\}^{\frac{1}{2}} \tag{2.20}$$

$$\Leftrightarrow r_{t+\Delta t} > r_t .$$

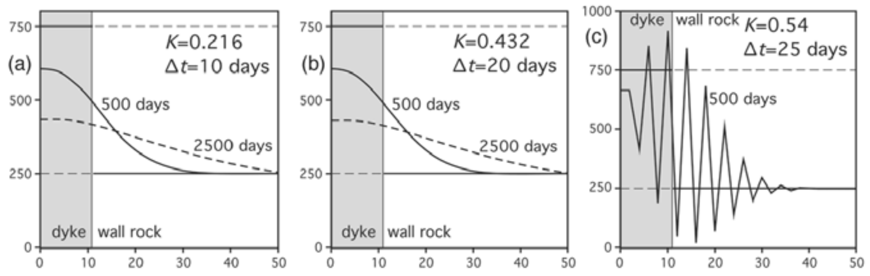


Fig. 2.20 Example of an explicit Finite Difference model to simulate the cooling of a 22 m wide dyke (infinite plate) with an initial temperature of 750 °C that intruded into a wall rock at 250 °C. Only one half of the system was modelled with 25 nodes, with a 2 m spacing. Heat conductivity was assumed equal in both dyke and wall rock at 10^{-6} m^2/s. Boundary conditions are: $T_1=T_2$ and $T_{25}=250$ °C. The temperature distribution is shown after 250 and 2500 days of cooling for three different time steps: **(a)** 10 days, **(b)** 20 days, and **(c)** 25 days. In the first two cases, the parameter *K* (Eq. 2.21) is less than 1/2, and both solutions are practically the same. *K* is 0.54 with a time step of 25 days, and the solution very quickly deteriorates and oscillates wildly

A second possible problematic aspect of the explicit method is the possibility of oscillation: the calculation does not converge towards the right solution but cycles wildly around it. Whether the explicit calculation as used in Eq. 2.15 is stable, depends on the conductivity constant, and the time and length resolution. These can be grouped together in one parameter *K* defined as:

$$K = \frac{\kappa \Delta t}{\left(\Delta x\right)^2} \Leftrightarrow T_i^{t+\Delta t} \approx T_i^t + K\left(T_{i+1}^t - 2T_i^t + T_{i-1}^t\right). \tag{2.21}$$

Interestingly, there is a critical value for *K*, above which the solution starts to oscillate. *K* should always be chosen below 1/2 to avoid oscillation (Fig. 2.20). This means that the choices of Δt, Δx are not completely free, and not independent of the conductivity constant. In some cases this can be a major problem. If a very high spatial resolution is needed, Δx must be very small. This implies that Δt must also be set at a small value, which means that many time steps must be calculated, leading to a large calculation time.

Since the state of a node after one time increment is only a function of the current state of that node and its nearest neighbours, it may take a long time for the effect of a perturbation to propagate through the system. The 100th node away from the dyke surface in our example only "notices" the temperature perturbation caused by the dyke after 100 time steps. If we would double the spatial resolution, the 100th node would only be half as far away from the dyke surface, but would still only "notice" the dyke after 100 time steps. We see that the time needed for a perturbation to propagate a given distance is a function of the spatial resolution of the simulation. This is clearly not satisfactory in some cases.

Equation 2.15 can easily be solved for almost all nodes in the model, because the equation only uses data that are already known. There are two exceptions, namely the first and last node (number i=1 and i=N). For those two nodes, Eq. 2.15 reads:

$$T_1^{t+\Delta t} \approx T_1^t + \left(\frac{\kappa \Delta t}{(\Delta x)^2} \right) \left(T_2^t - 2T_1^t + T_0^t \right), \tag{2.22a}$$

$$T_N^{t+\Delta t} \approx T_N^t + \left(\frac{\kappa \Delta t}{(\Delta x)^2} \right) \left(T_{N+1}^t - 2T_N^t + T_{N-1}^t \right). \tag{2.22b}$$

The problem is that node "0" and node "N+1" do not exist. These two nodes lie just outside the inevitable boundaries of the system. For these boundaries one must make exceptions, or special rules. These are normally called the "boundary conditions". In the example of the cooling dyke (Fig. 2.20), the following boundary conditions were chosen:

- $T_1 = T_2$. This means that the temperature gradient in the middle of the dyke is zero. This is correct if the system is symmetric, as is the case here.
- $T_N = T_{wall}$. This means that the temperature remains constant at the node furthest away from the dyke. Strictly speaking, a point infinitely far away from the dyke should still sense a slight rise in temperature. However, if node N is far away enough, this boundary condition will not alter the solution much.

Since the boundary conditions already determine the values for the first and last node, the explicit FD calculation is only applied to nodes 2 through to N-1, for which all information is available.

2.4.3 Implicit Finite Difference method

Some of the problems inherent to the explicit method can be avoided with the "implicit" method. The explicit method calculates the state of a node after a time increment completely from available data for the current state. This is different in the implicit method, where information is used from both the current state and the state after the time increment (Fig. 2.18b). Instead of Eq. 2.15, the implicit method uses:

$$T_i^{t+\Delta t} \approx T_i^t + \left(\frac{\kappa \Delta t}{(\Delta x)^2} \right) \left(T_{i+1}^{t+\Delta t} - 2T_i^{t+\Delta t} + T_{i-1}^{t+\Delta t} \right), \tag{2.23a}$$

$$\Leftrightarrow T_i^{t+\Delta t} \approx T_i^t + K\left(T_{i+1}^{t+\Delta t} - 2T_i^{t+\Delta t} + T_{i-1}^{t+\Delta t} \right). \tag{2.23b}$$

At first it may seem odd to calculate the second spatial derivative of temperature at time $t+\Delta t$. We cannot straightforwardly solve Eq. 2.23, because the values we want to calculate are on the right side of the equation as well. Still the equation can be solved, but only simultaneously for all nodes in one step. If we have N nodes, Eq. 2.23 can be written N times. This gives us a set of N linear equations with N+2 unknowns ($T_l{}^{t+\Delta t}$ through to $T_{N+1}{}^{t+\Delta t}$). The "+2" comes from the non-existent nodes on the left and right side of the model. If we use the boundary conditions to provide the values for T_1 and T_N, we are left with N-2 linear equations and N-2 unknowns, which means we can solve for all the unknowns. The set of equations looks like this:

$$\begin{aligned}
T_2^t &\approx -KT_1^{t+\Delta t} + (1+2K)T_2^{t+\Delta t} - KT_3^{t+\Delta t} \\
T_3^t &\approx -KT_2^{t+\Delta t} + (1+2K)T_3^{t+\Delta t} - KT_4^{t+\Delta t} \\
&\cdots \\
T_{N-2}^t &\approx -KT_{N-3}^{t+\Delta t} + (1+2K)T_{N-2}^{t+\Delta t} - KT_{N-1}^{t+\Delta t} \\
T_{N-1}^t &\approx -KT_{N-2}^{t+\Delta t} + (1+2K)T_{N-1}^{t+\Delta t} - KT_N^{t+\Delta t}
\end{aligned} \tag{2.24}$$

The set of equations can be rewritten in the form of a matrix multiplication (using the boundary conditions described before: T_l=T_2 and T_N=T_{wall}):

$$\begin{pmatrix} T_2^t \\ T_3^t \\ \vdots \\ T_{N-1}^t + KT_{wall} \end{pmatrix} = \begin{pmatrix} 1+K & -K & & & \\ -K & 1+2K & -K & & \\ & & & \ddots & \\ & & & -K & 1+2K \end{pmatrix} \begin{pmatrix} T_2^{t+\Delta t} \\ T_3^{t+\Delta t} \\ \vdots \\ T_{N-1}^{t+\Delta t} \end{pmatrix} \tag{2.25}$$

This is a vector-matrix multiplication of the type:

$$A_i = \mathbf{B}_{ij} C_j \tag{2.26}$$

where the vector A_i contains the data already known at time t, and which includes the boundary conditions. The vector C_j contains the unknown values ($T_i^{t+\Delta t}$) we want to determine. The tensor $\mathbf{B}_{ij}$ converts the two vectors into each other. Because of the similarity of this tensor with the stiffness matrix in linear elasticity, it is sometimes also called the stiffness matrix. All (...) we now need to do to find the unknown temperatures (vector C_j) at time $t+\Delta t$ is to invert matrix $\mathbf{B}_{ij}$ to get:

$$C_i = \mathbf{B}_{ij}^{-1} A_j . \tag{2.27}$$

Although the inversion of matrix $\mathbf{B}_{ij}$ may look daunting, many standard recipes exist to perform the calculation.

One major advantage of the implicit method is that all nodes are used together in the calculation. This means that a point that is m nodes away from a perturbation is immediately included in the calculations, and not only after m steps as in the explicit method.

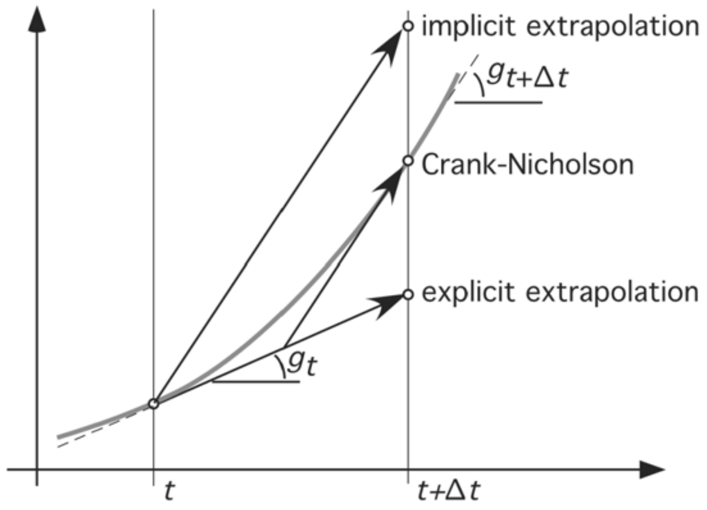


Fig. 2.21 Example of the different Finite Difference methods to predict the value of some function of time. To predict the value at time $t+\Delta t$, the explicit method uses the gradient (g) at time t, whereas the implicit method uses the gradient at time $t+\Delta t$. Neither gives the correct solution. The Crank-Nicholson method combines the two methods, which usually improves the estimate

2.4.4 The Crank-Nicholson scheme

Both the explicit and implicit methods are imperfect. Their disadvantages are illustrated in fig. 2.21. The explicit method uses the gradient (g_t) at time t and extrapolates forward for a small time increment. The implicit method, on the other hand, uses the gradient ($g_{t+\Delta t}$) at time t+Δt for this extrapolation. One way of getting the best of both methods is to use both. This is the so-called Crank-Nicholson scheme, which combines Eqs. 2.15 and 2.23 into:

$$T_i^{t+\Delta t} \approx T_i^t + \left(\frac{\kappa\Delta t}{(\Delta x)^2}\right)\left\{\frac{\left(T_{i+1}^{t+\Delta t} - 2T_i^{t+\Delta t} + T_{i-1}^{t+\Delta t}\right) + \left(T_{i+1}^t - 2T_i^t + T_{i-1}^t\right)}{2}\right\}. \tag{2.28}$$

In many cases, the Crank Nicholson scheme will give a better estimate of the true value after a time increment than the implicit or the explicit method.

2.4.5 Final remarks on the Finite Difference method

Two main FD methods (explicit and implicit) were introduced above, together with one variation, namely the Crank-Nicholson method, which combines both methods. There are various other possibilities to reach the best estimate for the value of a state variable after an increment in the calculation (typically an increment in time). Iterative methods use one estimate to improve the next one, iterating towards an optimised solution. Monte Carlo methods are also possible, where a random number generator randomly selects grid-nodes to update them one by one. The choice of method depends on the problem at hand, as well as on possible numerical limitations and constraints, such as computation time.

In the end, the user must make a sensible choice from the available options. The choice will usually be a compromise in resolution, sophistication of the calculations, computer memory use and computation time. Whatever choice is made, it is important that the user knows and understands the method they are using, and ensures it is appropriate for the problem under consideration. At all times, the model should be checked and compared with analytical solutions, where available, and tested for stability.

Boundary conditions are a critical part of any model, including FD models. Although some boundary conditions are always needed to constrain the solution, boundary conditions can also become an obstacle or dominate the solution. Boundary conditions should therefore be considered very carefully and even avoided where possible. In some cases, boundaries (and thus boundary conditions) can be avoided by using wrapping boundaries.

Another way of avoiding too much interference from boundaries is to move boundaries away from the area in interest. However this enlarges the model and increases the number of nodes, if node spacing or resolution is kept constant. Often, increasing the node spacing, or reducing the resolution, is not permitted because the complexity of the solution in the area of interest. Moving the boundaries away would then result in an increase in memory use and calculation time. There are two ways to solve this problem.

The first solution that can sometimes be employed is using symmetries in the system (Fig 2.22a,b). In the example of the cooling dyke this was already used. Rather than modelling both sides of the dyke, which should both be the same, only one side was modelled. This allows to either double the resolution, while maintaining the same use of memory space and computation time, or keep the same resolution, and halve memory use and computation time.

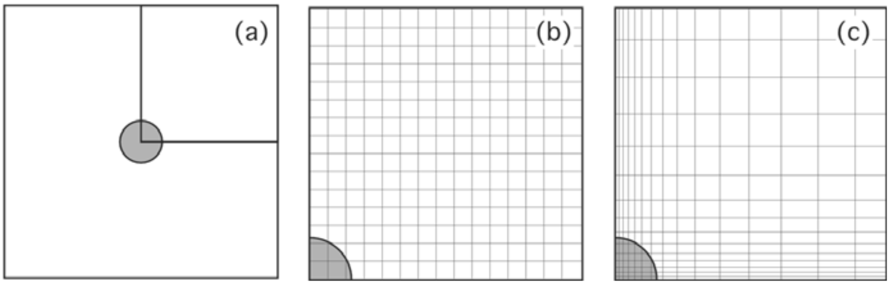


Fig. 2.22 Choosing the optimal grid and boundary conditions. **(a)** Case with a radial symmetric area of interest in the middle. **(b)** Because of the symmetry, only one quarter of the system needs to be modelled. **(c)** Variable grid spacing can optimise resolution in the area of interest

The second, and more powerful solution is to vary resolution, which means varying the spacing between grid nodes (Fig. 2.22c). This way, a high resolution can be achieved in areas where complexity is expected, while other areas are covered by only a sparse grid. Typically, the resolution will be lowest at the boundaries, away from the area of interest. Varying the spacing between nodes, of course necessitates some changes to the equations provided before. These are however minor and obvious. For example, Eqs. 2.9 and 2.11 become (for the forward calculation):

$$\left(\frac{df}{dx}\right)_i \approx \frac{f_{i+1}-f_i}{x_{i+1}-x_i}, \tag{2.29}$$

$$\left(\frac{\partial^2 f}{\partial x^2}\right)_i = \frac{\partial\left(\frac{\partial f}{\partial x}\right)_i}{\partial x} \approx \frac{\left(\frac{f_{i+1}-f_i}{x_{i+1}-x_i}\right)-\left(\frac{f_i-f_{i-1}}{x_i-x_{i-1}}\right)}{\frac{1}{2}\left(x_{i+1}-x_{i-1}\right)}. \tag{2.30}$$

It should be noted, that choosing a high resolution in parts of the system, does have consequences for the choice of time increment, because the requirement that $K<1/2$ (Eq. 2.21) must hold everywhere in the system. This means that the acceptable time increment size is determined by the smallest grid spacing within a model.

Symbols used in Chap. 2.4

A_i	Vector containing known values at all nodes
$\mathbf{B}_{ij}$	Tensor relating known and unknown values at all nodes
C_j	Vector containing unknown values at all nodes
c	Rate constant
$f()$	Arbitrary function
g	Gradient
K	Stability parameter ($\kappa\Delta t/(\Delta x)^2$)
N	Number of nodes
r	Distance from origin of a particle [m]
T_{dyke}	Intrusion temperature of dyke [K]
T_{wall}	Temperature of host rock in which dyke intrudes [K]
v	Velocity [m s^{-1}]
w	Width of a dyke [m]
x	Arbitrary independent variable, in particular position in space
y	Arbitrary dependent variable, or y-coordinate in 2D space
α	Angle
κ	Heat conductivity [m^2 s^{-1}]

2.5 Finite Element method

The Finite Difference method has one major drawback, in that it is not well suited for irregular grids, and that it heavily relies on the assumption that functions can be linearised for small increments. Imagine a model for a deforming piece of rock. At the start we could map the rock onto a regular rectangular grid. However, we may want to keep the grid nodes fixed to particles within that rock. This means that the grid deforms along with the rock, and is no longer rectangular after the first deformation increment. In this case, the Finite Element (FE) method is more suitable than the Finite Difference method. The FE method is the standard method for modelling deforming materials, and is widely used in engineering and studies of rock deformation (Zienkiewicz 1977; Owen & Hinton 1980; Gershenfeld 1999; Knabner & Angermann 2000).

The FE method is based on the following principles:

1. The system to be modelled is divided into a mesh of contiguous (non-overlapping) elements of finite size, for example triangles, hexagons or polygons of arbitrary shape. The elements are defined by nodes, which usually lie on the corners of the elements, but can also lie within the elements. These nodes have two functions: (a) they define the shape of the elements and (b) they store the values of the state variables.
2. We assume that the state variables of interest anywhere in the system can be approximated by certain functions. Such functions can be simple linear functions, but can also be complex non-linear functions.
3. We solve all functions for all nodes in the FE-mesh, taking into account (a) all the state- and kinematic equations, (b) the continuity conditions between the elements, and (c) the boundary conditions.

Since the FE-method is widely used to model solid-state deformation, we will use that application to illustrate the method. Imagine a deforming viscous material. We may be interested in the velocity of material points within that material, resulting from the application of stresses at the surface of the material. In the FE method, we are only interested in the values of the state variables at the nodes. Within the element, the state variables, e.g. velocity in our example, can be described with an unknown function $u_{(x,y)} = \kappa_{(x,y)}(f)$. The velocity is a function of position (x,y) and the "forces" acting on the system (f). These forces can be true forces, but can also include other factors that may determine the velocity. Now consider one element with a number of nodes in the material. At the i-th node, the velocity will be $\kappa_i(f)$. We usually cannot know the true function (κ), but we can assume a trial function (k) that should approximate the true function as much as possible at all nodes:

$$\sum\left(\kappa(f) - k(f)\right)_i^2 \downarrow 0. \tag{2.31}$$

Using the assumed function (k), we can now write for each node i one or more equations of the type:

$$k_i \cdot u_i = f_i. \tag{2.32}$$

This function relates the state variable(s) (u_i, e.g. velocity) to the force (f_i, e.g. stress) through an assumed function (k_i, e.g. a linear function). If we do this for all nodes, and use the boundary conditions, we get a set of equations:

$$\mathbf{K}_{ij} \cdot u_i = f_j. \tag{2.33}$$

$\mathbf{K}_{ij}$ is a matrix, the size of which is determined by the number of nodes in the mesh. Eq. 2.33 looks like the equation that relates elastic strain to stress. $\mathbf{K}_{ij}$ is therefore sometimes called the "stiffness matrix" in analogy to the stiffness matrix for elastic strain. In Eq. 2.33, u_i is unknown. To solve for u_i, we must invert $\mathbf{K}_{ij}$, to get:

$$u_i = \mathbf{K}_{ij}^{-1} \cdot f_j. \tag{2.34}$$

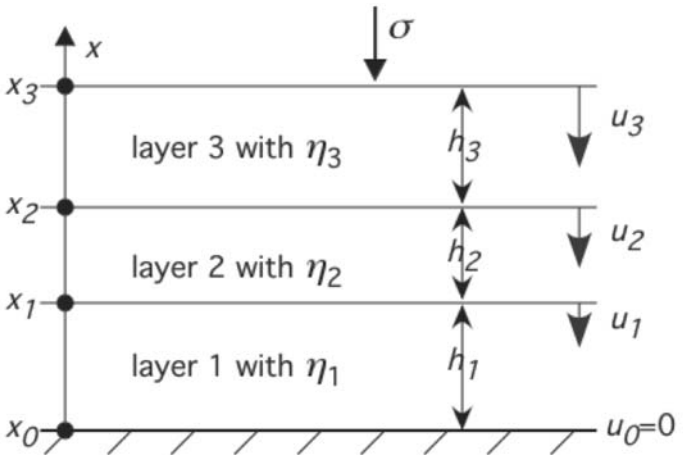


Fig. 2.23 A simple system of three infinite horizontal viscous discs under a vertical stress. The system can be modelled with three 1-dimensional elements and four nodes (black dots)

This is all very abstract, and it may be helpful to use a very simple illustrative example. Take a pile of three infinite homogeneous viscous discs, each having a viscosity η_i and thickness h_i (Fig. 2.23). This system can be modelled with a one-dimensional Finite Element mesh with only four nodes placed at the boundaries between the discs. We want to know how fast the discs will thicken or thin when a stress σ is applied to the pile. If we take tension to be positive, we can use the viscous flow law as a path-independent equation that relates stretching (positive) to stress (σ):

$$\frac{\partial h}{h\partial t} = \frac{\sigma}{\eta} \Leftrightarrow u_i - u_{i-1} = \frac{x_i - x_{i-1}}{\eta_i}\sigma, \tag{2.35}$$

where x_i is the position (height) of the i-th node.

With three elements, we have four nodes, but can only write three equations. We need to add a boundary condition. A useful one might be that the base of the plate does not move: $u_0 = 0$. Let us now write out the three equations in full:

$$\begin{array}{lcl} u_1 - u_0 = \dfrac{x_1 - x_0}{\eta_1}\sigma & & 1\cdot u_1 + 0\cdot u_2 + 0\cdot u_3 = \dfrac{x_1 - x_0}{\eta_1}\sigma \\ u_2 - u_1 = \dfrac{x_2 - x_1}{\eta_2}\sigma & \Leftrightarrow & -1\cdot u_1 + 1\cdot u_2 + 0\cdot u_3 = \dfrac{x_2 - x_1}{\eta_2}\sigma \\ u_3 - u_2 = \dfrac{x_3 - x_2}{\eta_3}\sigma & & 0\cdot u_1 - 1\cdot u_2 + 1\cdot u_3 = \dfrac{x_3 - x_2}{\eta_3}\sigma \end{array} \tag{2.36}$$

This equation is of the type of Eq. 2.33 ($K_{ij} \cdot u_i = f_j$), with:

$$u_i = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix},\ K_{ij} = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & -1 & 1 \end{pmatrix},\ \text{and}\ f_j = \begin{pmatrix} (x_1 - x_0)\sigma/\eta_1 \\ (x_2 - x_1)\sigma/\eta_2 \\ (x_3 - x_2)\sigma/\eta_3 \end{pmatrix}. \tag{2.34}$$

Notice that the "force vector" (f_j) is completely known. If we can invert $\mathbf{K}_{ij}$ to $\mathbf{K}_{ij}^{-1}$, we can solve the velocity vector u_i, using Eq. 2.34 ($u_i = \mathbf{K}_{ij}^{-1} \cdot f_j$). The solution is:

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} (x_1 - x_0)\sigma/\eta_1 \\ (x_2 - x_1)\sigma/\eta_2 \\ (x_3 - x_2)\sigma/\eta_3 \end{pmatrix}. \tag{2.35}$$

The example is of course trivial, but all Finite Element models work basically the same way. There is always a vector of unknowns to be calculated (u), a vector with known "forces" (f) and a "stiffness matrix" ($\mathbf{K}$)

that needs to be inverted. Table 2.1 lists what u and f can be for different processes. The inversion of $\mathbf{K}$ becomes non-trivial when there are many nodes, and the matrix becomes huge. Finding the solution can than become very time-consuming, especially if an exact solution is needed.

Table 2.1 Example of the governing equations, unknown variables and "forces" for processes that can be modelled well with the Finite Element method

Process	Governing equation	Unknown variables (u)	"force" (f)
Deformation	Viscous flow law or Hooke's law	Velocity or displacement	Stress
Conductivity	Fourier's law	Temperature	Temperature gradient
Diffusion	Fick's law	Concentration	Concentration gradient

Symbols used in Chap. 2.5

f	"Forces" acting on the system
h_i	Thickness of element i [m]
$\mathbf{K}_{ij}$	The "stiffness matrix"
$k()$	Trial or assumed function relating state variables to "forces" acting on the system
u	State variable, such as velocity
η	Viscosity [Pa s]
η_i	Viscosity of element i [pa s]
$\kappa()$	True function relating state variables to "forces" acting on the system
σ	Stress [Pa]

2.6 Phase Field modelling

Phase Field (PF) modelling is a lattice-based method that is increasingly used to simulate processes such as crystal growth from a melt or unmixing of a mixed material into two daughter materials. Although the concept and programming of PF-models is relatively simple, the mathematics behind it can sometimes be complex. Here, we will only briefly introduce the very basics of PF-modelling, with a simple example. A short but illuminating description of the basics of PF-modelling can be found in Biben (2005). The book by Emmerich (2003) gives a thorough overview of PF-modelling, but is only suitable for the advanced reader.

The boundary between phases (e.g. mineral grains, or crystal versus melt) is sharp in most modelling techniques. In a q-state Potts model, a point on a lattice has only one of q possible states; for example 1, 2, or 3. There are no points on the lattice that have an intermediate value, such as 2.7. Similarly, polygons in a Front-Tracking model enclose a region with certain properties. The boundary with the next polygon is a sharp line in 2D, or plane in 3D. Such models thus give a very precise position of a phase boundary. PF-modelling instead uses fuzzy boundaries instead. The price to pay for using a PF-model is giving up a bit of resolution in the position of phase boundaries, but in return many possibilities to model processes open up.

In PF-models the variable θ identifies what phase is present at a point in the system. θ can be a scalar if the phase is isotropic. In a simple binary system of, for example, melt with solid, one could define $\theta = -1$ as melt and $\theta = +1$ as solid. θ can be mapped on a lattice, where each lattice point has a certain value of θ. A boundary between solid and melt would be where θ goes from -1 to +1. The main point of PF-models is that θ can have any value, not only -1 or +1, but anything in between as well. This means that a point in the system can have $\theta = -0.5$, which would mean mostly melt, but not completely. Decreasing θ would make the point more melt than solid (= melting), whereas increasing θ would make the point more solid (crystallisation). This shows that the process of melting or crystallisation can be simulated by changing θ. All one needs to find are the right equations for $\partial\theta/\partial t$ for each point on the lattice.

For the model it may be all right to have $\theta = -1/2$, but it is not realistic: the material is either solid or melt, not something in between. To force θ towards either -1 or +1, one can design a free energy (W) function, that relates W to θ. There are some standard $W_{(\theta)}$ functions, such as the Landau-Ginzburg and Kobayashi functions:

$$\text{Landau-Ginzburg: } W_{(\theta)} = \frac{1}{4}\left(1-\theta^2\right)^2, \tag{2.36}$$

$$\text{Kobayashi: } W_{(\theta)} = \frac{\theta^4}{4} - \left(\frac{1}{2} - \frac{m}{3}\right)\theta^3 + \left(\frac{1}{4} - \frac{m}{2}\right)\theta^2. \tag{2.37}$$

The Landau-Ginzburg equation is symmetric around $\theta = 0$ and has minima at $\theta = -1$ and $\theta = +1$. These functions energetically favour particular values of θ, here -1 and +1, that represent the pure phases. Kobayashi's equation has minima at $\theta = 0$ and $\theta = +1$. The factor m determines the relative height of the minima. $m<0$ makes the minimum at $\theta = 0$ smaller than at $\theta = +1$. This would make the state represented by $\theta = 0$ energetically more favourable than that represented by $\theta = +1$. If melt is represented by $\theta = 0$ and solid by $\theta = +1$, melting would occur when $m<0$ and crystallisation when $m>0$.

Any evolution equation that calculates $\partial\theta/\partial t$ will try to minimise $W_{(\theta)}$, that is, will try to minimise the total free energy of the system. Temperature and pressure are other possible factors that may play a role in the calculation of $\partial\theta/\partial t$ for each lattice node in the system. Once the evolution equation(s) have been designed, they can be applied for small time steps, using Finite Difference or Finite Element schemes.

An elegant example of a PF-model for crystal growth from a melt can be found in Biben (2005). The simple code provided by Biben (2005) simulates the growth of a crystal from its melt. The model uses Kobayashi's energy function with an additional surface energy function. The growth rate of the crystal is determined by the difference between equilibrium melting temperature and actual temperature, as well as by the rate at which the latent heat of crystallisation can diffuse away from the growth front. The shape of the crystal is determined by its symmetry and the latent heat. Depending on the setting of the latent heat, one can get all crystal shapes from dendritic to facetted, and with any desired symmetry (Fig. 2.24).

Symbols used in Chap. 2.6

m	Variable defining difference in minima in the Kobayashi equation
w	Free energy
θ	Phase variable

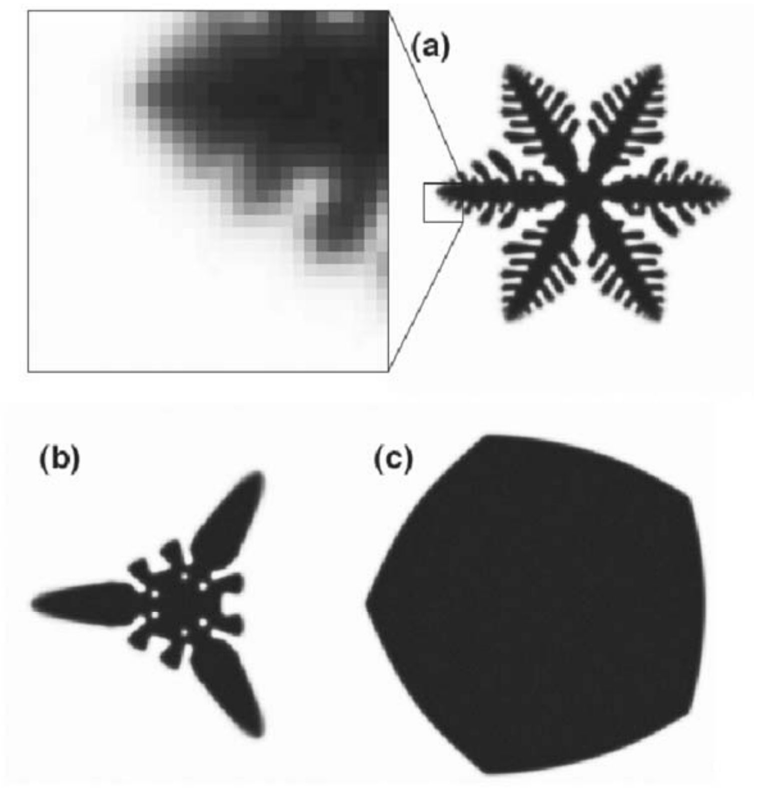


Fig. 2.24 Phase Field simulation of a crystal grown from a melt, using the code of Biben (2005). **(a)** Dendritic crystal with a 6-fold symmetry. Enlargement of the edge of the crystal shows that the boundary is actually fuzzy, because states between "crystal" and "melt" are allowed. **(b)** Crystal produced with the same settings, except that it has a 3-fold symmetry. **(c)** Crystal with a 5-fold symmetry and lower latent heat. EXPERIMENT 3

2.7 Molecular and Dislocation Dynamics

Molecular Dynamics simulation techniques are used to model the interactions of molecules or atoms. Materials are discretised into actual molecules or atoms that are represented by discrete particles in Molecular Dynamics (MD) simulations. Alternatively, materials are treated as a continuum, but contain discrete dislocation line segments in Dislocation Dynamics (DD) simulations, which are treated in the next section. These simulation techniques are very powerful since they give an understanding of the behaviour of materials based on microphysical laws. However, the disadvantage lies in the very small size of single molecules or dislocations so that only very small volumes of materials can be modelled. MD-simulations can produce grain-sizes on the nanometer scale and very short time scales in the range of nano-seconds. DD-simulations may treat larger systems so that the interactions of a set of dislocations can be studied (scale of several micrometers in three dimensions). However, in order to study the plasticity of polycrystalline rocks, interactions of dislocations with grain-boundaries become important so that larger scales are necessary. Some DD-simulations reach that scale but the resolution is still low, especially in 3D. In most cases different models have to be used at different scales and the information from a more precise model at the small scale has to be passed up to the simpler model at the larger scale (short review in Baskes 1999).

2.7.1 Molecular Dynamics

In Molecular Dynamics simulations, material is modelled on the molecular level by a multi-particle approach. The motion of molecules or atoms is expressed by Newton's law of motion. Particles move according to forces acting from neighbouring atoms or molecules. These forces are calculated using inter-atomic potentials, for example the Embedded Atom Method, EAM (Ercolessi and Adams 1994; Mishin et al. 1999), the Johnson pair potential (Johnson 1964) for Fe, or the Lennard-Jones pair potential.

Atoms have a lowest-energy equilibrium distance with respect to each. If they are closer together than the equilibrium distance they repel each other, and if they are further apart from each other they attract their neighbours. The Lennard-Jones pair potential ($U_{(r)}$) for example has the form (Merimaa et al. 2000):

$$U_{(r)} = e\left(\left(\frac{r_0}{r}\right)^{12} - \left(\frac{r_0}{r}\right)^{6}\right), \tag{2.39}$$

where r is the inter-atomic distance, r_0 defines the equilibrium distance (r_e), where $r_e = 2^{1/6} r_0$ and e changes the value of the potential, which is especially important at the equilibrium distance. Figure 2.25 illustrates the Lennard-Jones pair potential where e and r_0 have the value 1.0 and the equilibrium distance is 1.122.

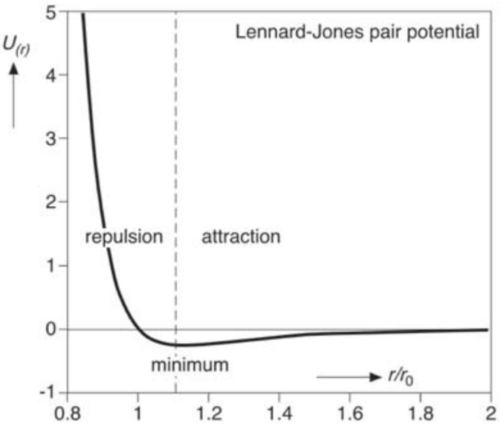


Fig. 2.25 Lennard-Jones pair potential with an equilibrium distance of 1.122 (r_0=1, e=1) using Eq. 2.39. $U_{(r)}$ is the potential and r the distance between atoms. Repulsive forces reach infinity whereas attractive forces go asymptotically to zero with distance

The Lennard-Jones potential is symmetric and thus leads to hexagonal packing of particles (Fig. 2.26a). Atoms in natural materials vibrate if the temperature is not at absolute zero. In Molecular Dynamics models, initial velocities of atoms depend on the temperature and are often applied using the Maxwell-Boltzmann distribution. The direction of these initial velocities may be chosen from a random distribution. These velocities resemble thermal fluctuations and can lead to phase-transitions since fluctuations normally increase with increasing temperature. If the fluctuations are larger than the attracting forces that structure the material, a solid will first become a fluid and then a gas (Fig. 2.26). If the fluid is compressed it may structure again and become a solid.

Molecular Dynamics simulations are used to study a wide variety of natural processes including phase transitions (Kuznetsov et al. 2001), dislocations (Chrzan and Erdonmez 2001; Chang et al. 2002), fracture propagation (Guo et al. 2003, Li et al. 2004), stress-corrosion cracking

(Li et al. 2002), grain boundary sliding and grain boundary diffusion creep (Haslam et al. 2003).

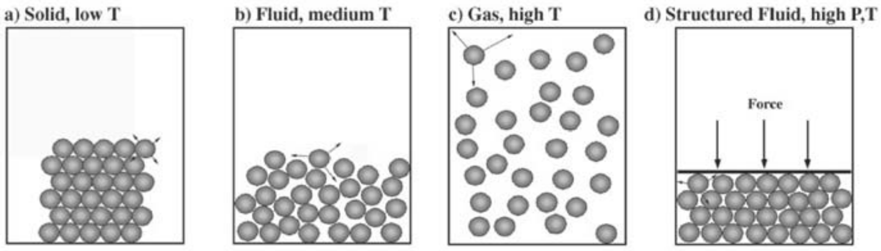


Fig. 2.26 Phase transitions in Molecular Dynamics models, schematic drawing. **(a)** Solid with low thermal fluctuations at low temperature, hexagonal structure. **(b)** Phase transition from solid to fluid due to higher thermal fluctuations at medium temperature. **(c)** Phase transition from fluid to gas due to very high thermal fluctuations at high temperature. **(d)** Pressure can lead to a phase transition from fluid to solid when the fluid is forced to structure and thermal fluctuations are damped

2.7.2 Modelling of dislocations

In order to understand microphysical laws of the flow of rocks in the crust of the Earth one has to study the movement of dislocations. Modelling the deformation of materials based on discrete dislocations allows us to further understand macroscopic aspects of material plasticity. Dislocations are either modelled using Molecular Dynamics approaches (Osetsky et al. 2000), by Front-Tracking methods that are then often called Dislocation Dynamics (codes) (Brown 1967; Kubin and Canova 1990; Kubin et al. 1992; Devincre and Condat 1992; Schwarz 1999; Von Blanckenhagen et al. 2004; Mohles 2004), by Phase Field approaches (Khachaturyan 2000; Wang et al. 2001) or Level Set methods (Xiang et al. 2003). The different methods are explained in detail in the following sections. Recent comparison between the discrete Molecular Dynamics approach to model dislocations and the classical Front-Tracking continuum approach Dislocation Dynamics show similar results (Chrzan and Erdonmez 2001).

One way of modelling dislocations is using Molecular Dynamics as described above, where discrete particles represent single atoms or molecules. Dislocations can be implemented in Molecular Dynamics simulations by displacing a row of atoms according to a displacement field due to external deformation (Fig. 2.27). Dislocations may also develop in these models at crack tips (Zhou et al. 1997) where shear stresses are high so that rows of atoms slip. Figure 2.27 illustrates dislocations that develop at tilt boundaries in stressed crystals. Large parts of the crystal relax stresses and deformation is localized in the tilt boundary by the inclusion of extra rows of atoms that are bounded by dislocations. The main problem in Molecular Dynamics is the very small temporal and spatial scale so that complex interactions of dislocations in polycrystalline solids are difficult to model. However, Molecular Dynamics offers the possibility to model interactions of tangling and annihilating dislocations and interactions of dislocations and grain boundaries.

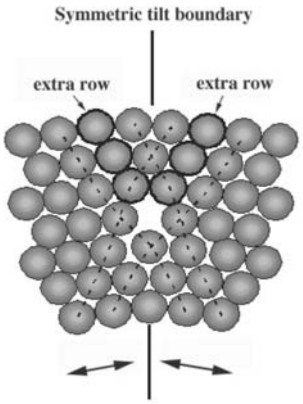


Fig. 2.27 Molecular Dynamics simulation of dislocations in a symmetric tilt boundary. Sketch after Merimaa et al. (2000). Large arrows mark the tilted lattice direction

Dislocation Dynamics is an alternative tha can be used to model dislocations. It is based on the calculation of forces on single dislocations using Front-Tracking methods (Fig. 2.28). The deforming material is treated as a linear elastic continuum with dislocations as line defects. The dislocations can then be represented as nodes connected by straight or curved line segments. The stress acting on a dislocation is a function of the bulk stress due to deformation and the local stress caused by the dislocation itself and possibly an obstacle stress (Foreman and Makin 1966; Brown 1967; Bacon et al. 1973; Kubin et al. 1992; Rhee et al. 1998; Zbib et al. 1998; Schwarz 1999; Ghoniem et al. 2000; Devincre et al. 2001; Von Blanckenhagen et al. 2004; Mohles 2004). The force on a line segment and node of the dislocation can be calculated using the Peach-Koehler formula (Hirth and Lothe 1982). Different approaches exist to calculate the stress tensor at a point within the modelled crystal due to a given dislocation line (Devincre and Condat 1992, Eq. 2.40; Xiang et al. 2003). In addition to the force due to interactions of dislocations, a virtual force due to line tension may be introduced (Devincre and Condat 1992) because the dislocation segments may grow in length and the elastic energy of finite segments depends partly on the length of the segment. The line tension force is short-ranged in contrast to long-range elastic interactions of different dislocation segments. One problem in determining the stress field at a dislocation is that most calculations are based on linear elasticity theory but linear elasticity is thought to break down close to the dislocation itself (Hirth and Lothe 1982).

Dislocation movement is a function of the Peach-Koehler force and can be calculated using a viscous drag law or a mobility tensor. The Peach-Koehler force (*f*) on a dislocation is given by

$$f = \sigma^{tot} b \times \xi, \tag{2.40}$$

where σ^{tot} is the sum of the stresses (bulk stress, self stress, etc.), b the Burgers vector and ξ a tangent to the dislocation line (Xiang et al. 2003). The velocity (v) of a dislocation is simply

$$v = M \cdot f\,, \tag{2.41}$$

where M is the mobility tensor.

Early Dislocation Dynamics simulations focused on the slip of single dislocations on their slip planes and their interaction with obstacles (Foreman and Makin 1966; Bacon 1967; Bacon et al. 1973). More advanced 3D models discretise dislocations into straight edge and screw segments (Fig. 2.28a, Kubin and Canova 1990; Kubin et al. 1992; Devincre and Condat 1992; Devincre et al. 2001) whereas the latest models include dislocations with mixed character (Fig. 2.28b,c, Zbib et al. 1998; Rhee et al. 1998; Schwarz 1999; Ghoniem et al. 2000). The problem with Front-Tracking methods is the complexity of the geometry of dislocations in 3D when dislocations meet and tangle and reach grain-boundaries (Fig. 2.28c). Therefore the calculations are time consuming and need a number of special rules for topology changes when dislocations meet. Some of these rules can be included from Molecular Dynamics studies of dislocation interactions.

The third approach uses a Phase Field method where density functions are used to model three dimensional dislocation arrays (Khachaturyan 2000; Wang et al. 2001, 2003; Hu et al. 2004). These models are based on the Phase Field micro-elasticity theory (PFM). The method allows the tracking of the movement of dislocations of arbitrary orientation without the need to track single dislocation lines. The number of phase fields that are needed to describe the system is given by the number of slip systems in the crystal. Dislocation loops are expressed as thin platelets that represent slipped parts of the crystal and are bordered by the dislocations themselves (for a detailed description see Wang et al. 2003). Phase Field approaches are described in Chap. 2.6. A fourth relatively new approach uses a Level Set method to solve problems with multiple interacting dislocations. This method is also based on linear elasticity and self-stress of dislocations and, like the Phase Field approach, does not need to discretise single dislocations. It is supposed to be more accurate than the Phase Field approaches that use some contributions of energy that are normally not present. For a review of different methods and a detailed description of the Level Set method see Xiang et al. (2003).

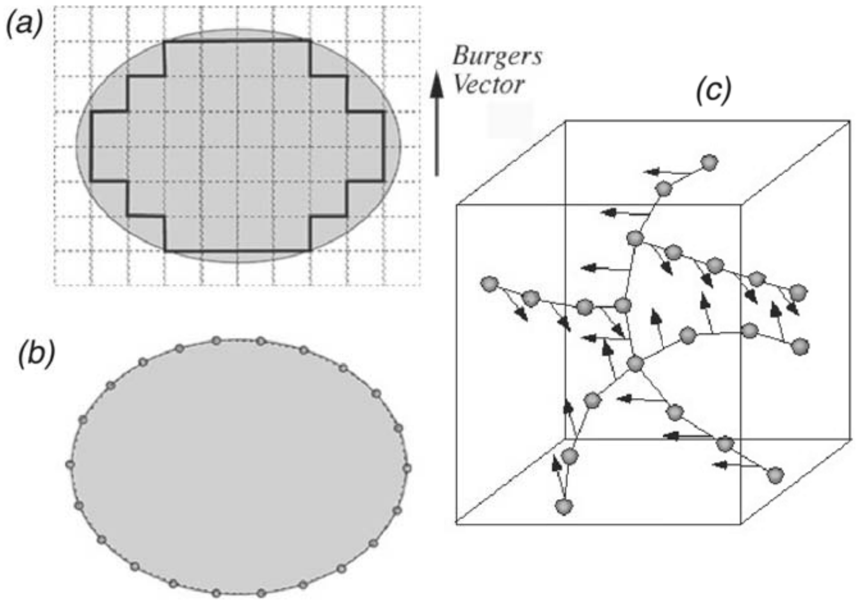


Fig. 2.28 Front-Tracking approach for dislocation modelling. Dislocations are discretised into nodes connected by straight line-segments (Front-Tracking). **(a)** Discretisation on a square lattice where the Burgers vector is parallel to a lattice direction. Dislocation loop (grey part of crystal has slipped) is discretised into screw (parallel to Burgers vector) and edge (perpendicular to Burgers vector) type dislocations. **(b)** Front-Tracking approach with mixed dislocations. Dislocation loop is split up into nodes connected by straight segments. **(c)** Three-dimensional Front-Tracking approach. Arrows mark Burgers vectors that lie in the slip planes. Nodes where dislocations meet need special rules

Symbols used in Chap. 2.7

b	Burgers vector [m]
e	Scaling parameter for the Lennard-Jones potential
f	Peach-Koehler force [N]
M	Mobility tensor of dislocation
r	Distance between atoms [Å]
r_e	Equilibrium distance between atoms [Å]
r	Distance between atoms at which the Lennard-Jones potential is zero [Å]
U	Potential
v	Velocity [m s^{-1}]
ξ	Tangent to dislocation line
σ^{tot}	Sum of stresses acting on a dislocation [Pa]

2.8 Lattice-Spring models

Lattice-Spring models (also called lattice, spring network or discrete element models) are in principal closely related to Molecular Dynamics models (section 2.7). They are also based on atomic models of materials but the basic units in Lattice-Spring models may represent larger volumes of heterogeneities in rocks, namely subgrains, grains or clusters of grains (Flekkoy et al. 2002). These discrete units are connected by rheological elements, where the spring is one of the simplest ones (Fig. 2.29). Coarse lattices may be used to model continuum systems and they are also a close relative to the common Finite Element method (section 2.5). We will first describe Lattice-Spring models in general, then address normal force models that use triangular lattices, introduce angular forces, beams and discrete element models with contact forces, illustrate the average stress tensor and finally introduce the possibilities to model fractures using Lattice-Spring models. An example of the latter can be found in EXPERIMENT 4.

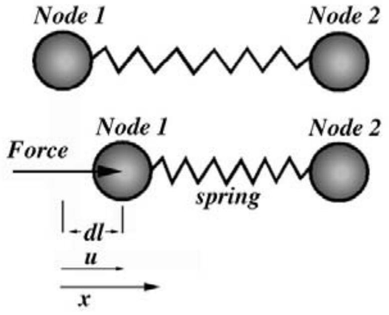


Fig. 2.29 One-dimensional linear-elastic spring as a rheological element. Displacement (u) of node 1 along x is proportional to a spring constant and the applied force. Node 2 is fixed in space. Strain in this simple case is dl divided by the initial length of the spring. The length of the displacement vector u is equivalent to dl

The basic idea of a Lattice-Spring model is that the strain energy in a unit cell of the lattice is equivalent to its continuum in the same volume V of a rock ($E_{cell}=E_{continuum}$). The elastic energy of a Lattice-Spring model is the sum of energies of single bonds in contrast to a volume integral in the continuum model. The relation

$$E_{cell} = \frac{1}{2}\sum_{b}^{|b|}(F \cdot u)^{(b)} \tag{2.42}$$

gives the elastic energy for a Lattice-Spring Model where b denotes the b-th spring, the sum is over all springs, F is the force on spring b and u the displacement of spring b (Ostoja-Starzewski et al. 1996) and

$$E_{continuum} = \frac{1}{2} \int_V \sigma \cdot \varepsilon dV \tag{2.43}$$

is the elastic energy for the corresponding continuum with V the volume, σ the stress, ε the strain and dV volume change. If the model is restricted to linear elastic springs and uniform strain fields these two energies can be used to compare spring constants to different parts of the stiffness tensor of the continuum descriptions (see Ostoja-Starzewski et al. 1996; Flekkoy et al. 2002) and Lattice-Spring models can be mapped onto continuum descriptions (comparison with Finite Element models).

2.8.1 Normal Force model

Triangular lattices have hexagonal unit cells that are connected by six springs to their neighbours (Fig. 2.30). They can be used to model in-plane elasticity in two-dimensions. The corresponding constitutive law is

$$\sigma_{ij} = C_{ijkl}\varepsilon_{kl}, \quad i,j,k,l = 1,2, \tag{2.44}$$

where σ_{ij} is the stress tensor, C_{ijkl} the stiffness tensor and ε_{kl} the strain tensor. If only normal forces along springs are considered in the triangular lattice, central forces on a spring can be described by

$$F_i = k^b n_i^b n_j^b u_j, \tag{2.45}$$

where F_i is the applied force, k^b the spring constant of spring b, n_i^b and n_j^b unit vectors along force and displacement directions of spring b and u_j the displacement vector (Ostoja-Starzewski 2002). If all springs are of unit length, the area of the unit cell is $V = 2\sqrt{3}l^2$ and the stiffness tensor can be related to the spring network by

$$C_{ijkl} = \frac{1}{2\sqrt{3}} \sum_{b=1}^{6} k^b n_i^b n_j^b n_k^b n_l^b \tag{2.46}$$

For a detailed analysis see Ostoja-Starzewski et al. (1996) and Ostoja-Starzewski (2002). If the spring constants are the same, Eq. 2.46 shows that

$$C_{1111} = C_{2222} = \frac{9}{8\sqrt{3}} k \text{, and} \tag{2.47a}$$

$$C_{1122} = C_{2211} = C_{1212} = \frac{3}{8\sqrt{3}} k, \tag{2.47b}$$

so that the modelled continuum is isotropic and there is only one independent elastic modulus (k). In order to vary the Poisson's ratio, angular springs are needed. Angular springs and varying spring constants in different directions allow the modelling of a real planar anisotropy with six independent parameters.

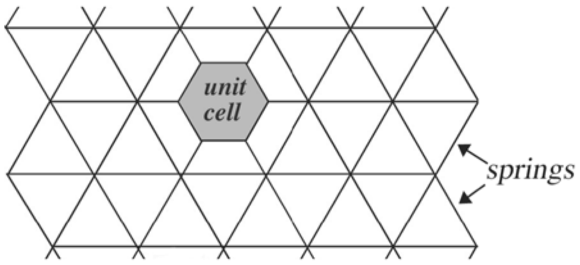


Fig. 2.30 Triangular Lattice-Spring model with hexagonal unit cells. Each cell is connected with six possible neighbours by rheological elements, for example linear elastic springs

2.8.2 Normal and Angular Force models

In order to apply angular forces, angle changes between neighbouring springs in a unit cell are considered. In this case the network consists of six normal springs and six angular springs per unit element (Fig. 2.31). Normal springs are represented by a spring constant k and change their length (l), whereas angular springs have a spring constant α and deform by changing the angle between two neighbouring normal springs (Fig. 2.31b,c). If the symmetry of the initial cell is kept this configuration gives six independent constants ($k(1),k(2),k(3),\alpha(1),\alpha(2),\alpha(3)$), since $k(1) = k(4)$, $k(2) = k(5)$, $k(3) = k(6)$, $\alpha(1) = \alpha(4)$, $\alpha(2) = \alpha(5)$ and $\alpha(3) = \alpha(6)$. The elastic energy (E^b) of an angular spring (b) is then

$$E^b = \frac{1}{2}\alpha^{(b)}|\Delta\chi|^2, \qquad (2.48)$$

where $\alpha^{(b)}$is the angular spring constant of spring b and $\Delta\chi$ the change in angle between the two normal springs that are connected by the angular spring.

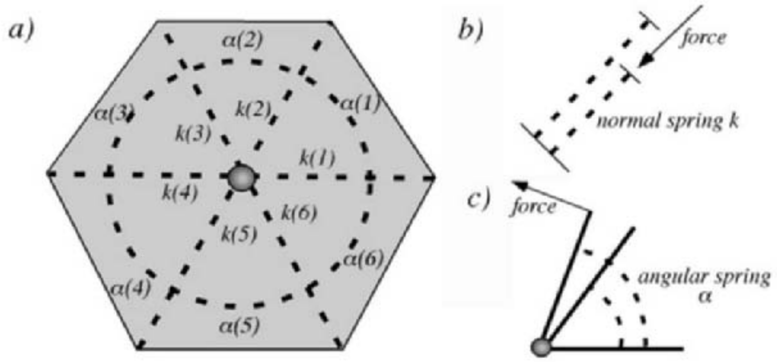


Fig. 2.31 (a) Springs in a hexagonal unit cell of a network with angular and normal springs. (α) and (k) denote spring constants of angular and normal springs. **(b)** Normal springs change their length parallel to the spring. **(c)** Angular springs change the angle between two neighbouring normal springs

If we assume that all normal spring constants have one value and all the angular spring constants another value, then the material is isotropic and is described by two elastic constants. The spring constants and the stiffness matrix are related by the following equations (Ostoja-Starzewski 2002):

$$C_{1111} = C_{2222} = \frac{1}{2\sqrt{3}}\left(\frac{9}{4}k + \frac{1}{l^2}\alpha\right), \qquad (2.49a)$$

$$C_{1122} = C_{2211} = \frac{1}{2\sqrt{3}}\left(\frac{3}{4}k - \frac{9}{4l^2}\alpha\right), \qquad (2.49b)$$

$$C_{1212} = \frac{1}{2\sqrt{3}}\left(\frac{3}{4}k + \frac{9}{4l^2}\alpha\right). \qquad (2.49c)$$

Such a model has planar bulk modulus (κ) and shear modulus (μ) of

$$\kappa = \frac{1}{2\sqrt{3}}\left(\frac{3}{2}k\right),\ \mu = \frac{1}{2\sqrt{3}}\left(\frac{3}{4}k + \frac{9}{4l^2}\alpha\right), \qquad (2.50)$$

and a Poisson's ratio (v) that can vary between 1/3 and -1:

$$v = \frac{1 - \frac{3\alpha}{kl^2}}{3 - \frac{3\alpha}{kl^2}}. \tag{2.51}$$

These models are also referred to as the Kirkwood (Kirkwood 1939) and Keating model (Keating 1966), where the Keating model uses a different calculation of the energy stored in angular bonds.

2.8.3 Elastic Beam models

In a beam model unit cells in the lattice are connected by beams that have a normal force, a shear force and also a bending moment (Fig. 2.32). These so-called micropolar media are described by fields of force-stresses and moment-stresses so that rotations of network nodes are considered (these are not considered in the angular/normal force models). If a beam between site i and j is considered, each lattice site contains three degrees of freedom, the two displacement vectors (u_x^i,u_y^i) and a bending angle θ^i. The beam has a cross section A^{ij} and a length l^{ij}. Between sites i and j the normal (a^{ij}), shear (b^{ij}) and bending (c^{ij}) flexibilities of the beam are (D'Addetta et al., 2001)

$$a^{ij} = \frac{l^{ij}}{E_b A^{ij}}, \quad b^{ij} = \frac{l^{ij}}{G_b A^{ij}} \quad \text{and} \quad c^{ij} = \frac{{l^{ij}}^3}{E_b I^{ij}}, \tag{2.52}$$

where E_b is the Young's modulus, G_b the shear modulus of the beam and I^{ij} is the moment of inertia of the beam for flexion. Between sites i and j there is a longitudinal force acting at site i of

$$F_{b,x}^i = \frac{1}{a^{ij}}\left(u_x^j - u_x^i\right), \tag{2.53}$$

a shear force of

$$F_{b,y}^i = \beta^{ij}\left(u_y^j - u_y^i\right) - \frac{\beta^{ij} l^{ij}}{2}\left(\theta^i + \theta^j\right), \tag{2.54}$$

and a flexural torque at site i

$$M_{b,z}^i = \frac{\beta^{ij} l^{ij}}{2}\left(u_y^i - u_y^i + l^{ij}\theta^{ij}\right) + \delta^{ij} {l^{ij}}^2 \left(\theta^j - \theta^i\right), \tag{2.55}$$

where $\beta^{ij} = 1/(b^{ij} + 1/12c^{iij})$ and $\delta^{ij} = \beta^{ij}(b^{ij}/c^{ij} + 1/3)$ (D'Addetta et al. 2001).

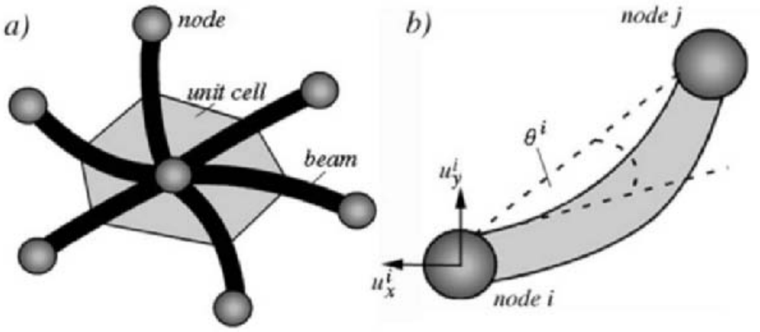


Fig. 2.32 (a) Unit cell connected to neighbours by elastic beams. These beams carry normal and shear forces as well as a torque. The displayed unit cell is deformed. **(b)** Beam between node *i* and *j*. Node *i* has three degrees of freedom, two displacement vectors and a bending angle

Beam networks can be used in ordered lattices (for example the triangular lattice) but they may also be used in random lattices with for example a variation in the size of the basic units (D'Addetta et al. 2001). A review of beam networks and mapping onto continuum descriptions is given in Ostoja-Starzewski (2002).

2.8.4 Discrete Element models with contact forces

Classical Discrete Element models are used to model granular aggregates. In this case contact forces are considered between different particles that may be circular or polygonal shaped (Cundall and Strack 1979; Bathurst and Rothenburg 1988). These forces may be normal or shear forces as well as contact moments (Kuhl et al. 2001; Mühlhaus et al. 2001). There are strong similarities between the Lattice-Spring and Beam models and the Discrete Element models with contact forces. In real granular media the contact forces are only compressive so that the material has no real cohesion in contrast to Lattice-Spring or Beam models. However, the introduction of a yield criterion and cohesion in a Discrete Element model is straightforward (Mühlhaus et al. 2001), which leads to descriptions where the Lattice-Spring or Beam models and the Contact Force models are basically identical. In addition, contact forces and moments may not only be elastic, but may contain a viscous part so that a real time-scale is introduced in the model. The simplest models use circular or spherical particles with normal (f_N^c) and tangential (f_T^c) contact forces following the lines of Hertz (1881)

$$f_N^c\left(n^c\right)=k_N\Delta l_N^c,\ \text{and}\ \ f_T^c\left(n^c\right)=k_T\Delta l_T^c, \tag{2.56}$$

where n^c is a unit vector pointing from a particle to its neighbour, k_N and k_T are the normal and tangential contact stiffness respectively and Δl_N^c and Δl_T^c the normal and tangential contact displacement respectively (Kuhl et al. 2001). The contact force (f^c) is made up of the normal and tangential force (Fig. 2.33)

$$f^c\left(n^c\right)=f_N^c n^c+f_T^c. \tag{2.57}$$

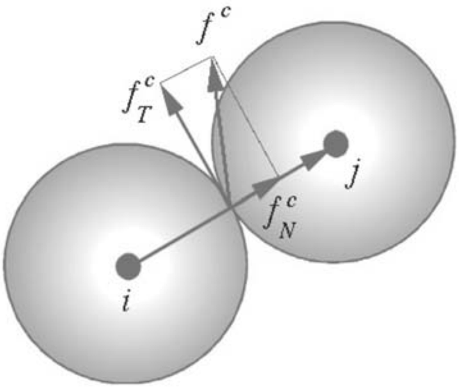


Fig. 2.33 Contact forces (f^c) between a particle i and its neighbour j. f_N^c and f_T^c are normal and tangential force respectively

In Fig. 2.33 it becomes obvious that the normal contact force may be identical to a normal spring in the Lattice-Spring model and the tangential contact force can be mapped on an angular spring. Beam normal and shear forces are similar to normal and tangential contact forces and the beam moment can be compared to the contact moments of discrete particles. Quite often the total contact forces and moments are the sum of an elastic and viscous part, which introduces normal, tangential and rotational viscosities in addition to the elastic stiffnesses. The viscous part introduces a real time and length scale. These viscosities may have a real physical significance for visco-elastic contacts or in fast granular flow.

For quasi-static problems the significance of the viscosities is purely numerical (Mühlhaus et al. 2001). The average macroscopic stress tensor within a Discrete Element model is determined by a principle of virtual work. It is assumed that the overall macroscopic virtual work and the virtual work of the granular assembly are equivalent (Kuhl et al. 2001). The discrete macroscopic stress tensor has the following form

$$\sigma = \frac{1}{V}\sum_{c \in V}\left[f^c \otimes l^c\right], \tag{2.58}$$

where σ is the stress tensor, V a representative volume, c the contacts within this volume, f^c the contact forces and l^c a unit vector pointing from a particle to its neighbour (Kuhl et al. 2001). In order to attain a continuous form of the stress tensor this discrete form of the stress tensor can be transformed into an integral form if a large enough representative volume (V) is chosen (see Kuhl et al. 2001 and D'Addetta et al. 2001 and references therein for a complete description). Moments of beams or contact moments of particles are generally not taken into account when the stress tensor is calculated.

The Poisson ratio (ν) of a Discrete Particle model with normal and tangential contact forces is (Kuhl et al. 2001)

$$\nu = \frac{k_N - k_T}{4k_N + k_T}, \tag{2.59}$$

where k_N and k_T are the contact stiffness. The Poisson's ratio can thus vary from -1 to 1/4.

2.8.5 Fracturing or failure

Springs in a Lattice-Spring model may break by different failure criteria that range from a critical energy to a critical force/strain that is reached within the spring. Normally, springs break under tension, which has to be specified when a critical energy criterion is used. The spring is simply removed from the lattice once it reaches the critical value. The lattice units may still have a compressive component, but the cohesion is lost when failure occurs. It can be shown that spring failure criteria lead to macroscopic Griffith scaling of fracture strength with crack length under tension if pre-existing cracks are present in the model (Jagota and Bennison 1995). However, the initial fracture length is always given by the actual length of the springs used and thus the resolution of the model. This means that if the failure criterion is fixed to represent the correct critical energy release rate for long cracks, the model behaves as if it was filled initially with micro-cracks of the size of single springs. Therefore different criteria may be used for fracture development at internal boundaries (around a fracture or hole) and within the rest of the lattice. One may also add a quenched noise to the breaking strengths of springs in order to model a heterogeneous solid. This will, however, alter the rheological behaviour of the solid and the localization of fractures.

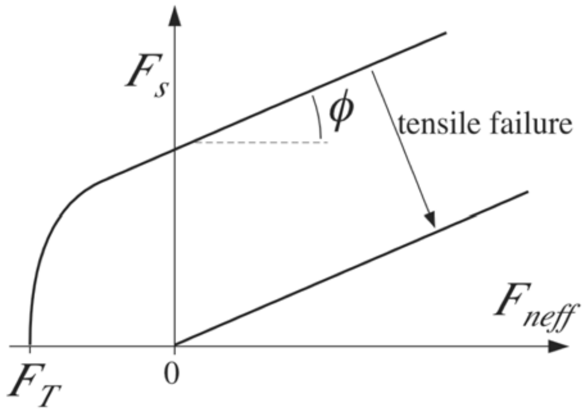


Fig. 2.34 Failure criterion for a discrete element model after Mühlhaus et al. (2001). Failure is either tensile expressed by the tensile strength f_T or by shear determined by the angle of friction ϕ and the tensile strength. Once the material fails under tension the cohesion is lost

Yield criteria can also be used in Discrete Element models with contact forces. In this case a critical tensile strength may be reached for the normal force component and a Mohr-Coulomb friction law is used for the tangential force component. In addition, the particle contact moment may be added (see Mühlhaus et al. 2001). Once the tensile strength of the normal component is reached and a tensile fracture occurs at the contact, the initial yield and tensile force are put to zero (Fig. 2.34). Contacts loose cohesion if

$$F_{neff} \le f_T, \tag{2.60}$$

and slip if

$$F_s \ge \tan\phi F_{neff}, \tag{2.61}$$

where F_{neff} is the effective normal force that depends on the normal force at the contact and the contact moment (Mühlhaus et al. 2001), f_T is the tensile strength, F_s the shear force at the contact and ϕ Coulomb's angle of friction.

Symbols used in Chap. 2.8

A	Cross section of beam [m^2]
a^{ij}	Normal flexibility of beam between sites i and j [m N^{-1}]
b	Spring index number
b^{ij}	Shear flexibility of beam between sites i and j [m N^{-1}]
C_{ijkl}	Stiffness tensor [Pa]
c^{ij}	Bending flexibility of beam between sites i and j [m N^{-1}]
E	Young's modulus [Pa]
E_{cell}	Strain energy of a cell [J]
$E_{continuum}$	Strain energy of a continuum [J]
F	Force on a spring [N]
F_{neff}	Effective normal force on a contact [N]
F_S	Shear force on a contact [N]
f_N^c, f_T^c	Normal and tangential contact forces acting on particles [N]
f^c	Contact forces acting on particles [N]
f_T	Tensile strength [Pa]
G	Shear modulus of beam [Pa]
I^{ij}	Moment of inertia of a beam for flexion [m^4]
k	Spring constant [Pa]
k_N, k_T	Normal and tangential contact stiffness [Pa]
Δl_N^c, l_T^c	Normal and tangential contact displacement [m]
l	Spring length [m]
M	Flexural torque on a beam [Pa]
n	unit vector
u	Displacement (vector) of a node [m]
V	Volume [m^3]
α	Spring constant of angular spring [Pa]
β^{ij}	Auxiliary variable used in Eqs. 2.54 and 2.55 [N m^{-1}]
δ^{ij}	Auxiliary variable used in Eq. 2.55 [N m^{-1}]
ε, ε_{kl}	Strain tensor
θ	Bending angle of a beam [rad]
κ	Bulk modulus [Pa]
μ	Shear modulus [Pa]
v	Poisson's ratio
σ, σ_{ij}	Stress tensor [Pa]
ϕ	Coulomb's angle of friction [rad]
χ	Angle between two normal springs that are connected by a normal spring [rad]

3 Microprocess Simulations

Editors: Daniel Koehn, Mark W. Jessell and Paul D. Bons

This chapter describes a range of microdynamic processes that have been implemented in the software *Elle* or related codes. The first section describes the software *Elle*, which is the main software used in this book. The *Elle* software is included on the CD.

Each subsequent section deals with a single process that may affect the microstructure of rocks, for example, grain growth or viscous deformation. Each section describes the process, how it can and has been modelled and finally it shows an example of how it is implemented in *Elle*. Most of the examples given in this and the next chapter can be run from the Data Structure files and scripts on the CD (Appendix B). The reader is encouraged to first run these examples to achieve the same results as presented in the figures in this chapter. Next, the reader can start to change some of the input-parameters to get acquainted with the code that simulates each particular process.

The reader is referred to Appendix H for a number of miscellaneous processes and utilities that are useful or necessary to carry out simulations with *Elle*, such as changing the distances to nodes, or reformatting the Data Structure to make it suitable for certain modules.

3.1 Introduction to *Elle*

Lynn Evans, Mark W. Jessell, Paul D. Bons and Daniel Koehn

3.1.1 The *Elle* Project

This chapter describes the principal components of the *Elle* simulation platform, together with a description of several different microstructural processes that have been simulated using this system. Finally we describe a few processes, which have been simulated using other systems, as equivalent *Elle* processes are not currently available and the fundamental data structures are sufficiently close to the *Elle* Data Structure to allow integration into the system in the future.

The example processes described in this chapter should not be taken to represent either the definitive simulations for a specific process, as we are refining them continuously, nor should they be taken to be an exhaustive list of all possible processes, as several processes (such as twinning) have not been considered at all at the time of writing. For the latest version of *Elle* go to http://www.microstructure.info/elle.

The aim of the *Elle* Project is to develop a generalised framework for the numerical simulation of the evolution of rock microstructures during deformation and metamorphism. This framework consists of four components (Fig. 3.1.1):

1. A numerical description of a 2D microstructure, also known as the Data Structure, which allows us to describe the geometrical distribution of physical and chemical properties.
2. Base Libraries that handle low-level interactions with the microstructure, which allow us to interrogate and modify this microstructure.
3. Precompiled *Elle* binaries (Processes and Utilities), which use the Base Libraries to calculate local driving forces and the resulting changes in microstructure to simulate the activity of individual microstructural processes, and Utilities functions that help us to create or modify the microstructure in non-physical ways.
4. A series of text files including an *Elle* File that stores the complete microstructure, together with various Experiment Scripts that describe the boundary conditions and global activity of individual processes for a single microstructural experiment, together with the controls on how the microstructure is graphically displayed.

At the simplest level a user can create a new microstructure, then run precompiled *Elle* binaries and change parameters in existing processes. At the next level of sophistication a user can use small scripts as control programs to run experiments with coupled processes. Finally a user can program or add their own processes or change existing processes and link them with the Base Libraries. *Elle* is written in C and C^{++} with small Fortran parts in order to link *Elle* with the Finite Element code named "*Basil*" (Ch. 3.8), which is written predominantly in Fortran.

In this project we had to choose between defining a Data Structure that was ideally suited to one specific process, and one that was sufficiently general that it could be adopted to simulate a whole range of physical and chemical problems. Since we intended to solve problems where there was a coupling between processes, we have opted for the second choice.

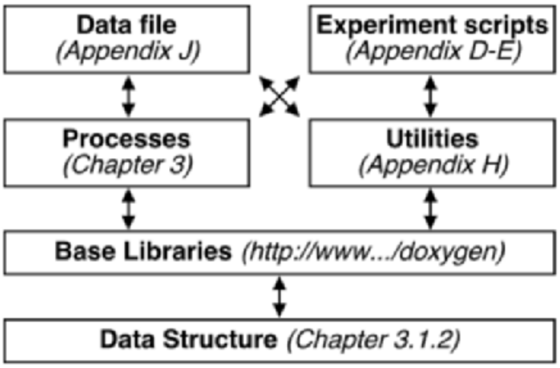


Fig. 3.1.1 The *Elle* simulation Framework, showing the interactions between different components

3.1.2 The *Elle* Data Structure

The *Elle* Data Structure provides the framework for describing the distribution of physical and chemical properties in a 2D microstructure. The three fundamental elements of the *Elle* Data Structure, shown in Fig. 3.1.2, are:

1. Boundary Nodes or 'bnodes' - the nodes which define the local position of grain or phase boundaries;
2. Grains or 'flynns' - the polygons defined by bnodes enclose whole grains or subgrains, and may be nested to represent subgrains;
3. Unconnected Nodes or 'unodes' - the interior nodes that define chemical and physical properties within grains.

All other elements within the *Elle* Data Structure, such as triangulations, are calculated from these basic elements, and are not stored in the *Elle* File. The list of physical and chemical properties that may be assigned to these elements is continually growing but includes attributes such as mineralogy, chemical concentration, lattice orientation, the stress and strain states, temperature, and dislocation density. The boundaries of the model are cyclic, to remove edge effects.

Microstructural processes in *Elle* either modify the position of a bnode or unode, or change the value of an attribute of a bnode, unode or flynn. Unodes are material points, and thus only move in response to deformation, whereas bnodes simply define the limit of a single phase and thus may migrate through the material.

Boundary Nodes (bnodes)

Bnodes define the local position of a grain or phase boundary and possess two primary attributes, namely their (x,y) location, with an origin in the lower left, and their connectivity to other bnodes. We assume that a maximum of three phases may meet at a point, which means that all bnodes either have two or three connected neighbours (Fig. 3.1.2). This restriction was enforced to simplify the topological checks that have to be applied following bnode movement, but can be reasonably justified in micro-structural terms by the paucity of four-way boundary junctions seen in nature due to the high surface energy states that arise for this case (Anderson et al., 1984).

Secondary attributes, such as grain boundary width, the chemical composition of a grain boundary fluid, and the local stress state may also be stored with each bnode. The values of these properties at any point along a grain boundary between bnodes may only be estimated by interpolation, so that the spacing of the bnodes provides a fundamental limit to the spatial resolution of the model. If bnodes are closely spaced the resolution is higher and the calculation will be more accurate, but the calculation time will increase. If bnode spacing is increased the calculation time will decrease, but the solution will become less accurate. The bnode-spacing during an experiment is kept within a fixed range so that the spatial resolution does not vary much within the modelled domain. As new bnodes are removed or created by the program, intermediate attributes are interpolated from the two neighbouring bnodes. Note that the bnode-spacing can be easily changed in *Elle* even during a simulation. It is always useful to perform runs with different resolutions in order to reach an optimal bnode spacing for accuracy and calculation time.

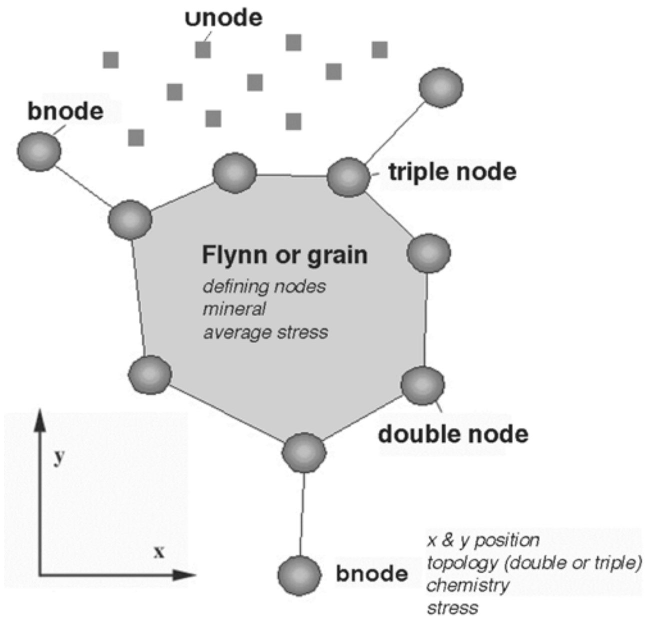


Fig. 3.1.2 Basic *Elle* Data Structure. A partial microstructure showing the three basic elements. bnodes (shaded circles) connected by straight boundary segments define grains or ‘flynns’. Bnodes and flynns can posses any number of physical or chemical properties. unodes (squares) provide information on the distribution of physical and chemical properties within grains

Two types of secondary checks need to be made as a result of bnode motion. The first type is a topological check, to make sure that a moving bnode has not crossed another grain boundary. If it has, a local readjustment of the topology of the grain boundary network needs to be made. The second type is related to bnode spacing, to ensure that the bnode spacing has not locally become too large, in which case a new bnode with interpolated properties is added; or too small, in which case a bnode is removed and the properties of neighbouring bnodes are adjusted accordingly.

Grains (flynns)

The term “flynn” is used to refer to any polygonal domain in *Elle*. These flynns are non-overlapping and as a complete set are space-filling (Fig. 3.1.3a). Flynns represent closed polygons of material with similar properties, and the limits of a flynn are defined by a list of bnodes. These flynns may have spatially uniform properties (such as mineralogy), in which case the property is stored with the flynn, or spatially varying properties (such as temperature), in which case the properties can be stored at the unode level. The flynns may be arranged in a two-level hierarchy, with a polygonal grain structure at the highest level of this hierarchy and grains divided into subgrains or subflynns at the second level (Fig. 3.1.3b).

We chose not to simply call these polygons grains for two reasons, firstly, the hierarchical system allows them to define subgrain areas, and secondly they may also represent non-crystalline materials such as melt pockets and voids. If required, the polygonal flynn network can be temporarily triangulated using a Delaunay triangulation routine (Section 2.1.1) using the bnodes to limit the triangulations, so that the triangles do not cut grain boundaries (we use the code described in Shewchuk 2002). This Delaunay triangulation may be used by the Finite Element deformation programs *Basil* and *OOF* that are linked to *Elle*. Using the Delaunay triangulation we can also specify a minimum internal angle for the triangles (i.e. no "flat" triangles). Decreasing the minimum angle constraint allows flatter triangles, which stops *Triangle* creating many smaller higher quality triangles. It is this and increasing the maximum area constraint, which allows fewer, larger triangles and reduces the total number of elements, thus decreasing the calculation time.

Unconnected Nodes (unodes)

Unodes provide a finer resolution of physical and chemical properties then may be defined at the flynn level. Unodes always posses an (x,y) position attribute in addition to a wide range of other physical or chemical attributes depending on the experiment. Unodes may be uniformly or randomly distributed through the microstructure, depending on the needs of the experiment. If unodes are used as connected particles they can be used to model elastic or visco-elastic deformations and they can change their volume, state and dissolve or precipitate. For these models, the *Elle* boundaries may be connected to the unodes so that *Elle* polygons deform with the particle lattice. Unode sites may also be used as grids for Cellular Automata in which case they may be partly connected to the *Elle* polygons and can be used for example for subgrain growth simulations using a Potts model (Ch. 2.4). When information needs to be transferred between unodes and bnodes, special, temporary, unodes are placed within the lattice immediately adjacent to each bnode.

Each unode represents the properties in its local area; however there are three additional techniques of temporarily dividing up a single grain, as shown in Fig. 3.1.3. For the sake of clarity, in this figure only one grain (z) is shown with its unodes, although normally all grains would posses them. The three unode-based subdivisions of a grain are Delaunay Triangulations, Voronoi Tessellations, and as Circular Regions of Interest.

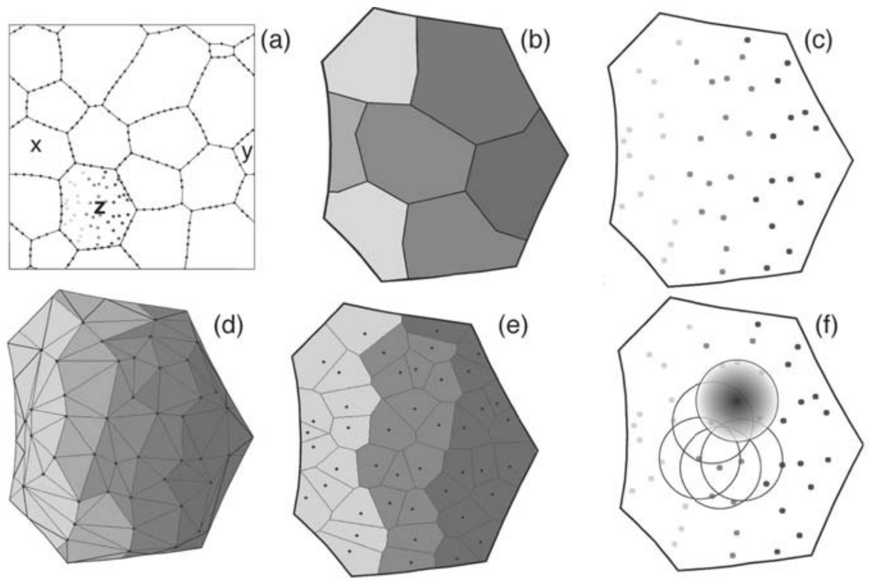


Fig. 3.1.3 Different ways of achieving a fine spatial resolution within grains. (**a**) A 2D microstructure described by bnodes, which in turn define closed polygonal boundaries. Note that the *Elle* Data Structure allows for cyclic boundaries, so that grains *x* & *y* are in fact parts of the same grain. unodes are shown only within grain z for clarity, and are shaded according to some property. (**b**) Hierarchical division of grain z into subgrains, each with uniform properties, ignoring unode values. (**c**) Enlargement of grain z, showing distribution of unodes. (**d**) Delaunay triangulation of grain z based on unode distribution only, each triangle within this grain now has properties based on an interpolation of the values of its three apical unodes. (**e**) Voronoi tessellation of grain z, based on unode distribution, Voronoi cells have uniform properties based on central unode value. (**f**) Overlapping circular regions of interest around unodes, for 5 of the unodes in grain z, one of which is shaded to show distribution of attribute within circular region. The value at any specific location is the weighted sum of values within each region of interest that overlaps that location

Delaunay Triangulations

Just as a triangulation may be made using bnodes, we can construct a triangulation that additionally takes into account the distribution of unodes within a grain (Fig. 3.1.3d). Triangle properties will need to be interpolated from their apical unodes, and after a micro-process calculation is completed, these values then need to be mapped back onto the unodes to be stored.

Voronoi Tessellations

Voronoi Tessellations are calculated using the same code, which generates the triangulations, and it produces a polygonal network of cells with the property that each point within a cell is closer to the central unode than any other unode (Fig. 3.1.3e). In this case the properties are assumed to be uniform within a Voronoi Cell and are taken directly from the central unode.

Circular Regions of Interest (ROI)

The circular regions of interest around a unode provide a simple method of estimating the value of a property at an arbitrary (x,y) location (Fig. 3.1.3f). Each region of interest (ROI) is assumed to have a cosine-shaped bell weighting factor that decays away from the unode. To calculate the value at a particular location, we sum the products of each weighting factor and the unode value. This is only valid if the unodes are evenly spaced and the ROI chosen so that the sum of the weights is similar at any point.

3.1.3 Base Libraries

The Base Libraries provide the low-level programming tools needed to define a microstructural process, but they do not define processes in their own right. Detailed information, including the source code, is provided on the CDROM associated with this book or at http://www.microstructure.info/doxygen. The principal Libraries used in *Elle* are:

1. **libelle:** This library handles access to the *Elle* Data Structure, including reading and writing *Elle* Data Files, and reading and modifying the attributes of specific elements of the Data Structure.
2. **Libclip:** This library handles specialised clipping algorithms used in handling flynns. It is based on the General Polygon Clipper library developed by Alan Murta (http://www.cs.man.ac.uk/~toby/alan/software/).
3. **Libmatrix:** This library handles simple matrix operations.
4. **libplot_x, libplot_wx:** These libraries handle the display of *Elle* microstructures using the X/Motif and wxWidgets libraries respectively (http://www.wxwidgets.org).
5. **libps:** This library handles writing of Postscript files.

3.1.4 Precompiled Binaries

There are two types of precompiled binaries available in the *Elle* system, ‘Processes’ that aim to reproduce some or all of the behaviour associated with a single microstructural process, and ‘Utilities’ that modify an *Elle* File in non-physically based ways. The majority of this chapter deals with *Elle* processes, and a short description of various *Elle* Utilities is given in Appendix H.

3.1.5 The *Elle* Data File

An *Elle* Data File is a text file that contains a complete description of an *Elle* file at a given instant in time. A description of the format of an *Elle* Data File is given in Appendix J.

3.1.6 Experiment Scripts

An *Elle* experiment consists of the calculation of the interaction of a set of locally-defined driving forces and micro-processes, calculated for small time steps, which is intended to simulate the behaviour of a rock at the grain scale for a given set of boundary conditions. *Elle* experiments are controlled by experiment scripts, which together with a starting *Elle* File, define a single *Elle* experiment. This experiment script controls the evolution of extrinsic variables, such as temperature, defines boundary conditions, deformation histories and determines which micro-processes will be involved by controlling the order and rate of execution of individual process algorithms (Fig. 3.1.4). Existing code may be used as a micro-process e.g. the non-linear viscous deformation program *Basil* (Barr and Houseman 1996). Conversion Utilities may be needed to translate the *Elle* data structure into the appropriate input format for the algorithm and incorporate the output back into the data structure.

One of the fundamental assumptions of the *Elle* system is that if we use small time steps we can approximate a set of synchronous processes by calculating their effects sequentially (this approach is known as Operator Splitting). One loop through all the processes thus represents a single experimental time step. In order to satisfy the numerical constraints of a particular numerical method, certain processes may need to define a shorter time step, and that the sum of these shorter steps then equals the overall experiment time step. It has been suggested that the order in which the process is calculated could lead to significant changes in the microstructural evolution; however in any tests we have performed so far, this has not been the case. If significant divergences in solutions did occur, it would simply suggest that the experimental time step was too large, which can of course be a problem in any numerical scheme.

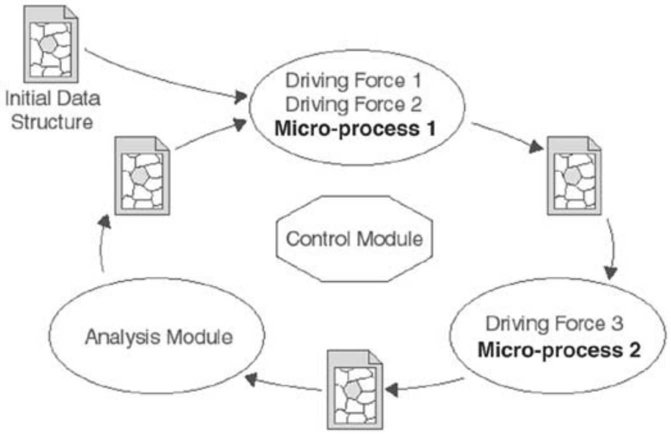


Fig. 3.1.4 The initial microstructure is defined by the user, and a central control script determines which processes will be applied to the microstructure. Different micro-processes may alter the microstructure depending on different driving forces. An analysis module can be included to save statistical data

3.1.7 Microprocess Simulation

The following sections describe the algorithms, which we have used to simulate microprocesses in rocks. These sections have been grouped according to how the Data Model is affected:

1. The processes described in Sections 3.2 to 3.4 describe those processes that simply change the values of bnodes, unodes or flynns, but not their position or topology.
2. Sections 3.5 to 3.7 describe those processes that additionally change the positions of bnodes and hence flynns. These processes may lead to the creation or removal of grains. This includes the codes *Vein Growth* and *Fringe Growth*, which are not currently part of the *Elle* platform.
3. Sections 3.8 to 3.10 are those processes directly associated with polycrystalline deformation. This includes *DiffForm*, which is not currently part of the *Elle* platform.
4. Sections 3.11 to 3.13 describe the implementation of discrete codes within *Elle*.

3.2 Garnet - biotite cation exchange

Youngdo Park, Lynn Evans, Dal Park and Jin-Han Ree

3.2.1 Phenomenological observation

Cation exchange reaction between two crystalline phases refers to the process of repartitioning of the cations that constitute the two phases. In metamorphic rocks, the process generally involves (1) volume diffusion within crystalline phases that participate in the exchange reaction, (2) diffusion along the grain boundaries, and (3) chemical exchange between the reaction-participating crystalline phases and the grain boundaries that are in contact with these crystalline phases.

The garnet-biotite exchange reaction involves Fe/Mg exchange between garnet and biotite (*Fe-garnet* + *Mg-biotite* ⇔ *Mg-garnet* + *Fe-biotite*). Since the partitioning of the cations is temperature-dependent, the element distribution in garnet and biotite can be used as a geothermometer. The garnet-biotite geothermometry is widely used by metamorphic petrologists because garnet and biotite are both common in pelitic schists over a wide range of pressures and temperatures (Spear 1993 and references therein).

3.2.2 Natural example

The most common evidence for the cation exchange reaction between garnet and biotite occurs in high-grade rocks equilibrated during peak metamorphism (e.g. Ehlers et al., 1994). During cooling, garnets in high grade rocks tend to develop diffusion zoning, or zoning patterns modified by diffusion process due to the faster diffusion rates at high temperature condition. On the other hand, garnets in low-grade rocks tend to develop growth zoning during prograde metamorphism due to the changes in partitioning behaviour of Fe/Mg ions during the garnet-forming net transfer reactions.

When high-grade rocks are subjected to cooling, zonation patterns in garnet start to develop in order to reequilibrate to the new P-T conditions, while zonation in biotites develops more quickly, but is also quickly removed due to its faster diffusion rates. While for geothermometry, one normally uses only garnets and biotites that are in contact, in this section we explore the role of grain boundary diffusion in modifying zonation patterns. The degree of zonation development by diffusive modification during exchange reactions is also known to be closely related to grain size and cooling rates (Spear 1993).

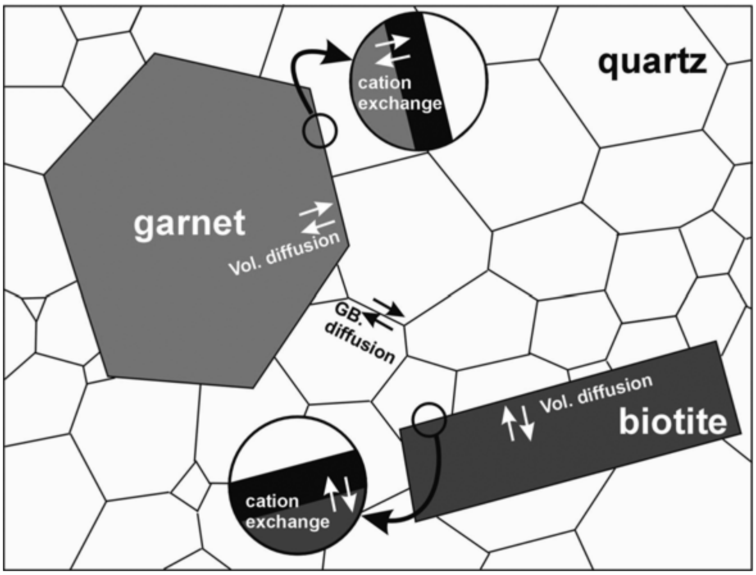


Fig. 3.2.1 Textures and phases present in the model rock. Five sub-processes for the Fe/Mg exchange reaction are also indicated (Vol.: volume, GB.: grain boundary)

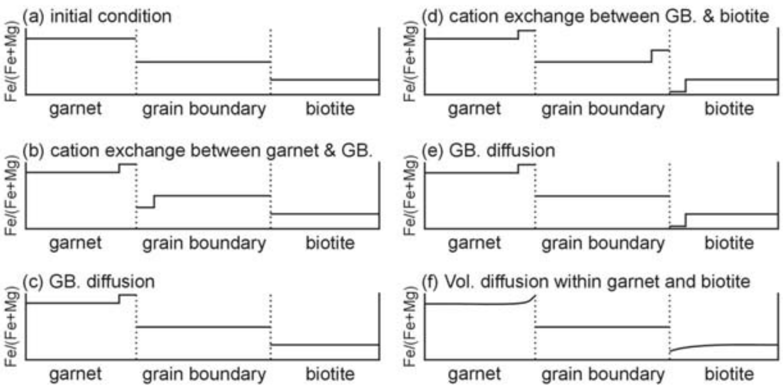


Fig. 3.2.2 Schematic one-dimensional representation of concentration changes in cations resulting from the five sub-processes for the Fe/Mg exchange reaction (Vol.: volume, GB.: grain boundary)

3.2.3 Inferred processes

Five sub-processes operate together to achieve these types of exchange reactions (Fig. 3.2.1); (1) volume diffusion within garnets, (2) cation exchange between a garnet and its grain boundaries, (3) grain-boundary diffusion, (4) cation exchange between a biotite and its grain boundaries, and (5) volume diffusion within biotites. The schematic concentration changes during the five sub-processes are shown in (Fig. 3.2.2).

3.2.4 Driving forces, controls and governing equations

The garnet-biotite exchange reaction is driven by instabilities due to disequilibrium in chemical free energy. Therefore, the fundamental approach when building the reaction model is to decide the direction of the exchange reaction (e.g. Fe loss or gain in garnet) after a change in temperature. This can be done by calculating the equilibrium state of the Gibbs free energy within the system. For this study we assume that Fe and Mg diffusion constants are equal, so that we can just consider the behaviour of one Fe as being representative. The next step is to compare the current energy state in the model with the equilibrium state and allow the reaction to progress in the direction that lowers the Gibbs free energy. Since only the rims of the garnet and biotite grains participate in the exchange reaction, only the volume of the marginal parts should be considered when calculating the energy state in the model. This is similar to the concept of "effective bulk composition" as discussed in Stüwe (1997).

Another approach to determining the direction of the exchange reaction is to compare the local value of the distribution coefficient (K_D) with a reference equilibrium value, where

$$K_D = \frac{(Mg/Fe)^{garnet}}{(Mg/Fe)^{biotite}}. \tag{3.2.1}$$

Although it is fundamentally possible to obtain the value of K_D for a given P-T condition by performing thermodynamic calculations, there are uncertainties related to thermodynamic database and thermodynamic models. For this reason, the experimentally estimated K_D is widely used for geothermometry (e.g. Ferry and Spear 1978). Ferry and Spear (1978) presented their results with the equation

$$\ln K_D = \frac{-2109}{T} + 0.782, \tag{3.2.2}$$

where T is temperature in Kelvin. In our model, we also used the experimentally estimated K_D as a reference for equilibrium when deciding the direction of the exchange reaction.

Once the direction of reaction progress is thermodynamically determined, the rate of the overall exchange reaction is controlled by the rates of five sub-processes (stated earlier). Among the five sub-processes, there are three diffusion-related processes. The calculation of concentration changes can be made with diffusion equations (described later; Park et al., 2003). The rate of the other two sub-processes related to cation exchange process between garnet/biotite crystal and grain boundary may be expressed as the relation that the rate is proportional to concentration difference as generally known in reaction kinetics (e.g. Martin and Doherty 1976). The temperature dependence of K_D and diffusivities (D) (Hoffman and Giletti 1974; Cygan and Lasaga 1985) in our model is shown in Fig. 3.2.3.

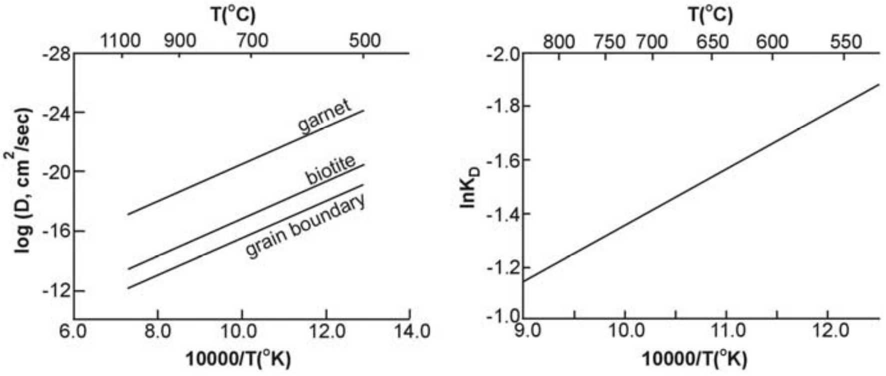


Fig. 3.2.3 Temperature dependence of diffusivities (D) and K_D in the model

3.2.5 Possible and actual simulation techniques

Finite Difference calculations have been performed to investigate the effects of various parameters (e.g. grain size and cooling histories) on the development of zonation patterns, and these calculations have produced geologically meaningful results (e.g., Spear 1991; Ehlers et al., 1994; Powell and White 1995). In these calculations, volume diffusion within a garnet crystal is usually assumed to be the rate-controlling process (i.e. volume diffusion in biotite and grain-boundary diffusion are assumed to occur at much faster rates). These models fail to show the effects of grain-boundary diffusion because of the intrinsic assumptions of the rate-controlling process. For example, the geometry of phase distribution and the distance between reactant and product phases are not considered in these models. To include the effects of geometric features, we need a model with a data structure which records grain boundaries (a 'digitized' map of a thin section image).

When performing Finite Difference calculations in a model with grain-boundary data structure, there is a problem related to computational time if the model maintains a constant node density per unit length. For example, if we want to have nodal spacing of 1 nm for 10 nm wide grain boundaries, we have to have 10^6 nodes for 1 mm grains in one dimension (1 mm divided by 1 nm). For two-dimensional calculations, 10^{12} nodes are necessary, and 10^{18} nodes are necessary in three dimensions. To overcome this problem, we adopted a hybrid data structure, which consists of (1) grain boundary nodes by which a grain-boundary segment is defined (bnodes) and (2) uniformly distributed nodes within grains (unodes). By having special nodes that allow chemical communication between a grain boundary node and a within-grain node, the number of nodes can be greatly reduced to the level at which the computation time becomes reasonable.

The summary texture of our model rock is shown in Fig. 3.2.1. Three phases are assumed to be present in our model rock; (1) garnet, (2) biotite, and (3) a non-reacting phase, such as quartz. Quartz forms polycrystalline aggregates with grain boundaries. The grain boundaries become pathways for the diffusion of Mg and Fe. We also assumed that there is no migration of grain and phase boundaries. Thus, no net transfer reaction involving phase-boundary migration occurs in our model.

For the determination of the direction of the reaction, we chose to use the experimentally determined K_D values as a reference. We compared the experimentally determined K_D with the observed K_D in the model, and then we started to change the Fe/Mg ratio in garnet toward the value of the experimentally determined K_D. Since only the grain margins are directly involved during the exchange reaction, we only considered the compositions of the rims of garnet and biotite grains when we evaluated the K_D in the model. The use of experimentally determined K_D (Ferry and Spear 1978) seems to be working for the exchange reaction, but a more general method would calculate the sum of the Gibbs free energies of the phases present in the model and compare this with the energy state at thermodynamic equilibrium. The calculation of the Gibbs free energy is more general because this method allows the prediction of the net transfer reactions.

To model the overall exchange reaction, we considered sub-processes (Fig. 3.2.1). The whole process of the exchange reaction is performed by repeating the five sub-processes in series. Calculations of flux at nodal points are made by diffusion equations (described later; Park et al., 2003). For the chemical exchange between garnet/biotite crystal and grain boundary, the finite changes in concentration are proportional to the concentration difference. There is no experimentally determined value for the proportionality constant so we chose a value based on the assumption that the rate of interfacial chemical exchange is faster than the volume diffusion within garnet and biotite.

3.2.6 Implementation in *Elle*

This model for cation exchange involves three processes: diffusion along grain boundaries, lattice diffusion within grains and cation exchange between the boundaries and grains. The *Elle* process *elle_gbdiff* is used to calculate the grain boundary diffusion, while the latter two processes were both implemented in *elle_exchange*. For diffusion calculations, we used data structures (bnodes and unodes) that hold information about positions and concentrations (Fig. 3.2.4).

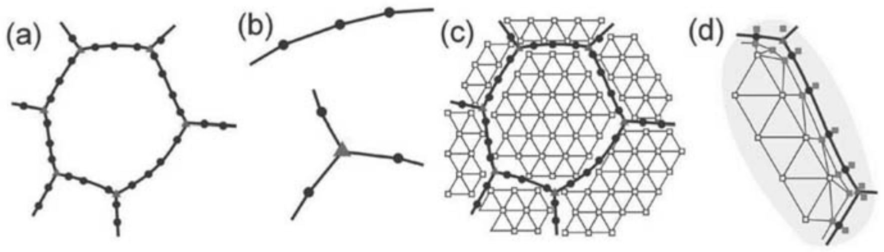


Fig. 3.2.4 Data structure of the model. **(a)** bnode structure for defining grains. **(b)** type-2 bnode (top) and type-3 bnode (bottom). **(c)** unodes within grains. **(d)** special unodes at grain margins (filled circle: type-2 bnode, filled triangle: type-3 bnode, open square: unode, filled square: special unode)

Grain boundary diffusion in *Elle* uses a Finite Difference scheme to calculate the changes in concentration at the bnodes when there is a flux along the two or three segments attached to each node. The transfer rate into a triple bnode, *O*, is the sum of the quantities from three neighboring bnodes (*A*, *B*, and *C*; Fig. 3.2.5):

$$q_{OA} = -AD\frac{C_O - C_A}{X_A} \cdot dt, \tag{3.2.3}$$

$$q_{OB} = -AD\frac{C_O - C_B}{X_B} \cdot dt, \tag{3.2.4}$$

$$q_{OC} = -AD\frac{C_O - C_C}{X_C} \cdot dt, \tag{3.2.5}$$

where *q*, *A*, *D*, *C* and *X* respectively represent moles transferred, area of grain boundary section, diffusivity, concentration and length of a grain boundary segment. Note that the area of grain boundary section (A) is equal to the quantity, grain-boundary width (W_{GB}) multiplied by the thickness (t_{SAMPLE}) of the two dimensional *Elle* data structure, or $W_{GB} \cdot t_{SAMPLE}$. The total moles transferred (Q) is therefore given by

$$Q = q_{OA} + q_{OB} + q_{OC}$$

$$= -(W_{GB} t_{SAMPLE}) D \left[\frac{C_O - C_A}{X_A} + \frac{C_O - C_B}{X_B} + \frac{C_O - C_C}{X_C} \right] \cdot dt. \tag{3.2.6}$$

Then, the total molar transfer rate becomes

$$\frac{dQ}{dt} = -(W_{GB} t_{SAMPLE}) D \left[\frac{C_O - C_A}{X_A} + \frac{C_O - C_B}{X_B} + \frac{C_O - C_C}{X_C} \right]. \tag{3.2.7}$$

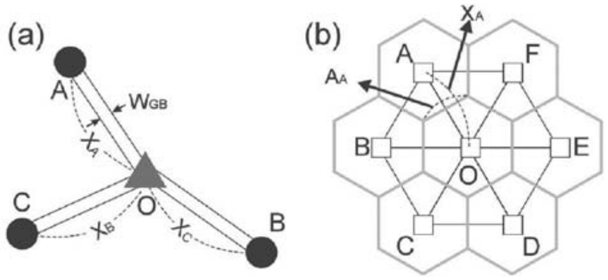


Fig. 3.2.5 Diffusion calculations **(a)** at grain boundaries, **(b)** within grain. See text for detail

The rate of change in concentration is the molar transfer rate divided by the volume occupied by the node *O*. The volume occupied by the node *O* can be approximated to be half the sum of the segments between the node *O* and its neighbours (*A*, *B*, *C*). Thus, the volume becomes

$$t_{SAMPLE} \left[\frac{1}{2}(X_A W_A) + \frac{1}{2}(X_B W_B) + \frac{1}{2}(X_C W_C) \right]., \tag{3.2.8}$$

and the change in concentration with time at each bnode is given by the equation

$$\frac{dC}{dt} = t_{SAMPLE}^{-1}\left[\frac{1}{2}(X_A W_A) + \frac{1}{2}(X_B W_B) + \frac{1}{2}(X_C W_C)\right]^{-1} \frac{dQ}{dt}. \qquad (3.2.9)$$

Volume Diffusion in *elle_exchange* uses the unode data structures to record the chemical concentrations within a crystal. Most of the unodes in a grain are evenly distributed with a two dimensional hexagonal close-packed structure. Special nodes are also made within a grain near the boundary nodes for chemical communication between grain and grain boundary. The evenly distributed unodes and special unodes are triangulated for the Finite Difference diffusion calculation of the flux between the regions represented by each unode. For this implementation, the unode area is constructed by connecting the center of gravity points of neighboring triangles (Fig. 3.2.5b). The flux into unode *O* from the neighboring regions (*A*, *B*, *C*, *D*, *E* and *F*; Fig. 3.2.5b) can be expressed with the series of equations

$$q_{OA} = -A_A D \frac{C_O - C_A}{X_A} \cdot dt, \ldots \quad q_{OF} = -A_F D \frac{C_O - C_F}{X_F} \cdot dt, \qquad (3.2.10)$$

where q, A, D, C and X respectively represent moles transferred, area of flux section, diffusivity, concentration and distance between two unodes (Fig. 3.2.5b). In a similar manner to grain boundary diffusion, Q and dQ/dt can be estimated using the material transferred from neighboring unodes. Then, $\partial Q/\partial$ can be estimated by dividing $\partial Q/\partial$ by the volume occupied by unode O.

Chemical exchange between a grain boundary and a grain is modelled in *elle_exchange* using the bnodes and coincident special unodes. The rate of exchange is assumed to be proportional to the concentration difference and for both biotite and garnet we used an arbitrary value (1.0) for the constant, which allowed sufficiently fast reactions within a run time.

3.2.7 Example *Elle* Run: garnet-biotite cation exchange

We present the results from two types of experiments; (1) isothermal experiments with different matrix grain sizes, and (2) a cooling experiment.

For the isothermal experiments, we have defined a model size of 1cm × 1cm and varied the grain size in the phase that does not participate in the exchange reaction, the quartz matrix. Two different matrix grain sizes (square grains with sides of 0.25 cm and 0.05 cm) were chosen to see the effect of matrix grain size on the rate of the exchange reaction. The initial compositions for garnet and biotite are assumed to be 0.8 and 0.6 (in Fe mole fraction), respectively. Since these compositions are disequilibrium compositions at the experimental temperature condition (600°C), garnet and biotite start to exchange cations to approach the equilibrium K_D value. The results (Fig. 3.2.6 and 3.2.7) show that the rate of the ion exchange reaction is enhanced as the grain size in the matrix phase decreases. Since the density of grain boundary area per unit volume increases as grain size decreases, the amount of ion transportation by diffusion will be greater for the sample with smaller matrix grain size.

For cooling experiments, zonation development is simulated while reducing the temperature from 750 °C to 450 °C at a rate equivalent to 10 °C per Ma. We defined the starting compositions of garnet as 0.8 and biotite as 0.54 in Fe mole fraction. Much of the changes in zonation pattern in garnet occur within the first 15 Ma period due to high temperature conditions at the earlier stages (Fig. 3.2.8a-b). Sluggish changes in zonation pattern occur during the late stages of the experiment due to slower diffusion rates after cooling (Fig. 3.2.8b-c). Under these conditions the contact between one of the biotites and the garnet has had little impact on the overall garnet zonation pattern, as grain boundary diffusion is quite rapid.

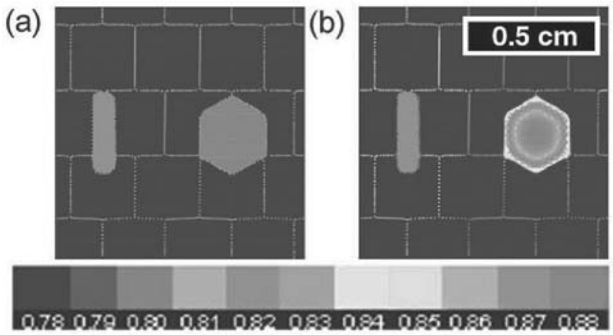


Fig. 3.2.6 Results from an isothermal experiment with matrix grain size of 0.25cm. **(a)** Initial condition and **(b)** after 50 Ma. The biotite grain is on the left, the garnet grain on the right. The CLUT (colour lookup table) of $C_{(Fe)}$ is optimized to highlight the garnet composition. EXPERIMENT 5

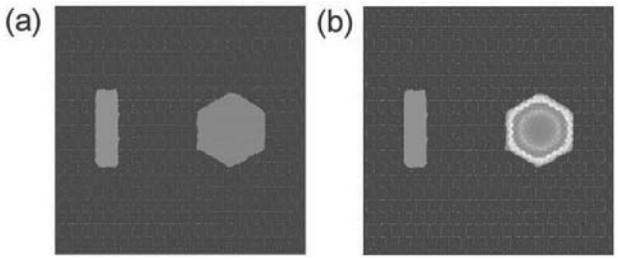


Fig. 3.2.7 Results from an isothermal experiment with matrix grain size of 0.05cm. **(a)** Initial condition and **(b)** after 50 Ma. The biotite grain is on the left, the garnet grain on the right. Same CLUT and scale as Fig. 3.2.6. EXPERIMENT 5

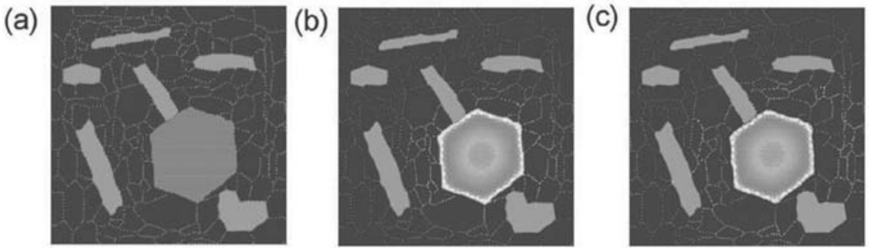


Fig. 3.2.8 Results from a cooling experiment. **(a)** Initial condition, **(b)** after 15 Ma and **(c)** after 25 Ma. The garnet grain is the large hexagonal grain near the middle of the model. Same CLUT and scale as Fig. 3.2.6. EXPERIMENT 5

Symbols used in Chap. 3.2

A	Cross-sectional area of grain boundary [m^2]
C	Concentration [mol m^{-3}]
D	Diffusion coefficient [m^2 s^{-1}]
K	Distribution coefficient []
Q	Total material transferred into a bnode or unode [mol s^{-1}]
q	Material transferred through one boundary segment [mol s^{-1}]
t_{SAMPLE}	Thickness of sample in 2D *Elle* data structure [m]
W_{GB}	Width of grain boundary [m]
X	Length of a grain boundary segment or distance between adjacent Unodes [m]

3.3 Subgrain growth Potts model

Sandra Piazolo and Mark W. Jessell

3.3.1 Phenomenological observations

In rocks deforming by crystal plastic mechanisms, a substructure of subgrain walls and linear defects (dislocations) develops (for formation of subgrains see Chap. 3.4). Subgrains are regions within a grain that have misorientation angles of less than 10-15° of their lattice orientation with those of the neighbouring subgrains (the misorientation angle is the angular mismatch of lattices on either side of the boundaries). This substructure is essentially made of high stacking-fault energy dislocation arrays and subgrains of various boundary types. During thermal activation such a substructure evolves with time. The evolution of subgrains consists of two processes: (a) progressive decrease in crystal lattice distortions within subgrains and (b) progressive increase of the average subgrain area and with that decrease of subgrain boundary length. Here, we are predominately concerned with process (b) the increase of average subgrain area.

Substructures are useful as indicators of deformation mechanisms (e.g. Urai et al., 1986, Trimby et al., 1998) and palaeo-stress levels in the lithosphere. In addition, the potential of subgrain misorientations as useful indicators of finite strain has recently been put forward by Pennock et al. (2005). Moreover, dislocation substructures play an important control on the mechanical properties of materials.

3.3.2 Natural examples and experiments

It is not easy to demonstrate the activity of subgrain growth in resultant microstructures in rocks because in most cases the characteristics of the initial substructure are unknown. Therefore it is largely by inference that we know that substructure evolution has taken place in natural samples, assuming that the system aims to attain a reduction in its energy state. This will be achieved if the overall length of subgrain boundaries is decreased, which in turn provides the thermodynamic driving force for subgrain growth.

In contrast, growth of subgrains can be and has been observed in experiments in which pre-deformed polycrystals are statically heated and the substructure is monitored before and after heating (Figs. 3.3.1 and 3.3.2).

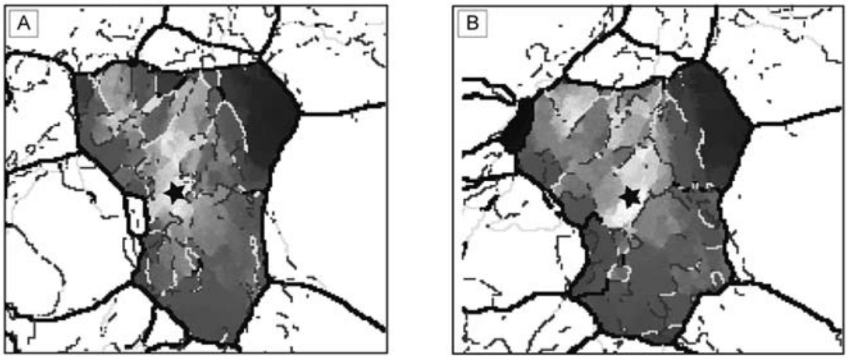


Fig. 3.3.1 Subgrain growth as observed during static heating of an experimentally deformed salt polycrystal. Figure illustrates two EBSD maps taken at a time interval of 1.5 hrs at 430 °C. **(a)** Before heating. **(b)** After heating. In each map the same area is shown; and the difference in misorientation in the central grain from one chosen crystallographic orientation (marked with a star) is given from white (0°) to black (25°); thick black, white, thin grey lines signify grain boundaries at >10°, 5-10°, 3-5° misorientation, respectively. Note the overall decrease in the number of subgrains present as well as the decrease in lattice distortion within individual subgrains. Note also that the high angle grain boundaries (black) move. Field of view is 400 μm

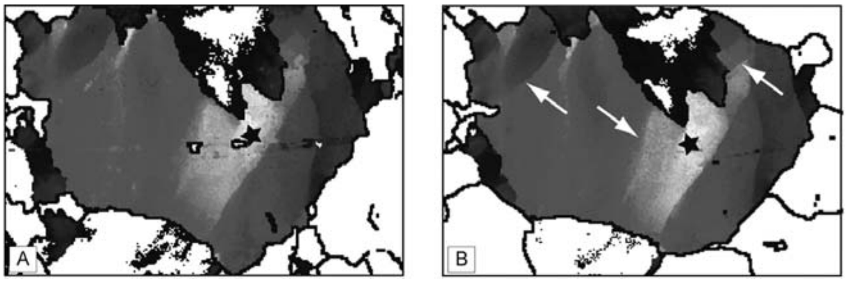


Fig. 3.3.2 Subgrain growth as observed during static heating of an experimentally deformed Mg-alloy polycrystal. Figure illustrates two EBSD maps taken at a time interval of 0.5 hrs at 330 °C. **(a)** Before heating. **(b)** After heating. In each map the same area is shown; in each the difference in misorientation from one chosen crystallographic orientation (marked with a star) is given from white (0°) to black (15°); black lines signify grain boundaries > 10° misorientation. Field of view is 1200 μm. White arrows point to areas where the subgrain boundary sharpened and/or moved during heating

3.3.3 Inferred processes

The stored energy of a subgrain-structured material is large and is lowered by the process of subgrain growth during heat treatment. This subgrain growth leads to a reduction in the total area of low angle boundaries in the material. In addition, during thermal activation annihilation of dislocations of different signs takes place within subgrains, which decreases the total stored energy of the system and decreases the local crystal lattice distortion.

When considering the process of subgrain growth, i.e. the increase in average subgrain area and decrease in total subgrain boundary length, the movement of subgrain boundaries is of major importance. This movement may take place by the diffusion of single or possibly groups of atoms from one subgrain to an adjacent grain, by the rotation and shuffling of atoms from one lattice to the other, or perhaps by piecemeal movement of clusters of atoms (Gottstein and Shvindlerman 1999). However at the grain scale these processes are indistinguishable. The motion of one or groups of atoms from one subgrain to another results in the migration of the boundary in the opposite direction to the atom or cluster motion.

3.3.4 Driving forces, controls and governing equations

The driving force for general boundary migration in rocks may be a product of deformation induced dislocation density contrasts, the result of chemical potential gradients, or the inherent surface energy of the boundaries themselves (Urai et al., 1986), the bulk elastic distortion of the lattice (Kamb 1959), and even an imposed magnetic field (Humphreys and Hatherley 1995). The driving force for migration is the energy reduction resulting from its motion, and will in fact in rocks always be the result of more than one driving force, since all crystals contain defects, and all grain boundaries have an inherent boundary energy. The rate at which a boundary migrates is both a function of this driving force, and the mobility of the boundary, which is an inherent property of a material (but one that varies according to temperature, the presence or absence of fluids, the nature of the boundary, and the impurity content of both the subgrains and the boundary). Note that the general driving forces and controls for subgrain boundaries are very similar to those of grain boundaries; however the driving forces are typically smaller and the relative importance of surface energies increases as the grain size reduces. At present the actual mobilities for mineral subgrain boundaries are not well known at all. In all cases the prediction of the velocity of the boundary *(v)* can be considered in terms of a fundamental material property: the grain boundary mobility *m* (in units of m/Js); and a driving force *(F)* acting on an area of grain boundary *A* (m^2)

$$v = \frac{mF}{A} \tag{3.3.1}$$

3.3.5 Possible and actual simulation techniques

Subgrain growth has been modelled using two main modelling techniques:

1. Discrete schemes mapped onto regular grids, where local changes in subgrain boundary position occur as a result of individual grid points switching their orientation, based on energy changes associated with the local distribution of orientations. Here, the most common model types include Potts (e.g. Holm et al., 2004) and Cellular Automata Models (e.g. Raabe 2002).
2. Continuous schemes mapped onto regular grids, where the local change in orientation property is a function of the properties of the whole system; here, the most common modelling type is the Phase Field Model (Radhakrishnan et al., 2001).

3.3.6 Implementation in *Elle*

In *Elle* the growth of subgrains is modelled using a kinetic Monte Carlo model. The program steps through all the unodes of one grain and runs through 6 main algorithm steps (Mehnert and Klimanek 1996). These are:

1. Randomly pick a crystallographic orientation S_i of one of the 6 neighbours for grid site *i*.
2. Calculate the energy E_i according to Eq. 3.3.2 (see below).
3. Pick a new orientation S_j from one of the neighbours.
4. Calculate the energy of the site with orientation S_j.
5. Flip the grid site *i* to orientation S_j with a probability *P* according to Eq. 3.3.3 (see below).
6. Repeat steps 1–5.

The equation to determine the energy, *E*, is

$$E_i = -J\sum_{j=1}^{nn}(\delta - 1), \tag{3.3.2}$$

where J is the energy factor, nn the number of neighbours and δ is the Kronecker delta. The number of neighbours is 6 if we are using a hexagonal unconnected grid. The probability *(P)* is defined as

$$\begin{aligned} P &= \exp\left(\frac{-\Delta E}{kT}\right) && \text{for } \Delta E > 0 \text{ and} \\ P &= 1 && \text{for } \Delta E \leq 0, \end{aligned} \tag{3.3.3}$$

where $\Delta E = E_{new} - E_{old}$, hence the energy difference due to the change in crystallographic orientation, k the Boltzmann constant and T the system temperature. For simplicity, in our simulations, $J = 1$ and $kT = 1$.

At present two subgrain growth models are used:

1. *Algorithm Subgrain growth I – isotropic energy.* The algorithm as described above is used for this model; the calculation of the energy does not consider subtle differences in misorientation. If the misorientation between the spin and the neighbouring site is the same the energy is 0, if it is above the critical misorientation ϕ_m is 1.
2. *Algorithm Subgrain growth II – anisotropic energy.* Here, the algorithm as described is modified where the energy differs with misorientation ϕ between neighbouring grid points following the linear relationship of the following form

$$E = E\frac{\phi}{\phi_m}. \tag{3.3.4}$$

This equation is only valid for misorientations below a critical value of misorientation ϕ_m at which and above which the boundary has its maximum surface energy.

3.3.7 Example *Elle* Runs

Subgrain growth – isotropic energy function: In the first example (Fig. 3.3.3), a simple subgrain growth experiment has been performed by taking an input file with a highly strained grain of NaCl derived from EBSD measurements, which exhibit high grain lattice distortions. We let the substructure of the grains evolve while the grain boundaries remain stable. The driving force is the reduction of energy where the energy below the critical misorientation (between adjacent subgrains) ϕ_m of 15° is isotropic.

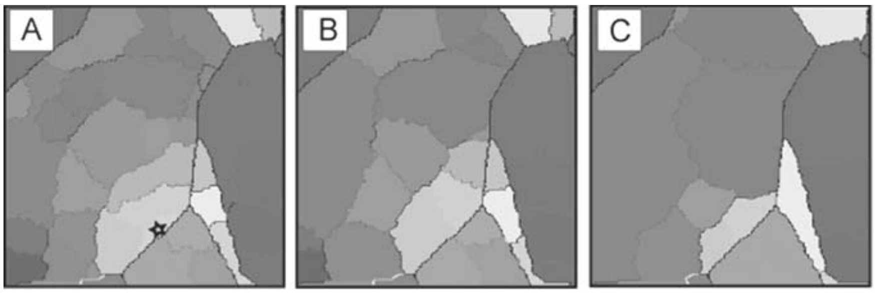


Fig. 3.3.3 Example run showing isotropic surface energy driven subgrain growth. Greyscale scheme shows relative misorientation from one crystallographic orientation (marked as black star) in greyscale; black lines signify grain boundaries with >15° misorientation; **(a-c)** results at model time steps 1500, 2500, 6000 (modified after Piazolo et al., 2004). EXPERIMENT 6

Subgrain growth – anisotropic energy function: In this simulation (Fig. 3.3.4) we take into account the anisotropy of surface energy and mobility of subgrain boundaries. The input microstructure is the same as for section 3.3.4, however the calculation of the energy state differs as now the energy between data points below the critical misorientation ϕ_m of 10° is taken to be anisotropic (see above for details). It can be seen that in this case more subgrains remain at the end of the simulation, as the anisotropy has the effect of slowing down the microstructure evolution.

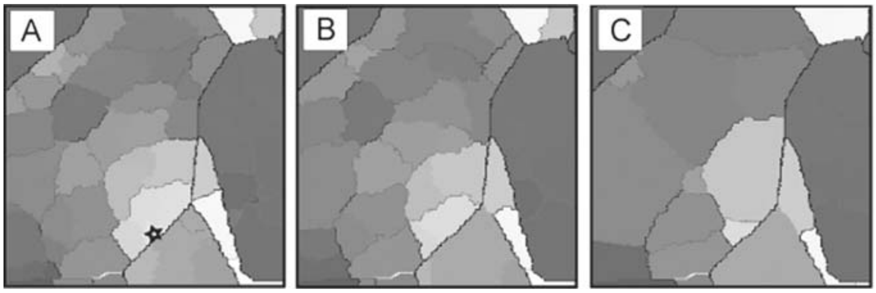


Fig. 3.3.4 Example run showing anisotropic surface energy driven subgrain growth. Greyscale scheme shows relative misorientation from one crystallographic orientation (marked as black star) in greyscale; black lines signify grain boundaries with >10° misorientation; **(a-c)** results at model time steps 1500, 2500, 6000 (modified after Piazolo et al., 2004). EXPERIMENT 6

Symbols used in Chap. 3.3

A	Area of grain boundary [m^2]
E	Energy of a lattice site [J]
F	Driving force [N]
ϕ	Lattice misorientation between neighbouring lattice sites [°]
ϕ_m	Critical lattice misorientation between neighbouring lattice sites [°]
J	Energy factor [J]
k	Boltzmann's constant = $1.3806503 \cdot 10^{-23}$ [J K^{-1}]
m	Grain boundary mobility [m^3 s^{-1} N^{-1}]
nn	Number of neighbours of a lattice site
P	Probability
v	Velocity of a boundary [m s^{-1}]
S	Crystallographic orientation of a lattice site [°]

3.4 Nucleation and subgrain formation

Sandra Piazolo and Mark W. Jessell

3.4.1 Phenomenological observations

Nucleation refers to the appearance of new grains within a polycrystalline mono- or polymineralic material. Two main types of nucleation need to be distinguished. There is (a) the nucleation of a strain-free grain within a matrix of deformed, highly strained grains and (b) the nucleation of a new mineral phase in a polymineralic rock during changing pressure, temperature and fluid conditions. Nucleation of type (a) is commonly referred to as dynamic recrystallisation in metallurgy (e.g. Gottstein and Shvindlerman 1999). Here, we will only consider case (a) i.e. nucleation in a strained monomineralic material. Experimentally the evidence for such strain-free grains is widespread (Fig. 3.4.1) (e.g. Humphreys and Hatherley 1995).

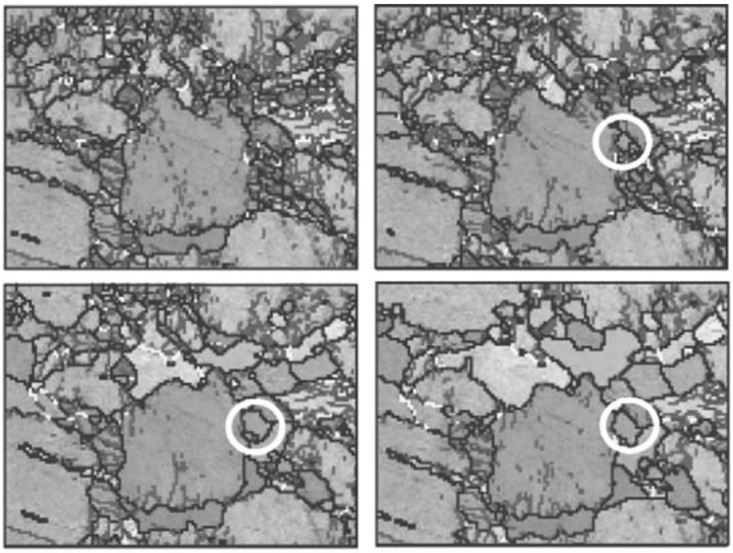


Fig. 3.4.1 Time series during static heating of an experimentally pre-deformed Mg-alloy. Time intervals 10 min from top left to bottom right. Different greyscales represent different crystallographic orientations. The grain encircled in white depicts a new, strain free grain that apparently nucleated during heating

The presence of subgrains, i.e. areas which are bound by boundaries with less than 10°-15° misorientation, is commonly observed in deformed crystalline materials (Fig. 3.4.2a), with increasing strain the density of subgrain boundaries is often observed to increase (e.g. Trimby et al., 1998). In geology, such subgrains are commonly recognised with a transmitted-light microscope by their undulose extinction (Fig. 3.4.2b).

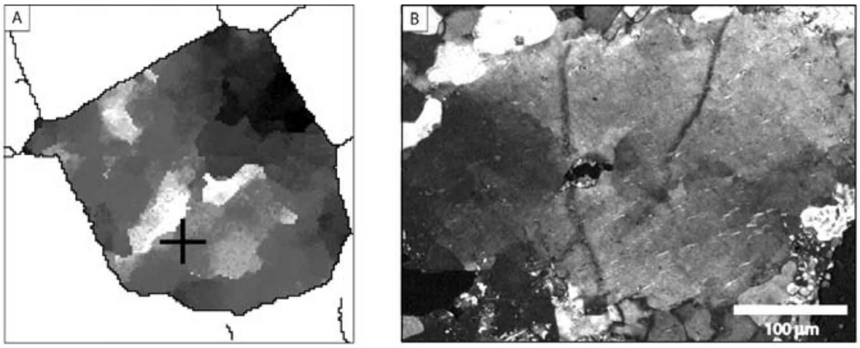


Fig. 3.4.2 Subgrain structure seen in experimentally deformed salt. **(a)** Figure illustrates the difference in misorientation from one chosen crystallographic orientation (marked with a cross) from white (0°) to black (15°). Field of view is 250 m, black lines signify grain boundaries at >10° misorientation. **(b)** Micrograph showing typical undulose extinction signifying the presence of subgrains. Sample from a feldspar in a deformed meta-psammite from Central Dronning Maud Land, Antarctica; crossed nicols

3.4.2 Natural examples

In many natural examples e.g. albite, quartz and calcite it has been suggested that subgrains show distinct discontinuous stages during and after deformation where the dislocation structure in a subgrain is cleared out producing relatively strain- and dislocation–free grains (Figs. 3.4.3) (e.g. Urai et al., 1986; ter Heege et al., 2002).

The formation of subgrains is commonly observed during solid state deformation of crystalline materials (Fig. 3.4.4). The orientation of subgrain boundaries is often crystallographically controlled. Subgrain boundaries in quartz are commonly parallel or perpendicular to the crystallographic c-axis of the original grain (Lloyd et al., 1992; Stöckert et al., 1999). Another well-known example are subgrain boundaries in olivine which develop sub-parallel to (100) (Lallemant 1985).

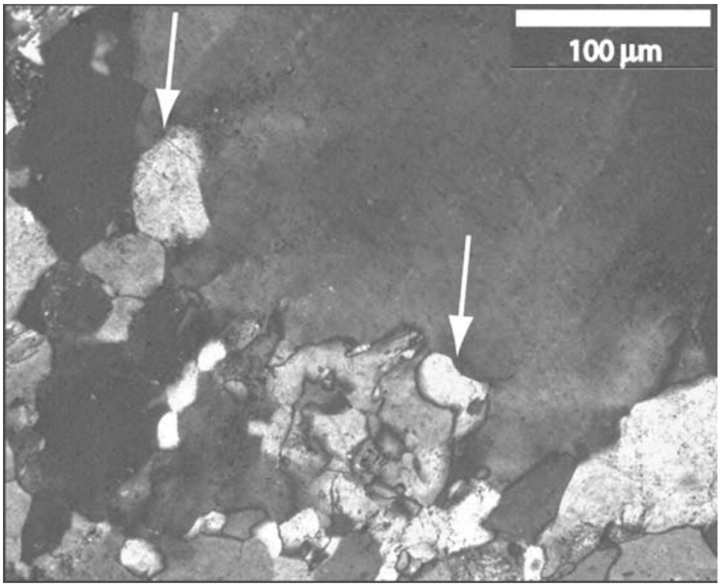


Fig. 3.4.3 Micrograph of showing 2 relatively defect-free small grains (arrows) within a grain with significant undulose extinction. These feldspar grains are interpreted as having nucleated during deformation. Sample from a deformed meta-psammite from Central Dronning Maud Land, Antarctica; crossed nicols

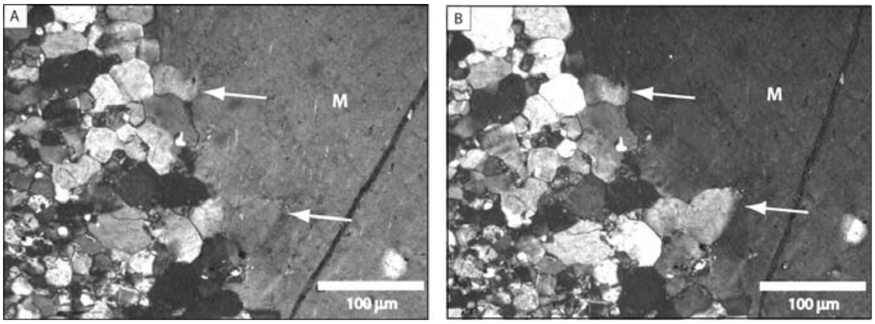


Fig. 3.4.4 Micrographs showing an example of a microstructure commonly interpreted to originate from progressive subgrain formation. **(a)** and **(b)** are taken under different rotations with respect to the cross-polarized light. Note that the two subgrains still show some internal deformation (undulose extinction) and are at the same time distinguished by a slightly different orientation from the mother grain (marked as “M”). Sample from a deformed meta-psammite from Central Dronning Maud Land, Antarctica

3.4.3 Inferred processes

There are two main models that have been put forward to explain the appearance of new, substructure-free grains:

1. Nucleation in the classical sense (Burke and Turnbull 1952); in this situation, nucleation is accomplished by random atomic fluctuations leading to the formation of a small crystallite with surrounding mobile grain boundaries.
2. Growth from precursor cells. Cells with considerably lower dislocation density than their neighbouring cells may already pre-exist or may form by recovery. At elevated temperature some of these cells may form mobile boundaries and grow rapidly to form larger, essentially substructure-free grains (Doherty et al., 1997) (Fig. 3.4.5).

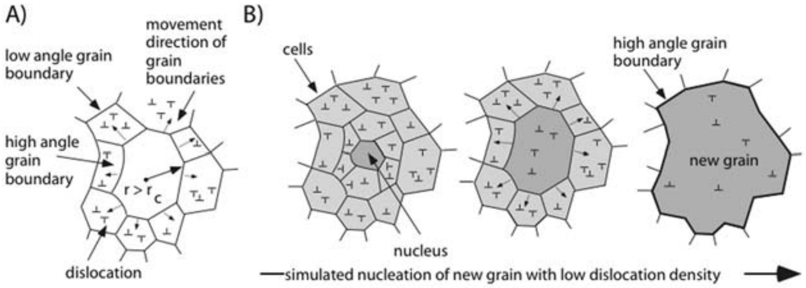

A)
movement direction of grain boundaries
low angle grain boundary
high angle grain boundary
r > r_c
dislocation
B)
cells
high angle grain boundary
new grain
nucleus
simulated nucleation of new grain with low dislocation density

Fig. 3.4.5 Schematic illustration of the mechanism of recrystallisation by nucleation from precursor cells (Humphreys and Hatherly 1995; Doherty et al., 1997; Gottstein and Shvindlermann 1999). **(a)** Recrystallisation nucleus with growth potential in a deformed aggregate (modified after Gottstein and Svindlermann 1999). **(b)** Schematic illustration how the process recrystallisation by nucleation is modelled. A grain with a high strain energy (i.e. high dislocation density) is envisaged as a group of small cells. The higher the strain energy the higher the probability that one of these cells has a significantly lower strain energy than its neighbours. Such a cell then acts as a nucleus whose boundaries may rapidly sweep through its neighbours. Eventually a new grain develops with low internal energy and high angle grain boundaries

During deformation, dislocations are generated, move and may be stored in a crystal. Dislocations do not occur in thermodynamic equilibrium and therefore these linear lattice defects also have a driving force that allows them to move to lower the energy of the system. One way to minimize the free energy is a rearrangement of dislocations in low energy dislocation structures such as planes of dislocations forming regular arrays, resulting in subgrain boundaries (also called low-angle grain boundaries). The character of these arrays depends on the types of dislocation involved (Humphreys and Hatherley 1995). A schematic diagram of the formation of a subgrain boundary is shown in Fig. 3.4.6.

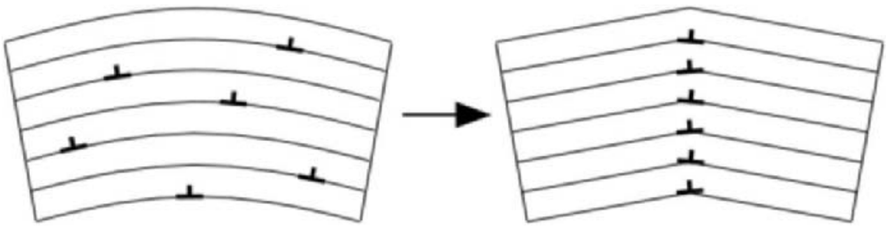


Fig. 3.4.6 Schematic diagram illustrating the formation of subgrain boundaries by arrangement of dislocations into regular arrays / low angle boundaries (modified after Passchier and Trouw 2005)

3.4.4 Driving forces, controls and governing equations

For both models cited above, the nucleus must fulfil two main characteristics: (a) its size must be above a critical radius so that there is a net driving force to grow rather than to contract, and (b) its grain boundaries must be mobile to allow observable growth of the nucleus.

The critical radius R_{cr} at which there is a positive driving force for growth depends on the stored energy difference between the nucleus and the surrounding grains, minus the driving force to shrink the nucleus by grain boundary area reduction.

Neglecting dislocation spacing and core energy effects, the driving stress (σ_{stored}) for grain boundary migration driven by differences between the stored energy inside and outside the nucleus is given by:

$$\sigma_{stored} = E_\rho \Delta\rho . \tag{3.4.1}$$

Here E_ρ is the energy per unit length of dislocation and $\Delta\rho$ is the difference in free dislocation density between nucleus and surrounding grains (e.g. Humphreys and Hatherley 1995; Gottstein and Svindlermann 1999). For a spherical grain nucleus, the driving stress ($\sigma_{surface}$) for grain boundary area reduction due to the grain boundary curvature effect is

$$\sigma_{surface} = \frac{-2\gamma}{R}, \tag{3.4.2}$$

where γ is the grain boundary energy and R the radius of the grain, cell or nucleus. Therefore, the resultant critical radius R_{cr}, below which a nucleus will shrink and above which it will grow, is

$$\sigma_{stored} + \sigma_{surface} = 0 \Leftrightarrow R_{cr} = \frac{2\gamma}{E_\rho \Delta\rho} \tag{3.4.3}$$

For typical metals at reasonable deformations, the critical nucleus size for recrystallisation is on the µm scale, so atomic fluctuations are insufficient to provide critical nuclei in any realistic timeframe. Therefore, a number of theories propose that nuclei arise from the recovered subgrain structure, in particular from abnormal growth of certain subgrains as stated above. In a more general sense though, the possibility that a strain-free nucleus satisfies the prerequisites (as stated in section 3.3.3) increases with increasing dislocation density (Kocks 1985).

An increase in the internal stresses, that is increase in the number of crystal point and line defects within a cell, provide the local driving force for rearrangement of dislocations into dislocation arrays/subgrain walls (Kocks 1985). These internal stresses are proportional to the dislocation density within the cell. Accordingly, a cell with a high dislocation density has a larger driving force to develop subgrain boundaries than a cell with low dislocation density.

3.4.5 Possible and actual simulation techniques

Nucleation has been modelled using the two main groups of techniques, which are:

- Discrete schemes mapped onto regular grids where a local change in orientation occurs as a result of individual grid points switching their orientation, as a function of the local distribution of orientations. These include Potts and Cellular Automata Models (see references in Doherty et al., 1997).
- Continuous schemes mapped onto regular grids; here the local change in orientation is a function of the properties of the whole system. One of the main modelling techniques is the Phase Field Model, where a stochastic law represents nucleation, where the nucleation rates are matched to the rate of formation of nuclei in the material (e.g. Simmons et al., 2000).

Subgrain formation has been modelled using (a) a Monte Carlo, Potts or Cellular Automata model, and (b) a Front-Tracking method (Piazolo et al., 2002), where new subgrain boundaries are added to the network of grain boundaries. The probabilistic law for this addition of subgrain boundaries is assumed to be a function of the dislocation density or local misorientation gradients. The positioning of the boundary may be chosen to be a function of crystallography, the local distribution of dislocations or local misorientation gradients.

3.4.6 Implementation in *Elle*

elle_angle_rx

The discontinuous nucleation and growth of relatively strain- and dislocation-free grains, which develop from highly strained subgrains is based on two apparently distinct steps: (1) initial formation of the new grain and (2) its growth. During process (1) the dislocation structure in an “old” subgrain is cleared out and the new grain exhibits a low dislocation density (Fig. 3.4.3b).

In simulations the critical threshold value τ_{nucl} for the probability of nucleation of a certain mineral species at a specific temperature is calculated from a user-defined base nucleation threshold value τ_0 specific to a mineral species

$$\tau_{nucl} = \frac{1000\tau_0}{T + 1000} \tag{3.4.4}$$

where T is the absolute temperature. If the stored energy derived from the dislocation density of a subgrain $E_{stored} > \tau_{nucl}$ there is a probability that a nucleus, which fulfils the requirements for recrystallisation by nucleation, is present in a polygon. The dislocation density of the newly nucleated grain is set to a specified low dislocation density value (e.g. 10^{10} m^{-2}) and the crystallographic orientation of the new grain is randomly picked. This randomly picked orientation must satisfy the condition that the boundaries at all sides of the grain of the recrystallised grains are high-angle and hence mobile boundaries. This prerequisite satisfies the fact that a new nucleus will only be able to grow if at least some of its boundaries are high angle boundaries (Fig. 3.4.5). In simulations all subgrain boundaries must be high angle boundaries to allow the “successful” nucleation of the grain, as the position of the subgrain boundaries does not change due to computational limitations.

elle_split

A discontinuous development of subgrain boundaries is modelled in *Elle* as the repeated "splitting" of a "parent" polygon (grain) into two smaller "daughter" polygons i.e. subgrains. The driving force for subgrain formation is the total strain energy per unit area, which depends on the dislocation density of the polygon. The driving force E_{stored} for subgrain formation is calculated according to

$$E_{stored} = \rho E_{\rho} \rho_{scale} \tag{3.4.5}$$

where ρ is the dislocation density, E_{ρ} the energy per dislocation, and ρ_{scale} a dislocation density scaling factor. If $E_{stored} > \tau_{nucl}$ (the energy threshold value for splitting of a polygon), the grain has a possibility of "splitting" into subgrains. The probability for "splitting" increases with increasing E_{stored}. The new subgrain boundaries can be chosen to be preferentially parallel or perpendicular to the crystallographic c-axis of the "parent" grain when quartz is modelled. No orientation change is imposed as a result of the initial formation of a subgrain.

3.4.7 Example *Elle* Run

In this example, we show the effect of the nucleation process in *Elle* (Fig. 3.4.7). The initial microstructure is a highly deformed polycrystal with locally high dislocation densities present in subgrains. During the nucleation process, some of these subgrains with high dislocation density evolve into new grains, characterized by low dislocation density and high-angle boundaries towards their surrounding subgrains.

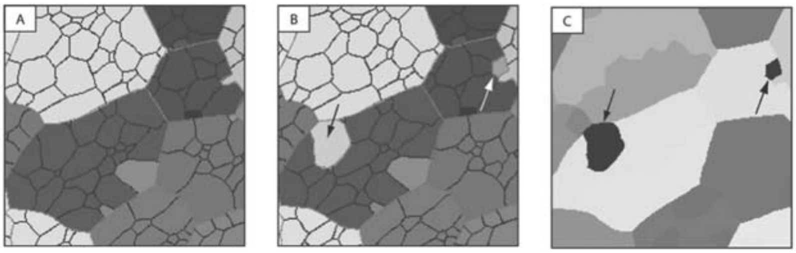


Fig. 3.4.7 Example of the nucleation process as modelled in *Elle*. **(a)** Before nucleation, different greyscales depict different crystallographic orientations; dark lines are low angle boundaries, light grey are high angle boundaries; **(b)** After nucleation of two new nuclei (shown with arrows) – shown in "new" different greyscale to their neighbours – hence exhibit a different crystallographic orientation. **(c)** Same stage as in (b) but this time dislocation density is shown (white is high, black is low density)

In the second example, we show the subgrain formation process in *Elle* (Fig. 3.4.8). The initial microstructure is a highly deformed polycrystal with locally high dislocation densities present in subgrains and grains. During the subgrain formation process, one grain is "split" into several subgrains, which are characterized by low-angle boundaries to their subgrain neighbours and a boundary orientation that is parallel or perpendicular to the c-axis of the initial grain.

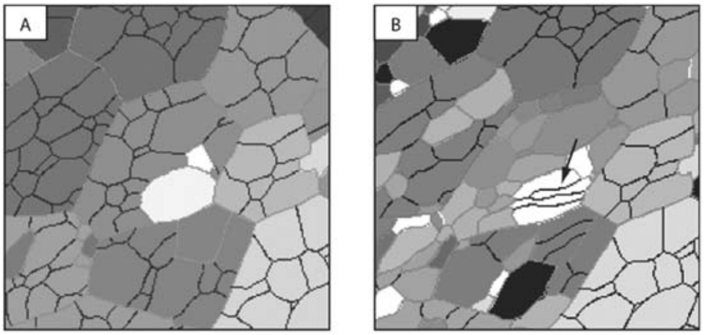


Fig. 3.4.8 Diagram illustrating the subgrain formation process in a polycrystalline quartzite as modelled in *Elle*. **(a)** Before subgrain formation; different greyscales depict different crystallographic orientation; dark lines are low angle boundaries, light grey lines are high angle boundaries. **(b)** State after subgrain formation; arrow points to area where new grain boundaries were formed; note that these are all roughly parallel since they are modelled as being crystallographically controlled

Symbols used in Chap. 3.4

E_ρ	Energy per dislocation [J m^{-1}]
E_{stored}	Stored energy (density) due to free dislocations in a nucleus relative to its neighbours [J m^{-3}]
γ	Grain boundary energy [J m^{-2}]
R	Radius of a grain [m]
R_{cr}	Critical radius for a nucleating grain [m]
$\Delta\rho$	Difference in free dislocation density between nucleus and surrounding grains [m m^{-3}]
ρ	Dislocation density [m m^{-3}]
ρ_{scale}	Dislocation density scaling factor
$\sigma_{surface}$	Driving force for grain boundary area reduction [Pa]
σ_{stored}	driving stress for grain boundary migration due to differences in stored energy [Pa]
τ_0	Base threshold for the probability of nucleation of a grain [J m^{-3} K]
τ_{nucl}	Temperature-dependent threshold for the probability of nucleation of a grain [J m^{-3}]

3.5 Grain boundary migration

Paul D. Bons, Mark W. Jessell and Jens Becker

3.5.1 Phenomenological observations

Grain boundary migration refers to the movement of the boundary separating two grains, relative to the material that makes up those grains. This definition therefore excludes boundary motion inherited from deformation of the entire system. Although recrystallisation had been identified by the beginning of the 20th century, recrystallisation and grain growth had not clearly been distinguished as separate processes. The work of Carpenter and Elam (1920) and Altherthum (1922) established that stored energy provided the driving force for recrystallisation and grain boundary energy that for grain growth by grain boundary migration. This is shown by the (German) terminology for these processes used by Altherthum - *Bearbeitungsrekristallisation* ([cold-] work recrystallisation) and *Oberflächenrekristallisation* (surface [tension] recrystallisation).

In 1898 Stead had proposed that grain growth occurred by grain rotation and coalescence, and although Ewing and Rosenhain (1900) presented convincing evidence that the mechanism was one of boundary migration, Stead's idea was periodically revived until the work of Carpenter and Elam (1920) finally settled the matter in favour of boundary migration. Observations of grain boundary migration were one of the earliest forms of in-situ studies of microstructural evolution (McCrone and Chen 1949) and proved conclusively the general nature of the process, although even today the exact atomic scale processes involved remain the subject of debate.

3.5.2 Natural examples

The most commonly cited evidence for grain boundary migration in rocks is the presence of lobate boundaries (Fig. 3.5.1a), which indicate an internal energy driving force: recrystallisation. In fact a foam texture (all boundaries smoothly curved, triple junctions which form 120° inter-boundary angles) is also evidence for grain boundary migration and it indicates grain boundary energy controlled grain boundary migration, as the grain boundaries move to minimize their total length.

This leads to grain growth: the average increase in grain size as a function of time.

Another class of indicators is the formation of orientation families that are isolated (in 2D) sets of grains with the same crystallographic orientation, which were once part of the same grain, but have been isolated by grain boundary migration. (Fig. 3.5.1a, and see Urai et al., 1986).

Finally a new class of indicators has recently been described by Reinecke et al. (2000), Holness and Watt (2001), Jessell et al. (2003) and Nakamura et al. (2005) that relies on the change in chemistry, dislocation density levels, or impurity content behind a migrating boundary (Fig. 3.5.1b).

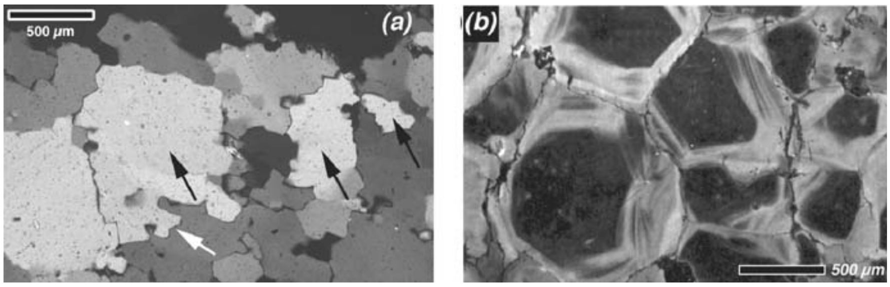


Fig. 3.5.1 Natural examples interpreted to demonstrate the past migration of grain boundaries. **(a)** Recrystallised quartz vein from Yudnamutana, South Australia, in cross-polarised light. Grains have serrated or lobate grain boundaries (white arrow). The black arrows show an orientation family: three grains that share the same lattice orientation and are interpreted to have derived from a single grain that has split into three by boundary migration. **(b)** Thermoluminescence image of growth bands in quartzite resulting from inclusion of water into the quartz lattice during grain boundary migration (Sample courtesy of M. Holness)

3.5.3 Inferred processes

The movement of grain boundaries may take place by the diffusion of single atoms from one grain across the boundary to the other grain through a supporting medium such as a fluid film, by the rotation and shuffling of atoms from one lattice to the other, or perhaps by piecemeal movement of clusters of atoms (Gottstein and Shvindlerman 1999). However, at the grain scale these processes are indistinguishable. This motion results in the migration of the boundary in the opposite direction to that of the atom or cluster motion.

3.5.4 Driving forces, controls and governing equations

There are various driving forces for grain boundary migration in rocks, such as of deformation induced dislocation density contrasts and surface energy of the boundaries themselves (see Chap. 3.3.4). The driving force for migration is the free energy reduction resulting from its motion, and will in fact in rocks always be the result of more than one driving force, since all crystals contain defects, and all grain boundaries have an inherent boundary energy. The rate at which a boundary migrates is both a function of this driving force, and the mobility of the boundary, which is an inherent property of a material (but which varies according to temperature, the presence or absence of fluids, the nature of the boundary, and the impurity content of both the grains and the boundary). Subgrain boundaries may also migrate, however the driving forces are typically smaller. The actual mobilities for most mineral grain boundaries are, unfortunately, poorly known and values have to be estimated. In all cases the prediction of the velocity of the boundary (v) at position p can be considered in terms of a fundamental material property: the grain boundary mobility (m, in units of m/Js); and a driving stress (σ_{gbm}), which is the driving force per unit area of grain boundary

$$v = \frac{dp}{dt} = \sigma_{gbm} m \Leftrightarrow \sigma_{gbm} = \frac{v}{m}. \tag{3.5.1}$$

3.5.5 Possible and actual simulation techniques

There have been a plethora of numerical simulation studies of grain boundary migration in engineering materials, mostly focusing on grain growth, and recrystallisation subsequent to deformation. The investigation of the grain growth phenomenon has in particular seen the application of a wide range of simulation techniques, in both 2D and 3D. These schemes can broadly be grouped into three classes:

1. Discrete schemes mapped onto regular grids, in which local changes in grain boundary position occur as a result of individual grid points switching their orientations, as a function of the local distribution of orientations. These include: Potts Models (Anderson et al., 1984; Kunaver and Kolar 1998) and Cellular Automata (Davies and Hong 1998).
2. Continuous schemes mapped onto regular grids, where the local change in orientation property is a function of the properties of the whole system, such as Phase Field techniques (Fan and Chen 1997; Le Bouar et al., 1998).

3. Discontinuous schemes mapped onto arbitrary networks. These schemes include Vertex Models (Soares et al., 1985; Bons and Urai 1992; Cleri 2000), Front-Tracking models (Wakai et al., 2000) and Finite Element techniques (Cocks and Gill 1996).

3.5.6 Implementation in *Elle*

In *Elle*, grain boundary migration is driven by the integrated Gibbs free energy reduction resulting from movement of the boundary. Components of the free energy can be the grain boundary surface energy, contrasts in notional defect energy across boundaries, or chemical energies related to phase changes. Grain boundary migration is simulated by calculating the driving force for boundary node movement, based on the orientation, length and properties of the grain boundary segments to which the node is attached, as well as the properties of the grains bounded by these segments.

Regardless of the driving force (f_{gbm}, positive in the direction of movement), the basic equation to describe grain boundary motion (Fig. 3.5.2) is the equality that relates the rate of work done (*dW/dt*) or the rate change of free energy change (*-dE/dt*) to that applied force and the rate of change of position (*dp/dt*=*v*) of the boundary:

$$f_{gbm} = \frac{-dE}{dp} \Leftrightarrow f_{gbm} v = \frac{-dE}{dt} = \frac{dW}{dt}. \tag{3.5.2}$$

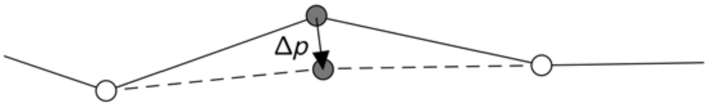


Fig. 3.5.2 Magnification of a moving grain boundary in *Elle*. The boundary is defined by a series of boundary nodes. The displacement (*p*) of a boundary node located at (*p*) is calculated by examining the forces on the boundary segments that adjoin that node. Because only one node is moved at a time, boundary segments pivot around the nodes neighbouring the moving node. The displacement here is exaggerated for clarity. In reality absolute displacements per time increment should only be a small fraction of the average distance between nodes

In order to predict the direction and magnitude of boundary migration we need to understand the local gradient in energy *(dE/dp*), i.e. we need to know how *E* changes as a function of position *p*. In the *Elle* implementation, where only one node is moved at a time, we can describe $E_{(p)}$ contours that describe the energy state for any hypothetical local position *p* to which the node could be moved (Fig. 3.5.3a).

We can simplify the problem if we assume, for very small Δp, that $E_{(p)}$ is a linear function of x- and y-coordinates of p. In that case we can calculate the function $E_{(x,y)}$ by calculating the total free energy for at least three positions around the current node position and making a linear fit through these three points (Fig. 3.5.3b). In the current *Elle* implementation the linear dependence of E on x and y is calculated from four trial positions at a very small distance s from the current node position (Fig. 3.5.3c). Essentially this means that the central Finite Difference method is employed.

It should be stressed that any hypothetical free energy "source" can be incorporated in the energy equation. This routine is therefore not restricted to particular types of boundary migration, but applies equally well to metamorphic reactions as to deformation-induced grain boundary migration.

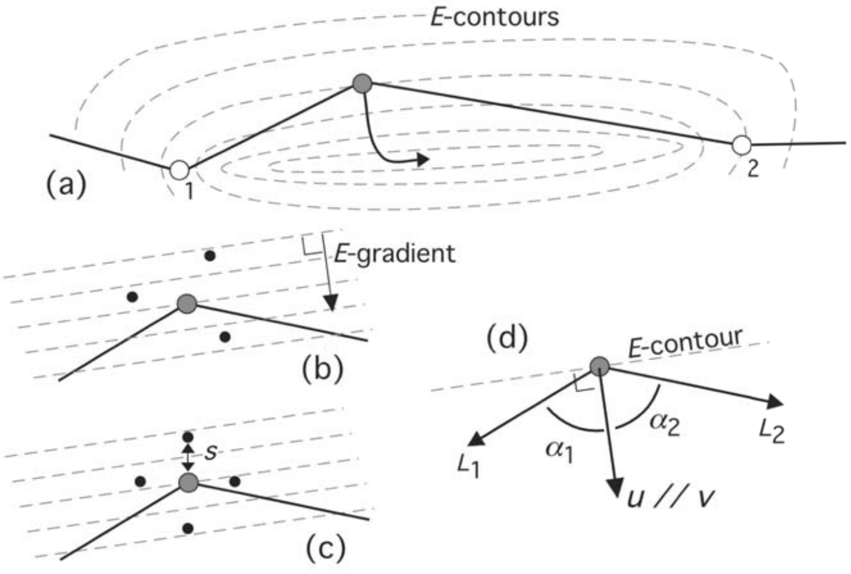


Fig. 3.5.3 The energy variation around a grain boundary node. **(a)** Hypothetical complex function of the energy function around a moving node. The arrow shows the path down the energy gradient that the node would follow if the neighbouring nodes would remain fixed. **(b)** At a finer scale we can simplify this function to a linear approximation that allows us to define the function based on only three trial locations (black dots) around the original boundary node location. **(c)** The current *Elle* implementation actually uses four trial positions at distance s from the original node location to calculate local gradients in the energy function in the x- and y-direction. **(d)** Definition of the angles α_i between segments joined at the moving node and the movement direction v

The direction u of maximum energy dissipation is easily calculated once the linear function $E_{(x,y)}$ is known. The node under consideration will move in the given direction u with a velocity v.

The rate of energy dissipation is equal to the sum of the rates of work done by each individual segment joined at the moving node. The work (W_i) for segment i is proportional to the normal force acting on the node and the normal component of the velocity of the moving node (Eq. 3.5.1). Let us now define L_i as the length of the i^{th} segment, and α_i as the angle between the vector to segment i and the movement direction (Fig. 3.5.3d), and σ_i^n being the normal stress acting on the segment. We can now write

$$\sum \frac{dW_i}{dt} = \tfrac{1}{2} \sum L_i \left|\sigma_i^n\right| \left|v\right| \sin(\alpha_i). \tag{3.5.3}$$

The factor 1/2 comes from the fact that only one side of the segment is moved, and the average velocity along each segment is thus only half the velocity of the moving node. From Eq. 3.5.1 it follows that

$$\left|\sigma_i^n\right| = \frac{\left|v\right| \sin(\alpha_i)}{m_i} \Leftrightarrow \sum \frac{dW_i}{dt} = \sum \frac{L_i \tfrac{1}{2} \left|v\right|^2 \sin^2(\alpha_i)}{m_i}. \tag{3.5.4}$$

We use Eq. 3.5.2 to describe the total work done by moving a node:

$$\left(\frac{dW}{dt}\right)_{node} = f_{gbm} v = \frac{-dE}{du} v. \tag{3.5.5}$$

Inserting Eq. 3.5.3 into 3.5.5 finally gives us an expression for the velocity v of the in the direction u:

$$\left(\frac{dW}{dt}\right)_{node} = \sum \left(\frac{dW}{dt}\right)_i \Leftrightarrow \frac{-dE}{du} \left|v\right| = \tfrac{1}{2} \sum \frac{L_i \left|v\right|^2 \sin^2(\alpha_i)}{m_i}, \tag{3.5.6}$$

$$\Leftrightarrow \left|v\right| = \frac{-2(dE/du)}{\sum \frac{L_i \sin^2(\alpha_i)}{m_i}}. \tag{3.5.7}$$

Equation 3.5.7 only contains variables that can be determined from the current state of the moving node and its surroundings. The final displacement of a node is calculated by simple linear integration of v over a small time increment (Δt), assuming that the velocity does not change during that time increment

$$p_{t+\Delta t} = p_t + v\Delta t. \tag{3.5.8}$$

Scaling and accuracy

When using this implementation (and most others!), the user should be aware that the above equations are only approximations. Strictly speaking, they only hold when Δt approaches zero. Furthermore, they only truly hold when node movement is perpendicular to the boundary segments, which would be on a straight boundary. Where three boundaries meet at a triple point, this is far from being the case. At all times, the user should carefully consider the settings for a simulation. These depend on several factors:

- the desired accuracy of the simulation;
- the values of the physical parameters of the processes and materials involved in the simulation (like grain boundary surface energy and mobility);
- practical constraints, such as available computer memory and duration of a simulation.

The user cannot just arbitrarily vary settings. The choice of the simulation "box" defines the size scale, while the choice of the time step (Δt) defines the time scale. The choice of the node distance (Δx) relative to the box size defines the spatial resultion of a simulation. Because the units of surface energy, mobility and strain energy also contain time and length, these units must be scaled according to the choice of Δx and Δt.

Whe grain boundary migration is driven by surface energy only, one can define a dimensionless parameter S to scale a simulation. S is defined as:

$$S = \frac{\gamma m \Delta t}{(\Delta x)^2} \tag{3.5.9}$$

Simulations run with the same value of S produce the same result. For example, doubling the grain boundary mobility, while halving the surface energy makes no difference for the final result of the simulation. The use of the scaling parameter can be illustrated with a simulation of the shrinking of a single circular grain in an infinite matrix. For the ideal case that the surface energy is constant, the analytical solution for the area ($A_{(t)}$) of the grain as a function of time is:

$$A_{(t)} = A_0 - 2\pi m \gamma t \tag{3.5.10}$$

This is a linear function and suitable to test the accuracy of the simulation. The case was simulated with a unit 1×1 model area, a unit time step, a unit surface energy and a grain boundary mobility of 10^{-6}. With these settings, a starting grain with an area of 0.18 should shrink in about 30,000 time steps (Fig. 3.5.4). Two settings for the node distance were used for the simulations: Δx=0.005 and 0.003. Using Eq. (3.5.9) we obtain scaling parameters of S=0.004 and 0.011, respectively. The simulations show a shrinkage rate that is slightly higher than the analytical solution (Fig. 3.5.4). The simulation with S=0.004 systematically overestimates the node shrinkage rate by about 2%, and the one with S=0.011 more than 5%.

We see that increasing the spatial resolution (Δx=0.003 instead of 0.005) actually decreases the accuracy of the node movement calculations, if the time step is kept the same. This is because incremental displacements of the nodes during one time step (Eq. 3.5.8) must be small relative to the node distances.

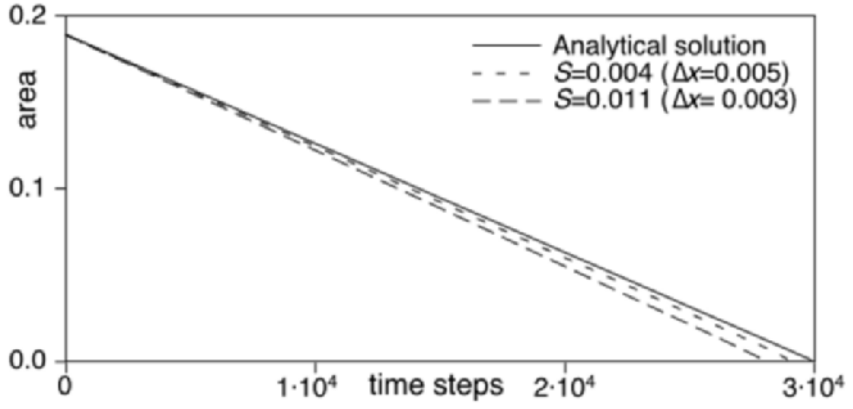


Fig. 3.5.4. Comparison between the analytical solution of the shrinking of an isolated circular grain and the numericl simulation with two different settings for the node distance, equivalent to a value of S = 0.004 and S = 0.011

The scaling factor S can be used to determine the accuracy of a simulation. Any combination of settings with S=0.004 will result in the same ~2% error in the shrinkage rate. If the time step would be reduced by a factor 10, reducing S to 0.0004, the simulation would be about 0.1% off the analytical solution. However, the simulation would take 300,000 time steps!

Let us suppose we want to simulate a quartz grain with an area of 20 mm^2 in a model area of 10×10 mm. The surface energy of quartz is about 0.5 J/m^2, and the grain boundary mobility about 10^{-15} m^2s/kg. Now suppose that we want a spatial resolution, defined by the node distance, of 10 μm (10^{-5} m). What would be the maximum duration of one time step that would give a satisfactory accuracy (say <2%, and therefore S=0.004) in the simulation? This can be determined by reordering Eq. (3.5.9):

$$\Delta t = \frac{S(\Delta x)^2}{\gamma m} \Leftrightarrow \Delta t = \frac{0.004\left(10^{-5}\right)^2}{0.5 \cdot 10^{-15}} = 800 \text{ s.}$$

The time step must be less than 800 s (13 min) for the desired accuracy in shrinkage rate of the grain. If one is satisfied with a 100 µm spatial resolution, one can increase the time step by a factor of one hundred to 80000 s, or 22 hours.

Increasing the spatial resultion (reducing node distance) by a factor of 10 increases the number of nodes by the same factor and the number of time steps by a factor of a hundred. This means RAM use is increased by a factor of 10 and the time a simulation takes by a factor of 1000. Memory use and duration of a simulation put practical constraints on the accuracy and spatial resolution of simulations. Every time, the user should find a compromise between spatial resolution (node distance), time resolution (Δt), calculation time, and memory use. *The user should always ensure that the outcome of a simulation is not (significantly) dependent on choice of node distance or* Δt.

3.5.7 Example *Elle* Run: Grain growth

Grain boundaries in a grain aggregate have a surface energy typically in the order of 0.1 - 1 J/m^2 (Urai et al., 1986 and refs therein). If we take a two-dimensional view of the problem, grain boundaries can migrate to achieve a local minimum surface energy configuration, given by straight boundaries and equal 120° angle intersections. Since the two cannot be achieved simultaneously, smoothly curved grain boundaries develop that meet at approximately 120°. This balance is very similar to the growth of bubbles in a foam (Glazier et al., 1987) and the result is a grain aggregate that resembles a foam and is therefore termed a foam texture. Larger than average, concave outward and many-sided grains on average grow, while smaller, convex and fewer-sided grains shrink and eventually disappear. Grain growth results, as the average grain size must increase as the total volume of grains remains the same, but their number decreases. Readers are referred to Smith (1964), Weaire and Rivier (1984), Anderson (1988) and Evans et al. (2001) for reviews on the topic. Grain growth is an important process in metamorphic rocks, and can for instance, modify or in some cases completely obliterate an existing deformation microstructure (Bons and Urai 1992; Bons et al., 2001).

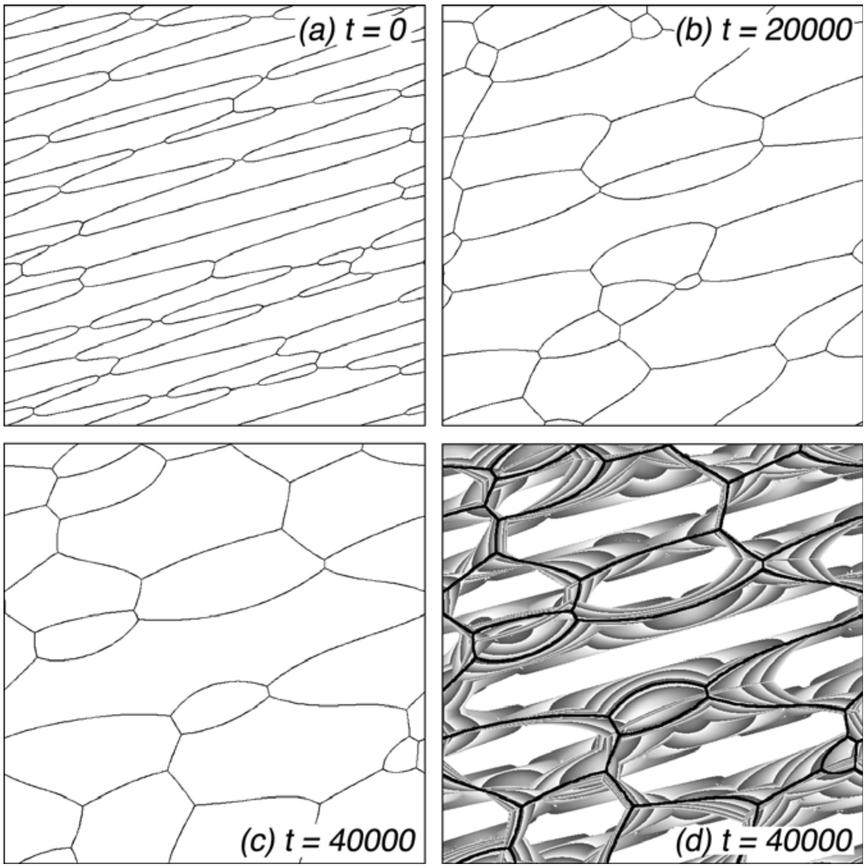


Fig. 3.5.5 (a-c) Example of a grain boundary migration experiment showing the evolution of grain boundary geometry driven by boundary energy reduction ("grain growth") in an originally foliated material at *t*=0, 20000 and 40000. **(d)** The successive positions of all grain boundaries at each time step during this time interval are shown in shades of grey that vary cyclically with time. White areas have never been swept by grain boundaries. EXPERIMENT 7

In this example (Fig. 3.5.5) we performed a simple grain growth experiment by taking an input file with a microstructure far from equilibrium in terms of grain shapes and evolve the grain boundaries by defining a boundary energy term, which provides the driving force for grain boundary migration.

3.5.8 Alternative implementations in *Elle*

The node movement routine that is implemented in *Elle* and that is described above is versatile: almost any driving force for grain boundary migration can be incorporated. The routine also deals well with variable mobilities of individual segments. Other implementations have been used in the past and may be useful for specific cases.

One way to model surface-energy driven grain boundary migration (grain growth) was already described in Chap. 2.3. Here a circle is fitted to the moving node and its two neighbours. The boundary velocity is inversely proportional to the radius of that circle, in ideal isotropic grain growth. The simple routine is implemented in *Elle* (*elle_gg*), but restricted to isotropic grain growth. For example, it cannot deal with surface energies or mobilities that are a function of the orientation of a surface relative to the crystallographic orientation of the adjacent grains.

Bons et al. (2001) and Piazolo et al. (2002) employed another routine (*elle_gbm*). This one is "in the same spirit" as the one described in section 3.5.6, but simpler. It does not calculate the gradient in the energy as a function of the position of a node around its original position. Instead, it takes four trial displacements at a small distance *(s)* from the original positions. First one displacement is chosen in a random direction, and then three others at 90° to each other. The displacement producing the maximum energy dissipation is chosen as the movement direction and the gradient in that direction determines the driving force and hence velocity of the node. Clearly, each individual node displacement is not exactly in the right direction, but on average nodes move in the right direction. In general, the grain boundary migration implementation described in this chapther is preferred over the other routines.

Symbols used in Chap. 3.5

$E_{(p)}$	Free energy state of a grain boundary (node) at position p [J]
f_{gbm}	General driving force to move a grain boundary (node) [N]
L_i	Length of a boundary segment linking a node and its ith neighbour [m]
m	Grain boundary mobility [m^2 s kg^{-1}, or m s kg^{-1} in 2D]
p (Δp)	Position (change in position) [m]
s	Small distance between trial position and original position of a node [m]
S	Dimensionless scaling factor
u	Direction of movement of a node, equal to the direction of maximum energy dissipation
v	Velocity of a grain boundary (node) [m s^{-1}]
W	Work done as a result of moving a grain boundary (node) [J]
α_i	Angle between normal vector to the ith boundary segment and the movement direction of a node
γ	Grain boudary surface energy [J m^{-2} = kg s^{-2}, or J m^{-1} = kg m s^{-2} in 2D]
Δt	Time step in a simulation [s]
Δx	User-defined distance between nodes [m]
σ_{gbm}	Driving stress (force per unit area of grain boundary) for grain boundary migration [Pa]
σ_i^n	Normal stress acting on the ith boundary segment [Pa]

3.6 Vein microstructures

Paul D. Bons and Daniel Koehn

3.6.1 Phenomenological Observations

Veins are mineral aggregates that grew by precipitation from a fluid in space created by dilatation of the host rock, such as the opening of a crack. Veins can have a variety of internal microstructures, ranging from equant blocky to extremely fibrous crystals, depending on factors such as the vein growth mechanism and conditions of growth (see Oliver and Bons, 2001, for a recent review of the topic). The microstructure is of particular interest for tectonic studies, since the shape and orientation of crystals often provide indications of the dynamic conditions during vein formation, such as stretching direction or type of deformation (e.g. Durney and Ramsay 1973).

The shape and arrangement of crystals inside a vein depends on a range of factors:

1. Spatial constraints. In some veins, crystals grow into an open, fluid-filled space, only constrained in their growth by their neighbours. In other veins, crystals grow into a narrow crack and are also constrained in their growth by the width and shape of the crack.
2. Crystal habit is a straightforward factor controlling the shape of grains, with naturally fibrous asbestos forming different microstructures than prismatic quartz under otherwise similar conditions.
3. The transport of dissolved components (the nutrients) towards the vein may also play a role in forming the microstructure.

Attempts have been made to numerically simulate vein microstructures in order to better understand their formation. Front-Tracking is one appropriate approach, since the growth of crystals in a vein involves the progressive movement of the crystal surfaces, usually into a fluid-filled space. The 2D model *Vein Growth* (Bons 2001) for growth of a row of crystals growing into a crack uses Front-Tracking, as does the model *Fringe Growth* (Koehn et al., 2000) that was derived from it. These two codes have not yet been incorporated into the *Elle* system, but are described here as they are currently the only codes available for crystal growth in constrained spaces. These models have been applied to address the following questions:

1. What is the shape of vein crystals that grow into a repeatedly opening crack-seal vein? Factors that play a role are mineral habit, crack opening distance and direction, opening frequency relative to mineral growth rate, and crack roughness (Hilgers et al., 2001).
2. How do complex fibre patterns in pressure fringes develop (Koehn et al., 2000)?
3. Why do some films of zeolites at first develop a crystallographic preferred orientation with the fastest growing direction oblique to the substrate (Bons and Bons 2003)?

3.6.2 Modeling of vein microstructures

In the model *Vein Growth* of Bons (2001) crystals are represented by polygons that are defined by nodes that are linked by straight segments. Movement of nodes changes the shape of the polygons, and hence the shape of the crystals. Growth of crystals is thus modelled by moving nodes.

The program models the growth of a row of crystals into a fluid-filled crack. Crystals stop growing when they reach the other side of the crack. However, the crack can open repeatedly to simulate the crack-seal process of Ramsay (1980). Rock, vein and fluid-filled crack are modelled with a raster where 0 (black) stands for rock and 255 (white) for vein or fluid. Opening of the crack is simulated by shifting all "rock" sites on the lower side of the crack by a given number of lattice points down and to the left or right (Fig. 3.6.1).

There are three types of nodes (Fig. 3.6.1):

1. Active nodes lie on crystal surfaces in contact with fluid. Moving these nodes in the direction of the fluid-filled crack simulates growth of vein crystals
2. Temporarily deactivated nodes. Once an active node moves onto a "wall rock" site on the bitmap, it is temporarily deactivated. If after a crack opening event the node is not on the wall-rock area anymore, it will be activated again to continue growth of the crystals.
3. Permanently deactivated nodes. Nodes on boundaries between crystals never move again, unless they are still on the fluid crystal interface.

To simulate growth of the crystals into the fluid-filled crack, rules have to be set that determine the direction and velocity by which nodes or boundary segments move.

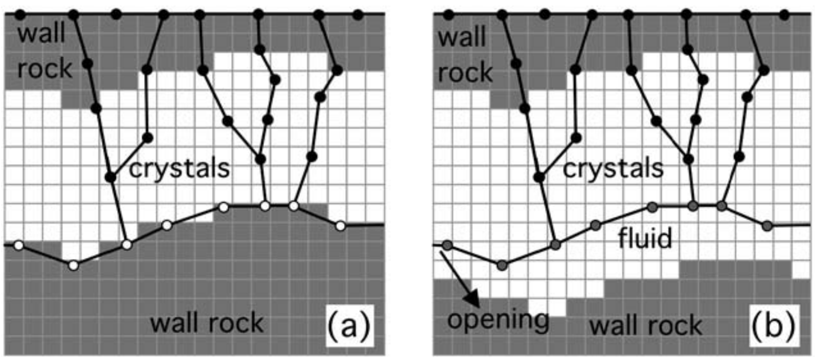


Fig. 3.6.1 Solid rock and crack (fluid or vein crystals) are defined by a bitmap. Vein crystals are described by nodes that define polygons. There are three types of nodes: black ones inside the vein or wall rock, which do not move any more, grey ones define the growth front of the vein and can move, while white ones are temporarily locked nodes that have reached the other side of the crack and cannot move until the crack opens again. **(a)** Situation where the vein crystals have all reached the other side of the crack as they moved onto a grey site on the bitmap. **(b)** After opening the crack 3 pixels down and 2 to the right (arrow) all locked nodes become mobile again

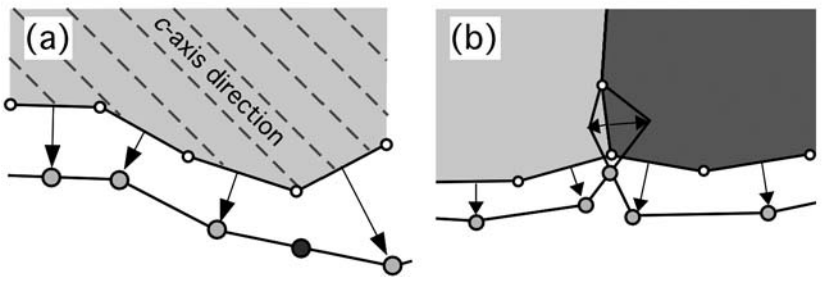


Fig. 3.6.2 (a) In *Vein Growth*, displacements of boundary segments are perpendicular to the segments. Nodes are positioned at the end of the displacement vectors. If segments become to short, a node is removed, while a node is added (dark extra node) when a segment is too long **(b)** Scheme for the movement of triple-junction node between two crystals and the fluid. Both crystals can grow outward according to the scheme shown in (a). The new position of the triple junction is on the new intersection between the two crystals

In *Vein Growth*, the displacements of all nodes are calculated for a small time increment (Fig. 3.6.2). Positions are then updated by applying the displacements to all nodes. *Vein Growth* calculates the growth rate of a segment as a function of the orientation of that segment and the lattice orientation, represented by a single "c-axis". Users can design their own functions, to create the desired crystal habit (Fig. 3.6.3). In *Vein Growth*, displacements are calculated for the middle of each segment, and new node positions are at the end of these segments. This effectively means that each node is shifted half a segment length each time step.

A special routine is used to move nodes at triple points between two crystals and the open crack (Fig. 3.6.2b). Here both crystals are allowed to grow outwards. The new triple point node position is where the two boundaries intersect. The result is a small dimple at the triple points, which is also observed in natural growth fronts. Models that only consider the growth of crystal facets, like the program *FACET* of Zhang and Adams (2002), do not incorporate this effect. However, the routine used by *Vein Growth* is only an approximation of what really happens at triple points. A proper routine should take into account the surface energies of the crystal-crystal and crystal-fluid surfaces, and diffusional effects in the fluid.

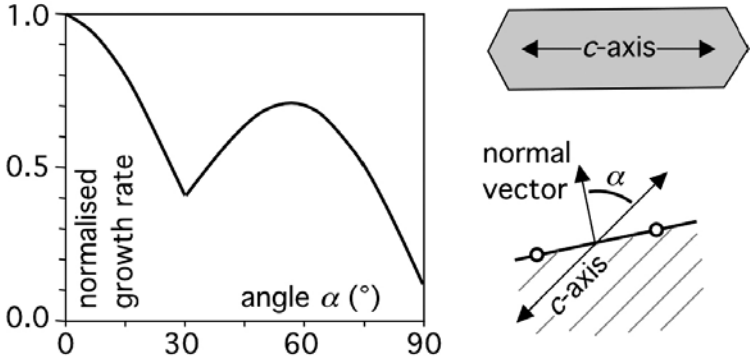


Fig. 3.6.3 Relationship between growth rate and angle (α) between the c-axis and a boundary segment. This function produces a prismatic crystal habit

Vein Growth and the spin-off *Fringe Growth* have been successfully applied to a variety of problems, as listed in the introduction. Some examples of their application are shown in Fig. 3.6.4. However, much could be improved in the modelling of veins. Although *Vein Growth* appears to correctly model the user-defined habit of crystals, Nollet et al. (2005) pointed out that the angle between the facets can deviate slightly from the true angle. They therefore suggested to develop a hybrid model, combining the approach of *FACET* (Zhang and Adams, 2002), which always gives correct angles between facets, and that of *Vein Growth*, which can cope with space constraints in a narrow crack.

Another remaining problem is that the models are so far purely kinematic as far as the creation of space for growth is concerned. The sequence of opening events of the fracture (Fig. 3.6.4b) or relative offset and rotation of object and fringes (Fig. 3.6.4c) are given at the start of a simulation. An improvement would be if a model would be combined with, for example, a Finite Element model to calculate the stresses and displacements, as well as failure of a healed crack. Such a model has not been developed yet.

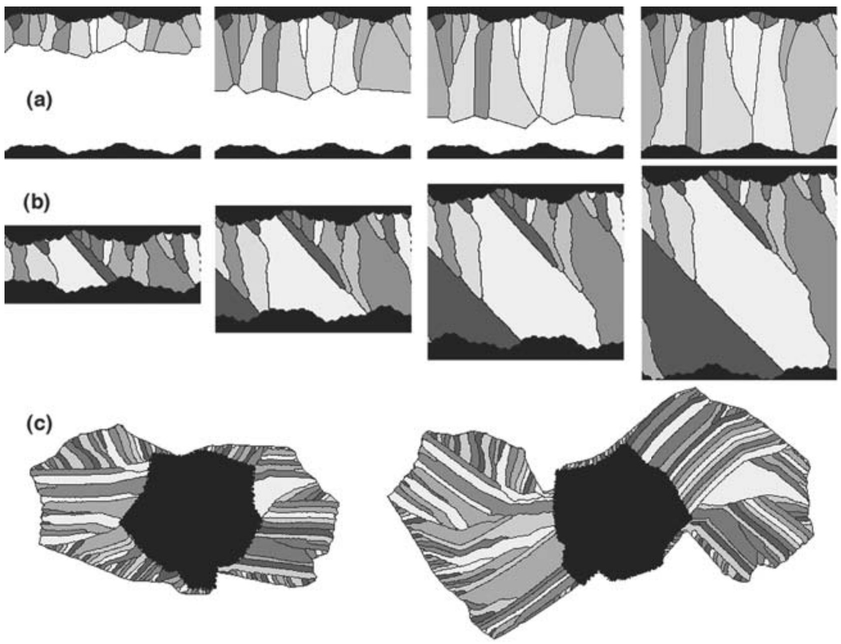


Fig. 3.6.4 Simulations of vein growth. **(a)** Growth of a "prismatic" mineral into an open crack in four stages. **(b)** Growth of the same mineral in a crack-seal vein with many oblique opening events. Shading in both simulations is according to *c*-axis orientation, from black (horizontal c-axis) to white (vertical c-axis). **(c)** Two stages of the simulation of the formation of a pressure fringe during dextral simple shear

3.7 Melt processes during grain growth

Jens Becker and Paul D. Bons

3.7.1 Phenomenological observations

Melt processes, especially those that lead to the existence of a three dimensional melt network, are of fundamental importance for melt migration and segregation, which is important in the upper mantle and the lower crust of the earth (McKenzie 1984; Scott and Stevenson, 1986; Sleep, 1988; Laporte et al., 1997; Rabinowicz et al., 2001; Wark et al., 2003). Furthermore, melt distribution and redistribution is a key factor in geological processes such as flow of partially molten (poly)-crystalline aggregates and the homogenization of partial melts by chemical diffusion (Cooper et al., 1984; Laporte et al., 1995). Many properties of systems with small amounts of melt depend on the distribution of the liquid phases at the grain scale. The same fundamental considerations also apply to the evolution of solid-state two-phase systems, and the code described here can equally well be applied to such phenomena as Ostwald ripening.

3.7.2 Natural examples

In high temperature experiments that were performed with major rock-forming minerals such as dunite + mafic or ultramafic melt (Waff and Bulau 1979; Hirth and Kohlstedt 1995; Faul 1997; Cmíral et al., 1998) or quartz + felsic melt (Jurewicz and Watson 1984; Laporte 1994; Laporte and Watson 1995) apparent disequilibrium features were described that deviated from the predicted regular melt geometry:

1. fully wetted grain boundaries and melt lenses on grain boundaries,
2. strongly distorted, melt-filled triple junctions,
3. large, multigrain-bounded, melt pools.

These disequilibrium features where shown to have a potentially large impact on the porosity–permeability function of partially molten rocks, especially at low melt-fraction such as considered in the upper mantle (Faul 1997; 2001). It has been recognized that variations in grain size and crystal lattice–controlled surface-energy anisotropy modifies the actual liquid distribution in natural aggregates (e.g. Waff and Faul 1992; Laporte and Watson 1995; Jung and Waff 1998; Wark et al., 2003). The frequent observation of straight crystal facets of grains in contact with melt supports the latter explanation for non-equilibrium melt geometries (Waff and Faul 1992; Hirth and Kohlstedt 1995; Jung and Waff 1998; Faul 1997). Although quartz is considered to be more isotropic than olivine, all the above mentioned non-equilibrium features were observed in quartz + melt systems as well (e.g. Laporte 1994).

3.7.3 Analogue modeling

In-situ experiments using analogue materials (for example using norcamphor and ethanol as substitutes for quartz and melt) have been performed to study the dynamic development of grain-boundary structures and melt segregation under deformation with a continuous observation of the experiment (Park and Means 1996; Bauer et al., 2000; Rosenberg and Handy 2000; Walte et al., 2003). Walte et al. (2003) found that all previously observed disequilibrium structures can develop due to the collapse of small grains during coarsening of the whole aggregate.

3.7.4 Driving forces

It is beyond the scope of this book to give a detailed view of how melt is produced within, or transported into, natural rocks. With *driving forces* we here mean forces that will influence the shape of the melt filled pockets within a rock and the redistribution of melt within a rock on the grain scale. These forces can be parameterized in the following way:

1. The permeability of the rock is a measure of how quickly melt can be redistributed to accommodate changes of the grain fabric (e.g. grain growth).
2. The surface energies of the melt and the surrounding grains influence the shape of the melt pocket and, to some extend, the way melt is redistributed within the grain fabric.
3. Although the melt fraction is not directly responsible for the shape of melt pockets it is important to know whether the melt fraction is fixed, if melt migrates into the rock (melt fraction increases) or is transported out of the rock or crystallizes (melt fraction decreases).

In the implementation that is presented here, we assume that the communication between melt pockets by diffusion is infinitely fast, so that only surface energy effects need to be considered.

3.7.5 Possible and actual techniques

Numerous theoretical predictions on the geometries and thresholds of melt pockets have been made (e.g. Laporte et al., 1997). These mainly geometrical evaluations do not involve the process of melt redistribution and hence fail to predict disequilibrium features that can frequently be observed in natural high-temperature experiments and analogue modelling.

High-temperature experiments, as have been performed, amongst others, by Waff and Bulau (1979), Hirth and Kohlstedt (1995), and Faul (1997) with natural rocks (or powders thereof) do not allow a full control (or measurement) of all involved parameters and also fall short of showing the processes involved, as they only show snapshots of the grain fabric at the time of quenching. In contrast, analogue models do show the processes involved and allow a certain control on the grain fabric and the parameters involved.

Numerical models allow a full control of all involved parameters, a continuous observation of the simulation and precise analysis of every stage of the simulation. However, some of the values for the necessary parameters, for example for surface energies, are far from being well studied.

3.7.6 Implementation in *Elle*

The implementation in *Elle* (*elle_melt*) uses flynns and bnodes to map the distribution of solid grains and melt. Each flynn can either be "solid" or "melt", and boundary segments therefore either represent solid-solid boundaries or solid-melt boundaries, each with an appropriate surface energy assigned to it. By setting the ratio between solid-solid (γ_{ss}) and solid-melt (γ_{sm}) surface energy, the user can set any desired wetting angle (ω), using:

$$\omega = 2\arccos\left(\frac{\gamma_{ss}}{2\gamma_{sm}}\right). \qquad (3.7.1)$$

The node movement is carried out according to the principles described in Chap. 3.5. The driving force for node movement has two components:

1. A component that moves the node in the direction to reduce the local total surface energy.
2. A component that moves the node in the direction to achieve the desired melt fraction. The magnitude of this force increases with increasing difference between actual and desired melt fraction.

The melt fraction can be set to remain constant, or to increase/decrease at a constant rate. This melt fraction is simply the summed area of all "melt-flynns" divided by the total model area. This means that melt can effectively move freely through the model, implying full connectivity along melt channels or, alternatively, relatively fast melt diffusion along solid-solid grain boundaries.

3.7.7 Example in *Elle*

The example shown below has a solid-solid to solid-liquid surface energy ratio such that the wetting angle is 10° at equilibrium. The melt fraction is fixed at 2%. The starting microstructure (Fig. 3.7.1a) already shows some disequilibrium features, so that not all melt pockets have wetting angles of exactly 10°. However during the start of the simulation these melt pockets quickly adjust to an equilibrium shape (convex triangle, E-shape). This behaviour was predicted by Laporte et al. (1997) for <60° wetting angles in a static system when grains have similar sizes.

Continuous grain growth due to a reduction of the overall surface energy of the aggregate disturbs the equilibrium wetting angles. This becomes obvious when small grains completely collapse. When the grains reach a certain critical size the total energy of their surfaces is relatively high because they have a very high surface curvature. Therefore they will dissolve or melt with increasing velocity. Decreasing their size just increases their surface curvature so once this processes starts they will collapse completely in a relatively short time (relative to grain boundary movements of the other grains). During the collapse, melt pockets are distorted and surrounding grains bulge towards the disappearing grain. The final collapse of the small grain leaves relatively large melt pockets (disequilibrium shape, D-shape). These disequilibrium structures disappear slowly (relative to their appearance) while the surrounding grains adjust their shape. Finally, melt pockets reappear that show the equilibrium shape and corresponding wetting angles.

The wetting angles of the melt pockets was initially set to 10° (indirectly through the ratio of surface energies), but later fluctuates. Especially during the formation of D-shaped melt pockets and the accompanying redistribution of melt, the E-shaped melt pockets display pronounced changes of their wetting angles. After destruction of a D-shaped melt pocket the wetting angles of the remaining melt pockets tend to adjust again to 10°. This is in contrast to what was predicted from e.g. Laporte et al. (1997). It appears that the wetting angle cannot be treated as a constant parameter at short time scales. This also implies that the mean curvature of solid-liquid interfaces is not constant.

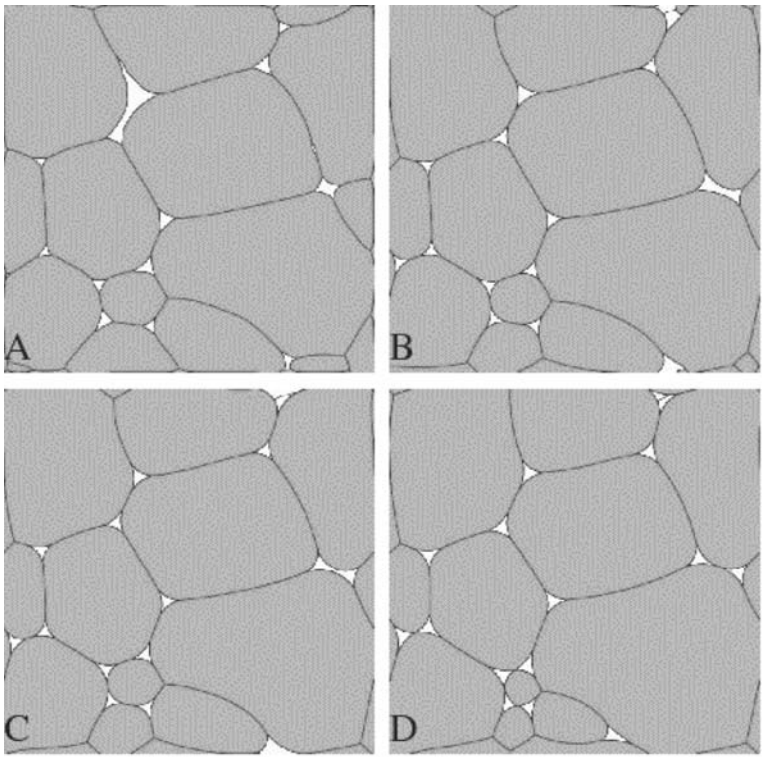


Fig. 3.7.1 Evolution in four stages (each 10 000 steps) of a grain aggregate with 2% melt in four stages. Equilibrium wetting angle was set at 10°. EXPERIMENT 8

Symbols used in Chap. 3.7

γ_{ss}, γ_{sm}	Surface energy of solid-solid and solid-melt boundaries [J m^{-2}]
ω	Wetting angle

3.8 *Basil:* stress and deformation in a viscous material

Greg Houseman, Terence Barr and Lynn Evans

3.8.1 Introduction

In this chapter we focus on the representation of creep processes that may be described in terms of 2D plane-strain viscous flow fields. For this purpose *Elle* may call on the *Basil* program, which computes internal stress-distribution, pressure, creep strain-rates, and displacement rates for a prescribed set of internal forces and boundary tractions, and an arbitrary internal distribution of creep strength. *Basil* assumes a constitutive law that can be cast in terms of an incompressible, possibly non-linear, viscous constitutive law.

The program *Basil* is a general-purpose finite deformation calculation package that is adapted to the solution of two distinct types of viscous flow problems: (a) viscous deformation of a 2D sheet in plane-strain, and (b) viscous deformation of a 2D sheet in thin sheet mode (in which the stress normal to the plane of calculation is externally set). In the context of *Elle* only the first of these functions is relevant, so we restrict our attention here to a description of how the plane-strain viscous flow problem is formulated. If viscous creep is the only active deformation process, *Basil* may be used independently to compute finite deformation fields using the methods described below. It has been applied to a range of geophysical problems, from thin-sheet representations of continental collision (Houseman and England 1996; Neil and Houseman 1997; Davis et al., 1997) to plane-strain representations of lithospheric (Houseman and Molnar 1997) or crustal-scale instability (Jull and Kelemen 2001), evolution of convergent orogenic belts (Houseman et al., 2000; Molnar and Houseman 2004; Billen and Houseman 2004), subduction of oceanic lithosphere (Houseman and Gubbins 1997), the development of shear zones (Barr and Houseman 1996; Grasemann and Stüwe 2001), and the evolution of grain scale microstructure (Bons et al., 1997; Piazolo and Passchier 2002). Because *Basil* works with dimensionless variables, the scale of the problem is arbitrary. Internal deformation and material properties, however, are assumed piecewise continuous i.e., discontinuities are permitted on surfaces (lines in 2D) that coincide with element boundaries.

In the present context, *Basil* is used within *Elle* to calculate the viscous strain rates and the associated stress field (including pressure) for typical applications that focus on grain-scale processes. The stress field is used in an *Elle* process (*elle_tbh*) to model the development of crystallographic preferred orientation and could also be used to compute the effects of grain boundary sliding and dissolution/precipitation. In general, the deformation is assumed to be continuous, but the material properties may be heterogeneous. We allow for piecewise continuous distributions of the material properties, meaning that properties are homogeneous within regions, but may jump at discontinuities. Such discontinuities can, for example, be grain boundaries that separate different mineral or rock types. The distribution of material properties is defined by arrays that may change with time as the calculation progresses. Although some processes in the *Elle* package permit anisotropy, the viscous deformation fields computed by *Basil* assume an isotropic constitutive relation (resistance to deformation is independent of the orientation or sign of the deviatoric stress field).

Our focus here is on the computation of finite deformation: strains that are large compared to the instantaneous elastic deformation that accompanies any stress field. We assume that elastic deformation is present also, but we only compute the viscous strain. The two components of strain are strictly separable if the stress-field is independent of time. In more general cases the interaction between elastic and viscous strains is small and the elastic strain can often be ignored. At any time we solve the general stress balance equations, by using the Finite Element method, to compute a flow field. We use these computed displacement rates to advect the material properties of the medium by time stepping, deforming the external boundaries and modifying the internal distribution of material properties at each step as we integrate forward in time. In the context of *Elle*, *Basil* is generally called upon to compute an instantaneous stress field and corresponding viscous flow field. The *Elle* program manages the time integration, including updating the medium properties (as described in section 3.1), and incorporating other microstructural processes as required.

In this section we firstly describe the assumptions and methods used in *Basil* , with results illustrated by examples of the viscous deformation of a relatively soft matrix around a harder porphyroblast, the whole exposed to a general shear. Secondly we describe how *Basil* is integrated into the *Elle* project, allowing the effects of viscous creep to be integrated through time concurrently with any of the other processes that are implemented in the *Elle* project.

3.8.2 Formulation and assumptions

The inertia terms are neglected in the balance of momentum for a viscous creeping fluid, so conservation of momentum in two dimensions requires a simple balance between internal stress in the continuous medium and the body force, which we assume here is provided by the gravity vector of amplitude g and direction defined by the dimensionless unit vector **e** acting on the specified density distribution ρ:

$$\frac{\partial \sigma_{\alpha\beta}}{\partial x_\beta} + \rho g e_\alpha = 0, \qquad (3.8.1)$$

with summation over the index β. The total stress components $\sigma_{\alpha\beta}$ are defined in terms of deviatoric stress $\tau_{\alpha\beta}$ and pressure p:

$$\sigma_{\alpha\beta} = p\delta_{\alpha\beta} + \tau_{\alpha\beta}, \qquad (3.8.2)$$

where $\delta_{\alpha\beta} = 1$ if $\alpha = \beta$, or else = 0.

Our flow field is described in terms of the velocity components (u,v) in the (x,y) plane. We assume that flow is incompressible, so:

$$\nabla \cdot \mathbf{u} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0. \qquad (3.8.3)$$

We also assume a constitutive law that relates deviatoric stress τ_{ij} to strain-rate $\dot{\varepsilon}_{ij}$, where the dot above the strain symbol indicates the time derivative:

$$\tau_{\alpha\beta} = 2\eta\dot{\varepsilon}_{\alpha\beta} = \eta\left(\frac{\partial u_\alpha}{\partial x_\beta} + \frac{\partial u_\beta}{\partial x_\alpha}\right). \qquad (3.8.4)$$

The symbol η denotes viscosity, which in general may be spatially variable, and specifically may depend on the magnitude of deviatoric stress, as is found experimentally for deformation of silicates. Such a relation, though non-linear, must be independent of any choice of reference frame, so it is formulated in terms of the second invariants of the deviatoric stress Θ and strain-rate $\dot{E}$, where

$$\dot{E} = \sqrt{\dot{\varepsilon}_{\alpha\beta}\dot{\varepsilon}_{\alpha\beta}}; \qquad \Theta = \sqrt{\tau_{\alpha\beta}\tau_{\alpha\beta}}; \qquad \text{and} \qquad \Theta = B\dot{E}^{(1/n)}, \qquad (3.8.5)$$

where B and n are material constants, which again may be spatially variable. The stress *versus* strain-rate exponent n for some silicate materials is about 3. B is referred to as the strength coefficient. The viscosity that is consistent with uniaxial experiments in which the strain-rate is observed to vary as the n-th power of the stress-difference is given by:

$$\eta = \frac{B}{2}\dot{E}^{\left(\frac{1}{n}-1\right)}, \tag{3.8.6}$$

so B is simply proportional to viscosity for a linear material (when n = 1).

Within *Basil* we use dimensionless variables: distance is rendered dimensionless by a characteristic length L, time by a characteristic time T_0, strain-rate by $1/T_0$, velocity by L/T_0, density by a characteristic density ρ_0, and viscosity by a characteristic viscosity η_0. Stress components are non-dimensionalised by the factor η_0/T_0. If $n \neq 1$, we choose $\eta_0 = (B/2)T_0^{(1-1/n)}$, consistent with Eq. 3.8.6. In dimensionless variables (denoted by $'$) Eq. 3.8.1 becomes:

$$\frac{\partial \sigma'_{\alpha\beta}}{\partial x'_\beta} + \left(\frac{g\rho_0 L T_0}{\eta_0}\right)\rho' e_\alpha = 0. \tag{3.8.7}$$

If deformation is driven by an imposed boundary velocity U_0, we generally assume $T_0 = L/U_0$, if driven by an imposed strain-rate ε_0, we generally assume $T_0 = 1/\varepsilon_0$. Equation 3.8.7 then includes the single dimensionless parameter F, which is a measure of the relative importance of gravitational stress to internal viscous stress:

$$F = \frac{g\rho_0 L T_0}{\eta_0}, \tag{3.8.8a}$$

$$\Leftrightarrow F = \frac{2g\rho_0 L T_0^{1/n}}{B_0} \qquad \text{if} \qquad n \neq 1. \tag{3.8.8b}$$

If the boundaries are passive, and the body force drives the flow, we may define $T_0 = \eta_0/(g\rho_0 L)$, so $F = 1$. The choice of how to define T_0 is left to the user, but must of course be consistent with the definition of boundary conditions, the specified value of F, and the other physical constants. For many small-scale deformation problems, F is negligible and is set to zero because $L \ll \eta_0/(g\rho_0 T_0)$, representing the situation in which body force is negligible compared to internal viscous stress when the material is forced to deform by externally imposed tractions or displacement rates.

To complete the specification of the problem, the distribution of forces (specified usually in terms of traction vectors or stress components) or of displacement rates on the boundary of the region is required. This requirement is discussed further in the next section.

3.8.3 Numerical method

To obtain a numerical solution for the flow field using the Finite Element method, the solution region in the *x-y* plane is discretised into triangular area segments or elements. Each triangular element is associated with three vertex nodes and three midpoint nodes (halfway between the three pairs of vertex nodes). The set of nodal velocities and pressures defines a solution everywhere, because we define the interpolated velocity and pressure fields within each element:

$$u_\alpha(x,y)=\sum_{k=1}^{6}u_{\alpha k}N_k(x,y); \qquad p(x,y)=\sum_{j=1}^{3}p_jL_j(x,y), \tag{3.8.9}$$

where $L_j(x,y)$ is the linear interpolation function which varies linearly in the (x,y) plane from one on vertex node j to zero on the other two vertex nodes, and $N_k(x,y)$ is the quadratic interpolation function which is one on node k and zero on the other five nodes.

The Finite Element method relies on a "weak" implementation of the dimensionless governing Eqs. 3.8.7 and 3.8.3, which are replaced by weighted integrals as follows. We use the Galerkin formulation in which the integral weighting functions of node *i*, $N_i(x,y)$ and $L_i(x,y)$ are precisely the interpolation functions defined in Eq. 3.8.9:

$$\int_\Omega N_i\frac{\partial\sigma_{\alpha\beta}}{\partial x_\beta}d\Omega=-F\int_\Omega N_i\rho e_\alpha\, d\Omega, \tag{3.8.10}$$

and

$$\int_\Omega L_i\frac{\partial u_\beta}{\partial x_\beta}d\Omega=0, \tag{3.8.11}$$

where summation over β is implicit in Eqs. 3.8.8 and 3.8.9, and the primes (′) are henceforth omitted from dimensionless variables. We use L_i instead of N_i for interpolation of pressures (Eq. 3.8.9) and for weighting function (Eq. 3.8.11) so that all terms in the system are represented with a consistent order of accuracy. This usage ensures numerical stability and symmetry of the resulting matrix. This formulation of the method was adapted from that of Yamada et al. (1975), as described by Huebner (1975).

Integrating Eq. 3.8.10 by parts gives:

$$\int_\Omega\left(\frac{\partial N_i}{\partial x_\beta}\sigma_{\alpha\beta}\right)d\Omega=\oint_{\partial\Omega}\left(N_in_\beta\sigma_{\alpha\beta}\right)dl+F\int_\Omega\left(N_i\rho e_\alpha\right)d\Omega. \tag{3.8.12}$$

The terms on the right of Eq. 3.8.12 represent the effects of the surface tractions $T_\alpha = n_\beta \sigma_{\alpha\beta}$ on the perimeter of the solution region, and the scaled body forces acting everywhere within the medium on the dimensionless density distribution ρ. Substituting Eqs. 3.8.1, 3.8.2 and 3.8.4 gives:

$$\int_\Omega \eta \frac{\partial N_i}{\partial x_\beta}\left(\frac{\partial u_\alpha}{\partial x_\beta}+\frac{\partial u_\beta}{\partial x_\alpha}\right)d\Omega+\int_\Omega\left(\frac{\partial N_i}{\partial x_\beta}p\right)d\Omega=\oint_{\partial\Omega}(N_i T_\alpha)dl+F\int_\Omega(N_i\rho e_\alpha)d\Omega. \quad (3.8.13)$$

Substituting Eq. 3.8.9, a set of discretised equations is obtained for x and y components of the momentum equation (Eq. 3.8.13), and for the continuity equation (Eq. 3.8.11). Equations are obtained for each of the nodal values u_i and v_i (from the momentum equation), and for each of the vertex pressure values (from the continuity equation) in each element. For the same node i, equations from different elements are summed in a process referred to as matrix assembly, to generate a complete matrix equation:

$$\begin{bmatrix}\mathbf{K}^1 & \mathbf{K}^4 & \mathbf{K}^7\\ \mathbf{K}^{4T} & \mathbf{K}^2 & \mathbf{K}^8\\ \mathbf{K}^{7T} & \mathbf{K}^{8T} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{u}\\ \mathbf{v}\\ \mathbf{p}\end{bmatrix}=\begin{bmatrix}\mathbf{a}\\ \mathbf{b}\\ \mathbf{0}\end{bmatrix}. \quad (3.8.14)$$

The numerical superscripts on the component matrices in Eq. 3.8.14 are simply labels, except that T denotes matrix transpose. The elements of the component matrices of Eq. 3.8.14 are:

$$K_{ij}^1 = \int_\Omega \eta\left(2\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x}+\frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}\right)d\Omega, \quad (3.8.15a)$$

$$K_{ij}^2 = \int_\Omega \eta\left(\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x}+2\frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}\right)d\Omega, \quad (3.8.15b)$$

$$K_{ij}^4 = \int_\Omega \eta\left(\frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial x}\right)d\Omega, \quad (3.8.15c)$$

$$K_{ij}^7 = \int_\Omega \eta\left(\frac{\partial N_i}{\partial x}L_j\right)d\Omega, \quad (3.8.15d)$$

$$K_{ij}^8 = \int_\Omega \eta\left(\frac{\partial N_i}{\partial y}L_j\right)d\Omega, \quad (3.8.15e)$$

$$a_i = \oint_{\partial\Omega} \left(N_i T_x\right) dl + \int_{\Omega} \left(N_i \rho g_x\right) d\Omega, \tag{3.8.15f}$$

$$b_i = \oint_{\partial\Omega} \left(N_i T_y\right) dl + \int_{\Omega} \left(N_i \rho g_y\right) d\Omega. \tag{3.8.15g}$$

The area integrals in Eq. 3.8.15 are computed using Gaussian quadrature with seven integration points. Dimensionless material properties such as viscosity coefficient η (or B in Eq. 3.8.6) and density ρ are defined at the seven Gaussian integration points in each triangular element to facilitate the computation of the integrals in Eq. 3.8.15, which are then simply expressed as a weighted sum of the integrand sampled at those seven points. The resulting integrals are exact if the integrand is quadratic, consistent with the order of approximation of the interpolation functions. For any node that is internal to the solution region, the contributions from the surface traction integrals on neighbouring elements cancel out. The traction integrals in a_i and b_i (Eq. 3.8.15f,g) are therefore only required on external boundaries.

Thus, we require for the solution of the problem that each of the two components of surface traction is specified everywhere on the boundary. A subset of these conditions may however be replaced by prescribed boundary velocity components ($u_i = U_0$). In that case the ii entry of the matrix $\mathbf{K}^1$ (or $\mathbf{K}^2$) is replaced with a number A which is at least 8 orders of magnitude greater than typical matrix entries, and the i-th entry of $\mathbf{a}$ (or $\mathbf{b}$) (Eq. 3.8.14) is replaced with AU_0. In that case the corresponding surface traction component is not required, and the velocity condition will be satisfied when the matrix equation is solved.

If we require spatially periodic solutions (the solution repeats with period λ in the x-direction for example), we must ensure continuity of both traction and displacement-rate components on the two corresponding boundaries. The boundary integrals in Eqs. 3.8.15f,g for the periodic boundary nodes are now unknown, however, so we incorporate the additional unknowns in a modified system of equations that ensures continuity of traction and displacement rates:

$$\begin{bmatrix} \mathbf{K}^1 & \mathbf{K}^4 & \mathbf{K}^7 & \mathbf{K}^p & \mathbf{0} \\ \mathbf{K}^{4T} & \mathbf{K}^2 & \mathbf{K}^8 & \mathbf{0} & \mathbf{K}^p \\ \mathbf{K}^{7T} & \mathbf{K}^{8T} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{K}^{pT} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}^{pT} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \\ \mathbf{p} \\ \mathbf{d}_x \\ \mathbf{d}_y \end{bmatrix} = \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}. \tag{3.8.16}$$

The vectors $\mathbf{d}_x$ and $\mathbf{d}_y$ comprise the set of unknown nodal traction components on the periodic boundary nodes (the first integrals in Eq.

3.815f,g), and the bottom row of the modified matrix equation (Eq. 3.8.16) ensures continuity of velocity components, with $\mathbf{K}^p$ generally zero except for +1 in position p and -1 in position q, with p and q being the corresponding node numbers on the two parts of the periodic boundary.

One further complication arises with boundary conditions: if the normal component of traction is nowhere specified on the external boundary, the resulting pressure field is undetermined to within an arbitrary additive constant. The pressure field in the solution returned by the conjugate gradient solver then includes an arbitrary unconstrained constant, which in practice is removed by subtracting the mean pressure from the pressure field.

Following assembly of the matrix, Eq. 3.8.14 (or Eq. 3.8.16) is solved by use of the pre-conditioned conjugate gradient algorithm. The matrix is naturally sparse, with non-zero elements occurring at irregular intervals in each row, so only the non-zero elements and their associated column numbers are stored. The method of conjugate gradients is well adapted to this form of matrix storage and solution is relatively efficient. Basic preconditioning of the matrix is applied by use of a diagonal preconditioning matrix, which is (for Eq. 3.8.14):

$$\begin{bmatrix} \mathbf{P}^1 & 0 & 0 \\ 0 & \mathbf{P}^2 & 0 \\ 0 & 0 & \mathbf{P}^3 \end{bmatrix} \quad \text{where} \quad \mathbf{P}^1 = diag\left[\mathbf{K}^1\right], \quad \mathbf{P}^2 = diag\left[\mathbf{K}^2\right],$$

$$\mathbf{P}^3 = diag\left[\begin{bmatrix}\mathbf{K}^{7T} & \mathbf{K}^{8T}\end{bmatrix} \begin{bmatrix} \mathbf{P}^1 & 0 \\ 0 & \mathbf{P}^2 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{K}^7 \\ \mathbf{K}^8 \end{bmatrix}\right]. \tag{3.8.17}$$

Conjugate gradients rely on iteration to converge to a solution. We test for convergence by measuring the norm of the residual vector at each iteration, and cease iterations once the mean square residual is sufficiently small. Typically convergence requires many fewer conjugate gradient iterations than the theoretical requirement of the number of degrees of freedom.

For non-linear materials ($n \neq 1$ in Eq. 3.8.6) the matrix entries (Eq. 3.8.15a-e) depend on the solution vector, so a second layer of iterations is required. Starting with an initial estimate of the solution field $\{u_i, v_i, p_i\}$ the matrix entries are computed using Eqs. 3.8.15 and 3.8.5-6. In each iterative step the matrix equation is solved and the matrix entries are then recomputed based on the new solution. Convergence of the non-linear iterations is attained when successive solutions differ by a sufficiently small amount as judged by the mean square norm of the difference between the two solutions, relative to the mean square norm of the solution.

The solution method described so far provides the instantaneous solution for two components of the velocity field and the pressure field. To advance the solution from one time level to the next, *Basil* moves the individual vertex nodes according to:

$$\frac{dX_\alpha}{dt} = u_\alpha . \qquad (3.8.18)$$

We use a two-step predictor-corrector method in which the first step uses the velocity field at time level n to advance the coordinates from time level n to level n+1, in order to compute a trial solution at time level n+1. In the correction step, the average of the velocity field at level n and the trial velocity field at level n+1 are used to advance X_α from level n to the corrected X_α at level n+1. At that point the trial velocity field provides a relatively accurate initial estimate for the corrected velocity field at level n+1. Material properties (density, viscosity) are transported with the deforming mesh, with minor corrections to account for advection of properties at midpoint nodes (which must remain midpoints in an updated triangular mesh).

The preceding formulation can be applied to any triangular mesh. Routines within *Basil* can generate simple regular triangular meshes based on either equilateral triangles with hexagonal symmetry, or on right angle triangles obtained by subdivision of rectangles. Irregular geometry however requires more flexibility, and we have found that the program *Triangle* (Shewchuk, 1996) provides the flexibility to generate a triangular mesh that preserves defined boundaries, for example those of a clast or crystal whose properties differ from the background material. The *Triangle* algorithm ensures by subdivision of boundary segments that all triangles in the resulting mesh have an area less than a specified maximum. The resulting mesh is irregular but relatively homogeneous, comprising approximately equidimensional triangles whose areas are less than, but close to, the specified maximum area. Construction of such a mesh requires an input *poly* file (Shewchuk 2005) that provides *Basil* with the information necessary to define the grain and external boundary shape, specified in the form of boundary polygon coordinates and grain numbers.

3.8.4 Porphyroblast example solution

The following example of a *Basil* application describes a model problem of a single hard inclusion in a softer matrix to which simple shear boundary conditions are applied. Bons et al. (1997), Tenczer et al. (2001) and Biermeier et al. (2001) have described such models.

The porphyroblast in Fig. 3.8.1 is represented by a circular disc that has an effective viscosity 100 times greater than that of the surrounding matrix, and both the porphyroblast and surrounding matrix are assumed to have a constant Newtonian viscosity; i.e. a stress exponent of $n = 1$. The top and bottom boundaries have applied tangential velocities of equal magnitude but opposite sign, and normal velocities set to zero. Periodicity is assumed in the x-direction in an effort to simulate a laboratory ring shear apparatus, in which the deforming material occupies an annular region with boundaries only on the inside and outside perimeters (e.g., Piazolo and Passchier 2002). The traction and velocity components are required within the solution to be equal at corresponding points on the two side boundaries.

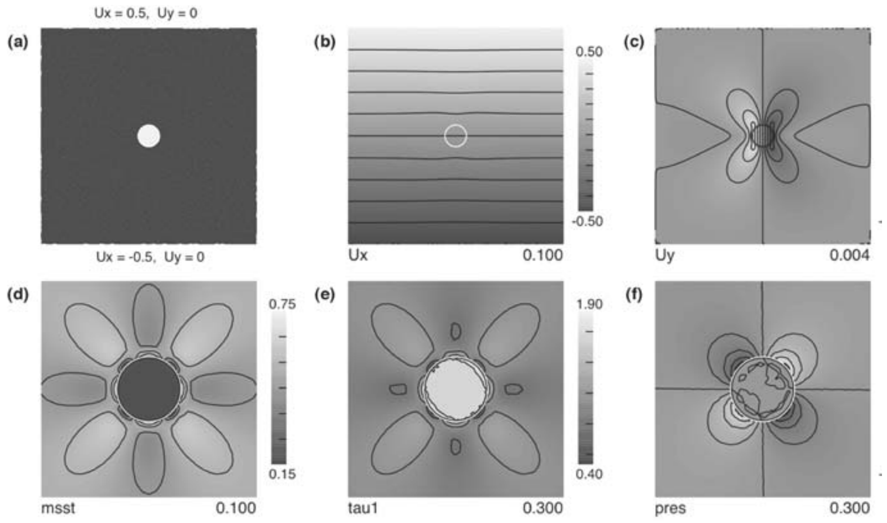


Fig. 3.8.1 Time-zero deformation calculation of a relatively rigid circular porphyroclast in a matrix undergoing 2D dextral simple shear. Six frames show: **(a)** viscosity variation (the light region has 100 times greater viscosity than the dark matrix), **(b)** x-component of velocity, **(c)** y-component of velocity, **(d)** the maximum shear strain rate in the x-y plane, **(e)** the maximum principal deviatoric stress, and **(f)** the pressure. (a-c) show the full solution region whereas (d-f) show a 3x zoom of the central region. This image is produced using the *Sybil* program, specifically designed for the graphical display of *Basil* solutions. EXPERIMENT 9

The effect of the relatively rigid inclusion in this calculation is to introduce a perturbation to the velocity field which is small compared to the overall shear flow, but clearly resolved in the vertical component (Fig. 3.8.1c), and showing the clockwise rotation of the inclusion ($u_y >$ 0 on the left, $u_y < 0$ on the right). The pressure field shows a clear dipolar signal outside the strong inclusion (Fig. 3.8.1f), with increased compression ($p < 0$) to the upper left and lower right and dilatation (p > 0) in the other two quadrants. The discontinuity at the perimeter of the inclusion is cleanly represented in the calculation, but interpolation of the final pressure solution produces some noise on the contours in the vicinity of the discontinuity in Fig. 3.8.1f. The maximum shear strain rates (Fig. 3.8.1d) and the maximum principal deviatoric stress show maxima (found at about twice the radius of the strong inclusion) that are clearly aligned with the lobes of maxima and minima in the pressure field. These frames also show the relatively small strain-rates and relatively large deviatoric stress within the inclusion.

3.8.5 Implementation in *Elle*: Growth and deformation of a porphyroblast

Flexibility in the representation of different physical processes is optimised by the modular structure of the *Elle* project and the external routines on which it depends. Within *Elle* an experiment cycle is defined by successive calls to program modules. For example: *elle2poly* → *Basil* → *basil2ell* → *reposition* → *elle_expand* defines an experiment in which *Elle* calls *Basil* to calculate the stress field and instantaneous deformation rates, then updates the spatial coordinates of the mesh defining the grain boundary structure (*reposition*), then executes a porphyroblast-growth procedure (*elle_expand*), and repeats this sequence at every time step. Data structures are transferred from *Elle* to *Basil* using the module *elle2poly*, which reads an *Elle* data file and creates a *poly* file for input to *Basil* (for examples, see Appendix B), specifying geometry and physical parameters. Similarly *basil2elle* is used to return velocity, strain-rate, and stress fields to *Elle*. The *Elle* microstructure, comprising a 2D network of grains and their associated boundaries and material properties, is periodic in both x and y directions. Grain boundaries are defined by a set of grain boundary nodes (bnodes). Within grains, variation in material properties may be defined on a set of Lagrangian marker points (unodes) regularly or randomly distributed in the 2D plane. These are used to track material properties as the deformation progresses. Within *Basil*, grains are clipped to a unit cell by creating temporary bnodes and segments on the external boundary of this cell. The *Basil* input file constructed by *elle2poly* therefore must specify all of the grain boundary segments, together with viscosity coefficient (η or B, Eq. 3.8.6). The viscosity coefficient and stress versus strain-rate exponent n may be constant within a grain or may have a variation defined by values attached to the unodes.

Elle uses the *Basil* velocity field to update the grain boundary node (bnode) positions and to obtain interpolated stress and/or strain-rate components at all required bnode and unode locations. The data required by *Elle* depend on the specific application and are specified by keywords in the input *Elle* file. These data potentially include the components of the strain-rate or stress tensors and their invariants. Attributes at grain boundary nodes (bnodes) are obtained directly from the corresponding *Basil* vertex node positions while attributes at internal nodes (unodes) are interpolated from the *Basil* solution. The local stress states and strain tensors may then be used in other processes such as the calculation of crystallographic rotation and work hardening.

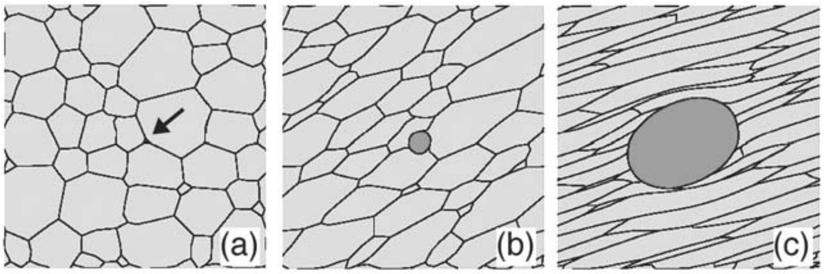


Fig. 3.8.2 Growth of a porphyroblast in a crystalline matrix: **(a)** the initial microstructure contains a grain (dark grey) with a viscosity defined to be 5 times that of the surrounding matrix. Arrow shows the site where a porphyroblast nucleates **(b)** and **(c)** show the structure after elapsed dimensionless times of 1 and 3 (50 and 150 dimensionless time steps of length 0.02). The harder region has grown, while the grains in the matrix have deformed in simple shear. Mechanical boundary conditions are the same as for Fig. 3.8.1. EXPERIMENT 9

In the example shown in Fig. 3.8.2 we use *Elle* and *Basil* to study the growth of a porphyroblast in a crystalline matrix that is subject to simple shear. The 2D square domain of unit dimensionless length shows a crystalline matrix with irregular grain size, (viscosity coefficient 1, representing quartz), surrounding a small grain of harder material (viscosity coefficient 5, representing garnet). We assume Newtonian constant viscosity (stress exponent $n = 1$) in both materials. During each time step, *Elle* simulates an increment of porphyroblast growth, simply represented in the *elle_expand* process by moving the grain boundary nodes of the porphyroblast outward at a dimensionless velocity of

$7.5\cdot10^{-2}$ in a direction that is radial to the local curvature of the boundary (the constant rate is used in this example for simplicity, but crystal growth should generally be driven by processes which take into account the local grain boundary energy and chemistry, e.g. Sect. 4.1). Additional bnodes are added to the growing porphyroblast by interpolation and removed from the matrix grains where they are overgrown by the porphyroblast. The elongation and rounding of the porphyroblast evident in Fig. 3.8.2.b is the outcome of the competing processes of porphyroblast growth, grain rotation and inhomogeneous simple shear. The grain boundaries of the background matrix are distorted by the shearing of the matrix but otherwise are not affected by the porphyroblast growth.

The crystalline matrix within *Elle* is defined by the set of bnodes on the boundaries of the polygonal grains. Each time *Basil* is called, it constructs a mesh that includes the *Elle* grain boundary nodes and grain boundaries (Fig. 3.8.3a). In this calculation porphyroblast growth is affecting only the boundaries of the garnet grain. The other grain boundaries in the matrix simply act as material tracers in the deformation field. Because the solution is periodic in the *x*-direction, nodes that exit the unit cell on the left boundary reappear immediately on the right, and vice versa. Thus the unit cell can be used to show the full solution at any time even though a finite strain of 3 is developed in the crystalline matrix by the end of the experiment (Fig. 3.8.2c). In principle, measurements of the grain geometry should constrain the relative rates of shear strain and grain growth, if the apparent viscosity contrast between the two minerals is known. In experiments however, one could turn this statement around and use measured rates of shear and porphyroblast growth to constrain the apparent viscosity contrast between the two minerals.

In the example described above, the Finite Element mesh was constructed using the grain boundary nodes to constrain the triangulation as illustrated in Fig. 3.8.3a. There is a limit, however, to the strain that can be obtained with this method of mesh construction. Problems arise as the matrix grains become long and narrow and the *Triangle* routine attempts to create triangles that are too many and too small. Available memory is used up or the triangles may be too quickly collapsed in regions of high strain-rate, halting the calculation. We have therefore also implemented an alternative re-meshing strategy, which allows us to simulate higher strain by mapping the *Elle* node data onto a regular *Basil* mesh at each step (Fig. 3.8.3b). This re-meshing strategy requires that viscosity values from the *Elle* data file are mapped appropriately to the Finite Element mesh constructed by *Basil*, and it requires that solution parameters (displacement rates, stress components, etc.) required at *Elle* bnodes are obtained by interpolation from the regular Basil grid.

These interpolations may introduce some additional inaccuracy (compared to the method of Fig. 3.8.3.a), but this method is more robust when the grains are highly elongated.

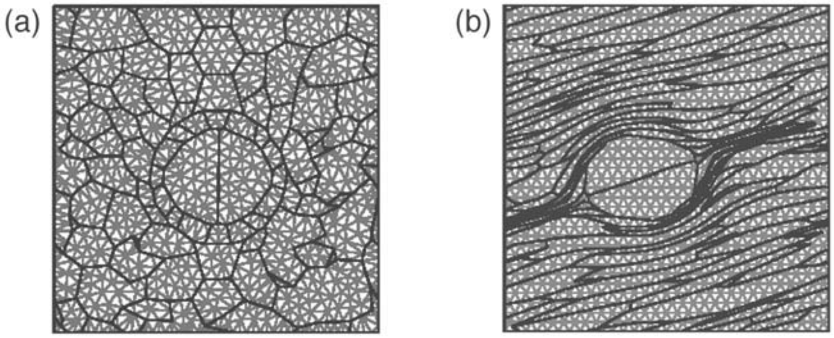


Fig. 3.8.3 Examples of the two methods of superimposing a Finite Element mesh on top of a grain boundary structure: **(a)** Finite Element mesh created by *Basil* using the *Triangle* program: the grain boundary nodes and segments defined in the *Elle* file (black) are preserved in the *Basil* computational mesh (grey). **(b)** Regular Finite Element mesh created by *Basil* (grey), using interpolated properties from the grain boundary nodes and segments defined in the *Elle* file (black)

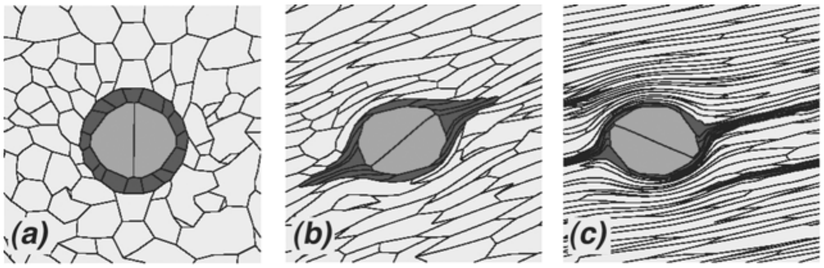


Fig. 3.8.4 Rotation of a porphyroblast (medium grey: viscosity $50\eta_o$) surrounded by a weak layer of crystals (dark grey: viscosity $0.5\eta_o$) embedded in a matrix (light grey: viscosity η_o) undergoing simple shear. As the deformation proceeds, the grains in the matrix and the soft layer become elongated while the hard central region rotates but retains its shape. The total finite shear strain is: **(a)** 0, **(b)** 2, and **(c)** 5. Boundary conditions are as in Fig. 3.8.1. EXPERIMENT 9

In a final example we show a dynamic shear experiment in which a constant volume porphyroblast is embedded in a matrix that undergoes a large simple shear strain (Fig. 3.8.4). In this case the viscosity of the porphyroblast is 50 times that of the background matrix, and a layer of grains whose viscosities are half that of the matrix surrounds the porphyroblast, and mechanical boundary conditions are the same as those specified for Fig. 3.8.1. With no growth of the porphyroblast in this experiment, the *Elle* cycle is simply *elle2poly* → *Basil* → *basil2elle* → *reposition*. The strong porphyroblast rotates at a rate of approximately 0.4 radians per unit time in this experiment, but undergoes little deformation while the background matrix is sheared to a total finite strain of 5. In this experiment we used the re-meshing method of Fig. 3.8.3b to continue the calculation to these large strains. The relatively weak (dark grey) grains are thinned and quickly distorted into a narrow band on which shear strain is concentrated.

3.8.6 Conclusions

In this section we have summarised the computation method used by the *Basil* program to solve general viscous flow problems in 2D plane-strain, as represented by a simple Stokes-type formulation of the momentum balance equations for incompressible flow of a very viscous fluid. The program is formulated using dimensionless variables based on a time and length scale chosen by the user. Arbitrary distributions of body force and material strength may be designated within the framework of a Newtonian or power-law viscosity. The *Basil* program may be used independently for 2D deformation experiments, or included within the *Elle* time step cycle to compute the deformation fields at very large strain or the interaction of deformation with other microstructural processes, such as grain growth.

Symbols used in Chap. 3.8

a, b	Load vectors which include the known boundary conditions
α, β	Indices
B	Strength coefficient (material constant)
$\mathbf{d}_x$, $\mathbf{d}_y$	Set of unknown nodal traction components on periodic boundaries
e	Direction of gravity vector
$\dot{\varepsilon}_{ij}$, ε	Strain rate tensor, imposed strain rate
$\dot{E}$	Second invariant of the strain rate tensor
F	Dimensionless parameter that is a measure of the relative importance of gravitational to internal viscous stress
g	Gravitational accelleration
K	Components of the stiffness matrix
L	Characteristic length
$L_j(x,y)$	Linear interpolation function for vertex node j in the (x,y) plane
n	Stress exponent (material constant)
$N_k(x,y)$	Quadratic interpolation function for vertex node k in the (x,y) plane
P	Components of the diagonal preconditioning matrix
p	Vector of pressure values at each vertex node
ρ, ρ_0	Density, characteristic density
σ	Stress
T_0	Characteristic time
Θ	Second invariant of the deviatoric stress tensor
τ_{ij}	Deviatoric stress tensor
u, v	Vectors of velocities in x and y directions at each node
U_0	Imposed boundary velocity
η	Viscosity [Pa s]
X_α	Position vector of a vertex node
Ω	Area of a finite element

3.9 A hybrid FEM/Taylor Bishop Hill method

Mark W. Jessell

3.9.1 Introduction

It is commonly observed that there is a pattern of preferred orientation of crystal lattice orientations in rocks (Fig. 3.9.1). Although many theories have been put forward to explain these patterns, they have long been accepted to be at least in part due to the reorientation of the lattice during deformation (Jessell and Lister 1990). The change in lattice orientation of part or all of a grain can result from both intra-crystalline and inter-crystalline processes including dislocation glide, twinning, kinking, grain rotation, subgrain formation and grain boundary migration.

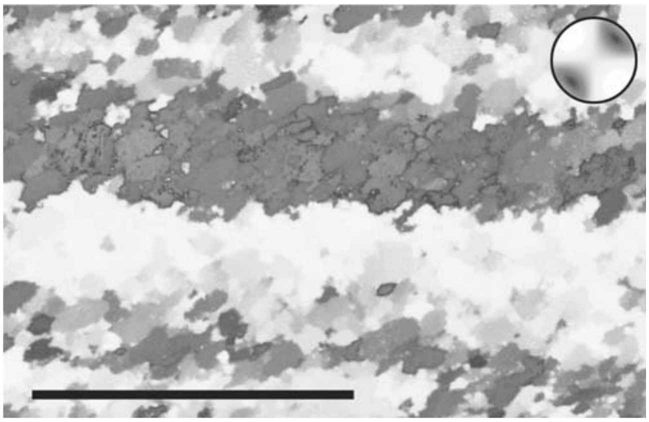


Fig. 3.9.1 Quartz mylonite from the Corvatsch mylonite, SE Switzerland. Grey tones represent c-axis orientations as measured by the CIP system, showing the development of a domainal fabric where two different domains each with a strong c-axis preferred orientation. The inset figure in the top right hand corner provides the representation of grey tones as a function of orientation, as displayed on a lower hemisphere equal-area stereogram. Scale bar is 1mm. Fromr R. Heilbronner, (2000). Measured by Christian Pauli

In this section we describe a hybrid Finite Element Model/Taylor-Bishop-Hill approach to predict the change in orientations resulting from deformation by dislocation glide in single-phase rocks. A different approach was taken by Bons and den Brok (2000), who used homogeneous stress instead of homogeneous strain rate to model lattice reorientations due to dissolution-precipitation creep, but that code is not currently available in the *Elle* system.

3.9.2 Models of Lattice Rotation during Deformation

The numerical basis for most theories of lattice rotation induced by dislocation glide was the paper by Taylor (1938) and subsequently reformulated by Bishop and Hill (1951). Taylor's original model relies on the assumption of kinematic compatibility (often referred to as the homogeneous strain assumption), which assumes that in a deforming polycrystal, all grains deform equally and homogeneously, regardless of their lattice orientation (Fig. 3.9.2a,b). The full set of assumptions for the 'Full Constraint' Taylor-Bishop-Hill (TBH) calculation is:

1. Dislocation glide is the only active deformation mechanism.
2. Strain is homogeneous at all scales.
3. Each grain has at least 5 independent slip systems (i.e. no slip system is capable of reproducing the strain achieved by any other system, also known as the von Mises criterion).
4. The minimum work necessary is carried out by the crystal to achieve the pre-defined strain
5. Each slip system has a defined Critical Resolved Shear Stress (CRSS) value: this is the shear stress (resolved in the direction of the slip vector and in the slip plane) necessary to induce dislocations to move in this plane. Below this value the dislocations do not move at all (hence the term crystalline plasticity, because they are assumed to behave as perfectly plastic materials).
6. The activity on any one slip system results in a small increment of simple shear parallel to the slip plane, in the direction of the slip vector. Until the entire strain increment can be achieved for a grain, no deformation takes place at all, so the stress gradually increases until the CRSS threshold is surpassed on just enough slip systems to allow the specific strain to take place. Once this state is achieved, the work term can be calculated as the products of the small strains achieved by each slip system and the imposed stress.

The assumptions of homogeneous strain and 5 independent slip systems demonstrably do not hold for engineering materials such as metals, let alone rocks and minerals, since most important rock forming minerals have low crystal symmetry, and hence high plastic anisotropy. As a consequence, a number of modified 'Relaxed Constraint' (Fig. 3.9.2c, Van Houtte 1988) or 'Self-Consistent' models have been developed (Fig. 3.9.2d, Hutchinson 1970) that try to address this problem by, in the first case, allowing strain heterogeneity in certain spatial directions, or in the second case, by embedding each grain in an uniform medium that has properties representing the average of the whole polycrystal. If sufficient computer power is available, these simplifying assumptions can be avoided by performing full Crystal Plastic Finite Element Methods (Fig. 3.9.2e, Gotoh and Ishise 1978; Bacroixt and Gilorminit 1995) or Fast Fourier Transform Methods (Fig. 3.9.2f, Lebensohn 2001) that take in account the full spatial distribution of grains, and the local variations in both stress and strain state, and allow intra-grain heterogeneity of lattice orientations within grains.

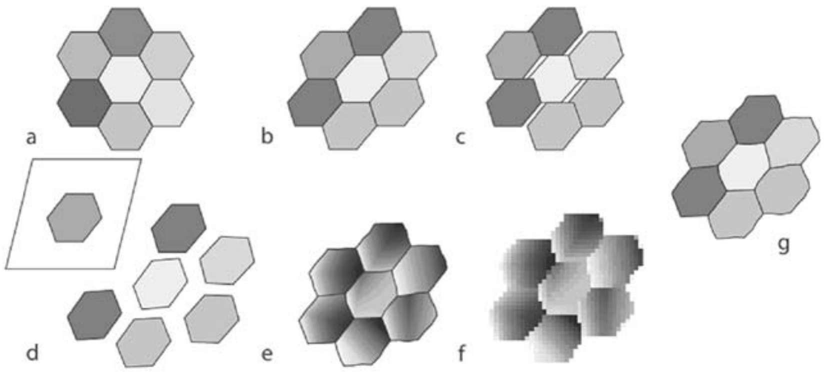


Fig. 3.9.2 Various forms of dislocation glide controlled lattice rotation models. **(a)** Starting grain configuration. Shading reflects lattice orientation. **(b)** Grain configuration resulting from Full Constraint Taylor model: all grains have the same internal strain, and uniform internal lattice orientations. **(c)** Grain configuration resulting from Relaxed Constraint Taylor model: all grains share some strain components, but grain boundary mismatches are allowed in other orientations. All grains have uniform internal lattice orientations. **(d)** Grain configuration resulting from Self-Consistent model: all grains can deform freely, within a uniform matrix reflecting the ensemble of the system. All grains have uniform internal lattice orientations. **(e)** Grain configuration resulting from Crystal Plastic Finite Element model: all grains can deform freely, with smoothly varying stress, strain and lattice orientations within and across grains. **(f)** Grain configuration resulting from Fast Fourier Transform model: all grains, which are discretised into a regular grid, can deform freely, with smoothly varying stress, strain and lattice orientations. In this model grain boundaries are inferred by sharp changes in lattice orientation. **(g)** Grain configuration resulting from hybrid Finite Element-Taylor: all grains can deform freely, with smoothly varying stress, strain and lattice orientations within and across grains, however a single lattice orientation is mapped back to each grain (or subgrain)

3.9.3 Implementation in *Elle*

In this section we have used a Full Constraint Taylor code (using the code described in Lister et al., 1978) coupled to the Finite Element code *Basil* to perform lattice reorientation calculations that allow local variations in stress and strain. However the lattice reorientation calculation is mapped back onto the whole grain or subgrain (Fig. 3.9.2g). These calculations are a simplification of a full Crystal-Plastic Finite Element Method calculation, and have the advantage that they are considerably faster to run.

The coupling of the *Basil* FEM calculation with the Taylor calculation can be carried out at the grain scale, or at the subgrain scale, depending on the input model. In order to simplify the explanation we will assume that each grain starts with a uniform lattice orientation. The calculation cycle then consists of:

1. Assignment of uniform viscosities to all grains for the first calculation step.
2. *Basil* FEM calculation for the deformation of polycrystals. The output of the FEM calculation is the displacement field (homogeneous only for the first time step), which is used to change the positions of grain boundary nodes (and hence grain boundaries).
3. Calculation of average strain tensor for each grain.
4. Calculation of lattice rotations using strict Taylor method (using *elle_tbh*) for each grain separately, using local strain tensor, each grain's unique lattice orientation, but the same CRSS model for each grain. The output of the calculation is the new lattice orientation for each grain, together with the work done to achieve the applied deformation for each grain.
5. The work term for each grain is then used in *elle_viscosity* to define the new viscosity of each grain for the next FEM calculation. In this way grains that are poorly oriented for slip with respect to the available slip systems will deform at a lower rate in the following time step.
6. Cycle through steps 2-5.

In this cycle no attempt is made to directly account for work hardening, or recovery; as these effects can be added using separate processes. Conversely the work term for each grain, as calculated by the TBH code and this could be used as the basis for predicting dislocation densities. If Step 5 is ignored, the calculation reverts to a strict Taylor calculation, albeit with the laborious inclusion of a redundant FEM calculation.

The model inputs required to perform this calculation are:

1. Crystal symmetry and slip system definition for the given mineral (constant for all flynns of that mineral type).
2. CRSS for each slip system (plane + direction) that may be activated which are constant for constant temperature and strain rate experiments (ignoring work hardening, recovery and recrystallisation issues!).
3. The initial lattice orientation of each flynn as 3 Euler angles (unique to each flynn).

There are three main limitations to the calculation as implemented:

1. When linked to *Basil* this model still assumes isotropic mechanical behaviour, when in fact crystals deform anisotropically (for example a natural mica grain can only really deform by sliding on its basal plane under most conditions, hence it cannot generally achieve the homogeneous strains necessary to satisfy the TBH theory).
2. The strains from applying a plane stress to an arbitrarily oriented crystal will not generally be plane strain. In a 2D world we simply ignore the out of plane stresses.
3. TBH theory does not take into account lattice reorientations due to rigid body rotations (for example a rotating but not internally deforming porphyroblast).

3.9.4 Example *Elle* Run

In this example we couple a TBH calculation with the *Basil* FEM code to simulate deformation of a quartz polycrystal (see appendix B, experiment 10, for parameters used), and base the updated viscosities on the work term calculated by the TBH code (Fig. 3.9.3). In this way grains which are able to deform by glide on slip systems with low Critical Resolved Shear Stress will deform more rapidly than grains which only have 'hard' slip systems available.

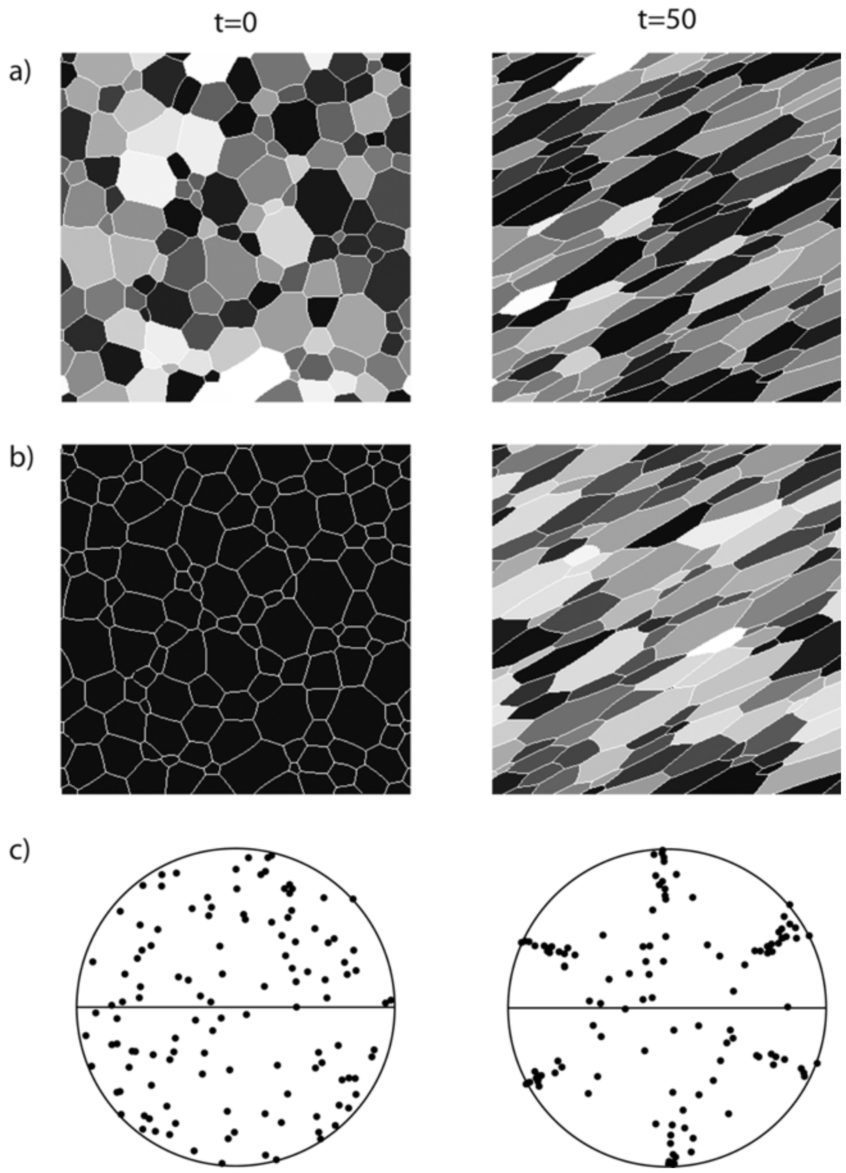


Fig. 3.9.3 Simple coupled TBH-FEM calculation. **(a)** Lattice orientations at *t*=0 and *t*=50, grey scale is a function of only the Alpha Euler angle. Notice that by *t*=50 many of the grains share a similar orientation. **(b)** Instantaneous viscosities at *t*=0 and *t*=50 (bright=high viscosity) **(c)** c-axes stereograms at *t*=0 and *t*=30, the latter equivalent to a shear strain of 1.5. EXPERIMENT 10

3.10 Diffusion creep

John Wheeler and Judy Ford

3.10.1 Phenomenological observation

Diffusion creep is deformation accomplished by diffusion of vacancies, atoms or ions in response to stress. Net flux of material from high- to low-stress interfaces of individual grains causes them to change shape, and macroscopic strain results. This can be thought of as dissolution at high-stress interfaces and precipitation at lower stress interfaces to give overgrowths (metallurgists would not necessarily use the same words). Pressure solution or dissolution-precipitation creep is a term used for diffusion creep when diffusion is mediated by water in pores or H_2O in interfaces (where it may not have the properties of bulk water).

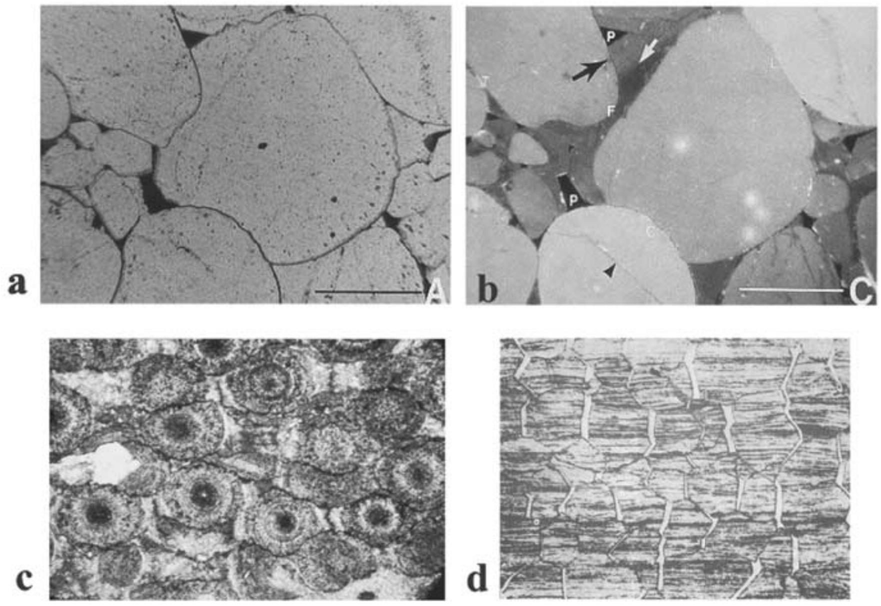


Fig. 3.10.1 Pressure-solved quartz grains in **(a)** Plane-polarised light and **(b)** cathodoluminescence (Barker and Kopp 1991). **(c)** Oolites showing dissolution and overgrowth. **(d)** Denuded zones and overgrowths due to diffusion creep in Mg-0.5%Zr (Squires et al., 1963)

3.10.2 Natural examples

Diffusion creep is not particularly straightforward to identify in resultant microstructures because, for example, a "clean" overgrowth on a clean crystal may not be recognizable, and an interface along which dissolution has occurred may look no different to another. However, when pre-existing marker shapes are available, obvious microstructures result (e.g. Fig. 3.10.1c). Here the spherical shape of an oolite is truncated along two sides. Overgrowths are recognizable as being free of insoluble inclusions. Conversely those inclusions have been concentrated along other boundaries. Overgrowths are easily recognizable around objects of different composition, e.g. pressure shadows around pyrite. Where inclusions are not present, chemical differences may characterize overgrowths (Misch 1970), or microstructures may indicate indentation of one quartz grain into another without any sign of crystal plasticity (Fig. 3.10.1a,b). Overgrowths may be of different mineralogy to the dissolving mineral(s) in which case the deformation mechanism is called "incongruent pressure solution" (Beach 1979; Brodie 1995).

3.10.3 Inferred processes

Overgrowths on certain interfaces, and the increased abundance of insoluble residue particles along others, imply mass transport. Atoms, ions or vacancies must be generated at some interfaces, move by diffusion and then become incorporated into lattices at other interfaces. Diffusion pathways include lattices and the interfaces themselves.

3.10.4 Driving forces, controls and governing equations

Diffusion creep is driven by stress. The current form of diffusion creep theory was first put forward for lattice diffusion in metals (Nabarro 1948). He showed that at interfaces subject to high normal stress (σ_1), there would be a lower concentration of vacancies than at those interfaces with low normal stress (σ_3). Hence vacancies would diffuse from σ_3 to σ_1 interfaces, which in macroscopic terms is equivalent to a flux of atoms from σ_1 to σ_3 interfaces. In either description, dissolution results at σ_1 interfaces and precipitation at σ_3 interfaces. It is important to note that it is not the stress *inside* a grain that drives diffusion; it is only the "boundary conditions" of different vacancy concentration that do this, which, in turn, result from the differences in normal stress on different interfaces. Because interfaces play a key role as sources and sinks of vacancies, the strain rate is grain size dependent, in fact

$$\dot{e} \propto a^{-2}, \tag{3.10.1}$$

where a is grain size. It is salutary to note that Nabarro first proposed this mechanism 15 years before experimental verification was made (Fig. 3.10.1d) (Squires et al., 1963). Coble (1963) proposed a theory where the driving force was similar but diffusion was along interfaces, resulting in:

$$\dot{e} \propto a^{-3}, \tag{3.10.2}$$

but strain rate is linear in stress, as in the lattice diffusion mechanism. In geology, it was long recognised that dissolution could occur so as to allow grains to approach each other, but there were arguments about how this was accomplished. In one model dissolution was proposed to occur just at the grain/grain/water triple junction in a porous rock, but not along the interface itself (Bathurst 1958). Others (de Boer 1977; Weyl 1959) proposed that the high normal stress elevated the concentration of the dissolved material *within* the interface, and that prompted diffusion along the interface. The fundamental controlling equation is (e.g. Eq. 8 in Paterson 1973):

$$\mu = F + \sigma_n V, \tag{3.10.3}$$

where σ_n is the normal stress across the interface, F is the molar Helmholtz free energy of the solid, V is the molar volume of the solid, and is the chemical potential of the solid material in an adjacent "phase" in which it can dissolve. We use the quotation marks because the "phase" is, in fact, the interface region, which may have a rather complex structure involving pockets of water and narrower regions, possibly with H_2O present but with properties different to those of bulk water. This equation can be approximated as

$$\mu = \mu_0 + \sigma_n V, \tag{3.10.4}$$

where μ_0 is the chemical potential at zero pressure. This is the form used by many authors, not always making it clear that it is an approximation. If we postulate the diffusing material is an ideal solution with concentration c then from (3.10.4) we have

$$c = c_0 \exp\left(\frac{\sigma_n V}{RT}\right), \tag{3.10.5}$$

which is very similar to Nabarro's expression for the vacancy fraction (ϕ) at an interface with pressure P:

$$\phi = \phi_0 \exp\left(-\frac{PV}{RT}\right). \qquad (3.10.6)$$

So, although presented quite differently, the fundamental mathematics of diffusion creep is much the same according to materials scientists or geologists.

In many works the mathematical framework has been used in conjunction with some idealized regular geometry to deduce a flow law for diffusion creep. The details of grain-scale behaviour are not necessarily modelled in detail. However models based on simple tessellated microstructures (e.g. rectangular or hexagonal grains) do not capture the rich behaviour of more complex and realistic microstructures.

How, then, is a grain scale model of diffusion creep to be constructed? From now on we will focus on interface diffusion creep. The evolution of general interface networks has been addressed by Hazzledine and Schneibel (1993), though they only address one time increment of evolution. Other models (Ford et al., 2002; Pan and Cocks 1993) simulate long-term evolution. In outline the models share these features:

Along an interface segment, diffusive fluxes are driven by gradients in chemical potential

$$C = -wL\frac{\partial \mu}{\partial x}, \qquad (3.10.7)$$

where C is the current (flux $\times$ interface width) in mol/m/s, w is the notional interface width, L is the Onsager diffusion coefficient and x is position along the boundary.

The growth or dissolution rate is governed by the gradient in C. Imagine a growing boundary: the material added (for example as overgrowths) must be supplied by the diffusive current, so this current varies with position. If r is the precipitation rate in mol/m^2/s then mass conservation dictates

$$r = -\frac{\partial C}{\partial x}. \qquad (3.10.8)$$

Note that r is simply linked to the divergence speed (u) of the two grains

$$r = u/V, \qquad (3.10.9)$$

but now let us assume the grains are rigid (a reasonable approximation) in which case their relative motion must be equivalent to a relative velocity plus a rotation about some reference point. This means that u has to be constant or a linear function of position. In other words incremental overgrowths must be of constant width or wedge-shaped but cannot have more complex shapes because that would imply the grains have deformed internally. Combine Eqs. 3.10.4 and 3.10.7-9 to get

$$u = wLV^2 \frac{\partial^2 \sigma_n}{\partial x^2}. \qquad (3.10.10)$$

Since u is linear in x, σ_n is at most cubic in x. If there is no relative rotation of grains around an interface then u is fixed and σ_n is quadratic in x. If there is no growth or dissolution, σ_n may be constant, but can still be linear in x, in which case there will be a (constant) diffusive flux along the boundary.

Equation 3.10.10 implies that each straight boundary segment will support a stress specified by a maximum of four parameters. Grain movements can be completely specified by two velocities and one angular velocity per grain. Table 3.10.1 (simplified from Ford et al., 2002) indicates how we can build up the system of equations that constrains those unknowns. Some equations relate to the number n_g of grains, some to the number n_{gb} of interfaces, some to the number n_t of triple junctions and some to the number n_s of interface intersections with the edge of the model. The letter in each box indicates the appropriate part of this section. We omit double junctions for simplicity.

There are a number of constraints at triple junctions. Since they have no volume, the net diffusive current (sum of the three C values, with signs adjusted appropriately) into a triple junction must be zero. This gives rise to one constraint per triple junction.

Equation 3.10.8 implies that σ_n must be a smooth function of position, otherwise infinite fluxes will result. This implies that the three σ_n values predicted from the cubic form on each of the three boundaries must, at a triple junction where they meet, be equal. This gives two constraints per triple junction.

Accelerations and angular accelerations must be negligible for each grain. Zero net force gives two constraints per grain and zero net torque an additional one.

The growth speed u at an interface can be expressed in terms of the relative velocity and angular velocity of the two grains (and, being linear, is completely specified by two parameters). However, according to Eq. 3.10.10, those two parameters are also functions of the stress coefficients on the boundary. We thus obtain two more equations per boundary, which we call “growth rate geometric compatibility”.

Any model must have boundary conditions: for example, if the system is chemically closed we may dictate that the diffusive current is zero at each place where an interface intersects the edge of the model, giving n_s constraints.

Summing up it appears that the number of unknowns does not match the number of equations. However, network topology tells us that

$$3n_t+n_s = 2n_{gb}, \tag{3.10.11}$$

from which it follows that the number of equations equals number of unknowns and the system has a unique solution.

This gives us the stresses and velocities at a given time, but not the positions of the new interfaces. These are non-unique because there is no physics (in the models cited above) that tells us where the new boundary should be. It seems reasonable to draw it along the middle of the overgrowth or dissolution zone. This gives rise to a geometric problem at triple junctions – new boundaries do not generally meet at a point, a problem recognised by Pan and Cocks (1993) and Ford et al. (2002), though dealt with in slightly different ways by those authors.

Table 3.10.1. Number of equations versus number of unknowns in interface diffusion creep model.

Unknowns		Equations	
Stress coefficients (d)	$4n_{gb}$	Mass conservation at triple junctions	n_t
Grain velocities (d)	$3n_g$	Stress continuity at triple junctions	$2n_t$
		Force and torque balance	$3n_g$
		Growth rate geometric compatibility	$2n_{gb}$
		Boundary conditions	n_s
TOTAL	$4n_{gb}+3n_g$	TOTAL	$3n_t+3n_g+2n_{gb}+n_s$

3.10.5 Possible simulation techniques

The set of equations described above can be cast as

$$\mathbf{A}\boldsymbol{x} = \boldsymbol{b}, \tag{3.10.12}$$

where $\boldsymbol{x}$ is the vector of unknowns (stress coefficients and velocities), $\boldsymbol{b}$ is a vector of knowns (boundary condition forces, etc.) and $\mathbf{A}$ is a matrix expressing the structure of the equations. The system is then solved by inverting the (quite large) matrix $\mathbf{A}$. This is what is done in the *DiffForm* program based on the work of Ford et al. (2002). The governing equations can also be expressed in the form of a variational principle (Cocks et al., 1999; Pan and Cocks 1993) which has advantages in setting up the numerical solution although ultimately manipulation of large matrices is again required for numerical solution (Eq. 26 in Pan and Cocks 1993).

3.10.6 Implementation in *DiffForm*

At each time step the matrix **A** is constructed from the governing equations and the network geometry. Its structure is somewhat different depending on the types of boundary conditions (which include constant velocity, constant stress and zero flux in various combinations). The matrix is then inverted. Refinements of the numerical technique lead to considerable improvements in speed (Ford et al., 2004). The new interfaces are drawn using a best-fit solution to the triple junction problem mentioned above. The process is then repeated for the next time step. Neighbour swapping is a natural part of diffusion creep, and this is implemented. *DiffForm*, written in C, produces text files as output. A separate package, *DiffView* (written in *Matlab*), displays output as pictures in a variety of formats. At present this code has not been integrated with the *Elle* system.

3.10.7 Simple example and description of input parameters

Figure 3.10.2a shows a model interface network with a large grain in the middle. Figures 3.10.2b-c show the network after a given time interval and then the same time increment later. Boundary conditions are: constant velocity at top; sides constrained to remain vertical; zero diffusive flux into or out of model at boundaries. Figures 3.10.2e-f show the outlines of the original grains (which we refer to as "ghosts") displaced in accordance with actual grain movements. Ghosts appear to overlap where there has been dissolution. Apparent gaps between ghosts are sites where overgrowth has occurred (in the actual microstructure porosity is zero at all times). In such situations the ghost outline might be visible, e.g. between clouded original and clean overgrowth mineral. Figures 3.10.2e-f show significant grain rotations, especially the two just above the large grain. Such rotations are not at all obvious in the interface maps (Fig. 3.10.2b-c), and they may be far from obvious in actual microstructures. Complex overgrowth/dissolution patterns are likely, but *DiffForm* cannot currently display overgrowth patterns. Figure 3.10.2d shows a 3D diagram of the stress system. The x- and y-axes represent the actual microstructure, and the z-axis is the normal component of stress. Since this is only relevant along the interfaces, it is displaced as a shaded “fence” along each interface. Commonly the stresses show parabolic patterns, in accordance with a constant dissolution/overgrowth rate along an interface segment. However the stress patterns around the rotating grains are cubic, also as required by Eq. 3.10.10. The fences are shaded darker according to dissolution rate, so many have constant shade, but the rotating interfaces show varying shades. The fences running “east-west” are under positive stress.

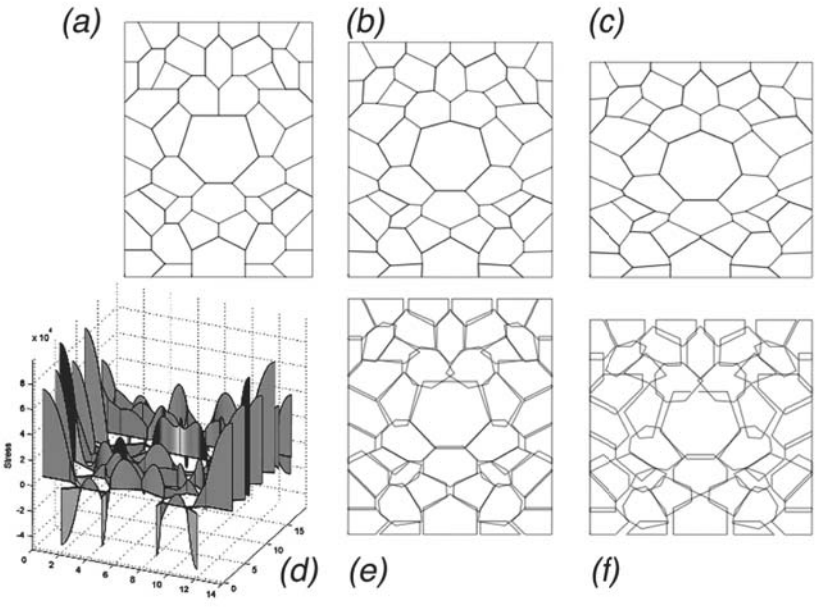


Fig. 3.10.2 Simulation of interface diffusion creep in a microstructure with grains of irregular shapes. **(a-c)** Three stages in the vertical shortening of a grain aggregate. **(d)** “Fence” diagram showing the normal stresses on the interfaces. **(e-f)** Outlines of the original grain shapes to illustrate sites of dissolution (overlaps) and precipitation (gaps)

Symbols used in Chap. 3.10

$\mathbf{A}$	Matrix to express structure of equations [various units]
a	Grain size [m]
$\boldsymbol{b}$	Vector of knowns (boundary condition forces, etc.)
c, c_0	Concentration in solution; concentration at zero normal stress [mol m^{-3}]
C	Current (flux × interface width) [mol m^{-1} s^{-1}]
$\dot{e}$	Strain rate [s^{-1}]
F	Molar Helmholtz free energy of the solid [J mol^{-1}]
ϕ, ϕ_0	Vacancy fraction; vacancy fraction at zero pressure []
L	Onsager diffusion coefficient [mol^2 J^{-1} m^{-1} s^{-1}]
μ, μ_0	Chemical potential of a solid in an adjacent "phase" in which it can dissolve; chemical potential at zero pressure [J mol^{-1}]
n_g, n_{gb},	Number of grains and of of interfaces []
n_t, n_s	Number of of triple junctions and of interface intersections with the edge of the model []
r	Precipitation rate [mol m^{-2} s^{-1}]
σ_n	Normal stress across an interface [Pa]
σ, σ_3	Highest and lowest normal stress, respectively [Pa]
u	Divergence speed of two grains [m s^{-1}]
V	Molar volume of the solid [mol m^{-3}]
w	Notional interface width [m]
$\boldsymbol{x}$	Vector of unknowns (stress coefficients and velocities)

3.11 Fracturing

Daniel Koehn

3.11.1 Phenomenological observations

Fractures are discontinuous surfaces that form in a material while it breaks and looses its cohesion. Fractures can occur on all scales ranging from micro-cracks with sizes of a few nanometres up to kilometres. Large-scale fractures in geology are termed joints, or faults if displacement has taken place along the fracture plane. Fractures may open and leave voids where new material can precipitate or they may show displacement. They are therefore also classified into extension and shear fractures. Micro-cracks are generally classified into mode I, mode II and mode III fractures (Pollard and Segall 1987; Scholz 2002) (Fig. 3.11.1). Only Mode I and II fractures will be considered here since these are the fracture modes possible in two dimensions. Mode I fractures are extensional and open perpendicular to a maximum tensile stress. Mode II fractures open with an angle towards the maximum compressive stress and accommodate strain by shear displacement.

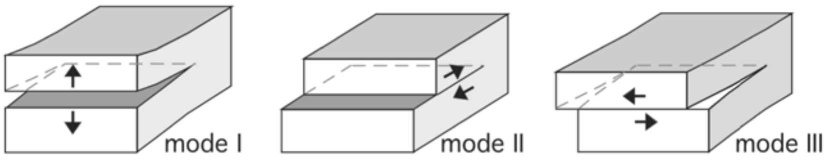


Fig. 3.11.1. The three failure modes I, II and III

Fracture propagation is a highly non-linear and complex process. The initiation of a fracture causes stress concentrations at the fracture tip, which then in turn will provoke the failure of material so that the fracture will propagate. This is especially critical for mode I or extension fractures (Griffith 1920; Jaeger and Cook 1976). The stress at the fracture tip increases with the length of the fracture so that fracture propagation and final failure of a material under tensile stress may be a runaway process. Fractures that propagate by themselves may propagate as fast as the speed of sound. They are therefore developing instantaneously within a geological context and do not consume time. Fractures that propagate very fast may become unstable once they travel as fast as seismic waves. In that case the fracture tip may split, which slows down further propagation.

The properties of the host material are very important for the propagation of fractures. Fracturing of material is mainly thought to be a plastic process. This means that the fracture process itself is time-independent but it needs a critical yield stress in order to be initiated. A material with uniform properties (such as elastic constants and failure strength) will break in a brittle manner so that most fractures develop instantaneously once a yield stress is reached and the whole material will loose its strength. A material with non-uniform properties may, however, break in a ductile manner where small fractures develop once a yield stress is reached but the material as a whole does not fail. An increase in stress is needed to deform the material further.

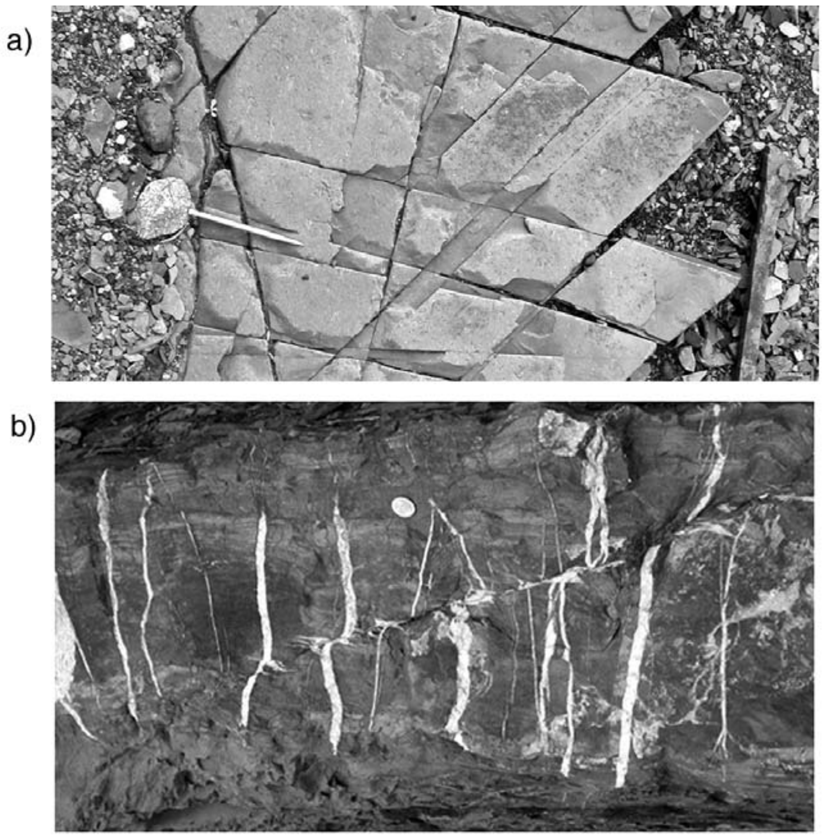


Fig. 3.11.2 (a) joints in slate, pencil for scale **(b)** veins and a shear fracture (or fault), coin for scale

3.11.2 Natural examples

Brittle deformation that involves fracturing of rocks represents one of the main deformation mechanisms in the upper crust of the Earth. Fractures may also be found much deeper within the Earth depending on strain rates of deformation, fluid pressure and local volume changes. A wide range of evidence for fracture processes can be found in natural systems (Fig. 3.11.2).

Near the surface of the Earth fractures are expressed in the form of joints and faults. Joints are mainly thin and relatively long discontinuities (Fig. 3.11.2a) without a displacement in contrast to faults. Deeper within the Earth, fractures are often healed. Extension fractures may then be preserved in the form of veins or dykes (Fig. 3.11.2b).

3.11.3 Inferred processes

Fractures develop because of local tensile or shear stresses. These stresses may be induced by external boundary conditions such as deformation or local changes due to volume expansion or contraction as well as fluid pressure gradients. Nucleation of fractures depends on the local boundary conditions and on the material properties of a rock sample. The propagation of fractures also depends strongly on the boundary conditions. External deformation may lead to a runaway process of fracture propagation especially when the material is under tension. Internal boundary conditions such as the expansion of material will drive the propagation of sub-critical fractures that grow while the material is expanding or contracting. Hydrofractures that develop due to high fluid pressure gradients will also tend to grow at least partly sub-critical because fluid pressures often drain once fractures develop due to an increase in void volume or an increase in permeability (Flekkøy et al., 2002).

3.11.4 Driving forces, controls and governing equations

The driving force for fracture propagation is a local tensile or shear stress that exceeds a critical value. The local fracturing process is normally time independent in a geological context because it takes place very quickly. However, fracturing on a large scale may become time dependent if the system is very non-uniform, so that local fractures develop at a range of critical stresses. Fracturing is also partly time-dependent when the process that leads to the critical stress is time-dependent and fracturing releases local stresses.

One of the basic theories for fracture development was defined by Griffith (Griffith 1920; Jaeger and Cock 1976). Stress concentrations at the tips of an elliptical hole with a length *2l* and a radius of curvature ρ are

$$\sigma_{loc} = \sigma_2 \sqrt{\frac{l}{\rho}}, \tag{3.11.1}$$

where σ_{loc} is the local stress at the tip and σ the external stress (for example an applied tension). If the curvature of the ellipse in Eq. 3.11.1 becomes very small, which it should for a thin crack, the stress will go towards infinity. Griffith used an energy balance between elastic energy due to the stress concentrations and surface energies at the crack tip, which also become very high if the curvature is small. This results in a critical micro-crack size for a material. These so called Griffith-cracks are thought to exist in most materials and are used to explain their low tensile strength. Griffith theory leads to the following two criteria for brittle failure:

$$\sigma_1 = T_0 \text{ , if } (3\sigma_1 + \sigma_2) > 0, \text{ and} \tag{3.11.2a}$$

$$(\sigma_1 - \sigma_2)^2 + 8T_0(\sigma_1 + \sigma_2) = 0 \text{ , if } (3\sigma_1 + \sigma_2) < 0, \tag{3.11.2b}$$

where T_0 is the tensile strength, σ_1 (principal normal stress)is larger than σ_2 and compressive stress is negative. The first criterion explains the development of mode I cracks based on a local tensile stress (Eq. 3.11.2a). The second criterion marks the development of shear fractures (mode II) as a function of the differential stress, the mean stress and a tensile strength (Eq. 3.11.2b).

A second concept of brittle failure is the Navier-Coulomb shear failure criterion (Price and Cosgrove 1990) that is based on a friction coefficient and a cohesion:

$$\tau = C_0 - \mu\sigma_n, \tag{3.11.3}$$

where τ is the critical shear stress, C_0 the cohesion, σ_n the normal stress (compression again negative) and μ the friction coefficient. Shear strength increases with decreasing normal stress expressed by the friction coefficient. In the tensile failure regime the Navier-Coulomb criteria results in unrealistically high breaking-strengths (because the tensile strength is as high as the cohesion). Therefore a combination of the Griffith and the Navier-Coulomb criteria is thought to represent a realistic failure criterion spanning both tensile and compressive regimes of failure. The combined Navier-Coulomb Griffith failure curve predicts the angle that a fracture will have with respect to the principal stresses.

Tensile extension fractures will develop parallel to the main compressive stress. Shear fractures will develop with an angle of less than 45° with respect to the main compressive stress. So called hybrid shear fractures may develop in a small regime between extension and shear fractures. Shear fractures will always have the possibility to develop to either side of the main compressive stress direction and therefore form conjugate sets.

3.11.5 Possible and actual simulation techniques

Brittle deformation can be modelled using continuum models, discrete element models and Cellular Automata. In the continuum applications different formulations can be used. One is the use of the Navier-Coulomb failure criteria, which gives the orientation and location of faults. These models are also referred to as Micro-plane models (Kuhl et al., 2001). Once a fault develops, elements in the continuum model are said to have failed and their properties are modified accordingly. Another approach is a Lagrangian Particle Finite Element Method where the position of a finite number of material points is traced within, for example, a Finite Element mesh (Mühlhaus et al., 2001). In this case large-scale flow and high deformation can be achieved without having to re-mesh.

The widest range of research using numerical simulations on fracture processes has been performed using discrete element models. Fracturing itself is a process that produces discrete materials so that continuum theory breaks down, at least partly. Therefore it is useful to perform numerical simulations with materials that can become discontinuous. In these models, particles are connected with each other by springs, with only normal forces or angular and normal forces, or by elastic beams that can bend. In two dimensions a hexagonal arrangement of particles connected with normal springs produces linear elasticity properties on a larger scale. If the lattice is not hexagonal or if it is a three-dimensional lattice, a normal force model only produces linear elasticity when more than the nearest neighbours are connected. Therefore a square lattice needs connections to four direct neighbours as well as to the next four neighbours in order to show no large-scale anisotropy. The same applies for models with particles that have no uniform size. In addition to normal force models, beam models or models with angular springs produce linear elasticity for different lattices and variations in particle sizes. Failure is then represented by the breaking of beams or springs so that the tensile strength is locally lost, and only the repulsive interactions between particles remain. In order to model granular materials particles can also be modelled with contact forces that are only repulsive or represent friction (Vermeer et al., 2001).

Cellular Automata can also be used to model fracture processes. An example is the use of Diffusion-Limited Aggregation (DLA) models that can be used to model dielectric breakdown, which has scaling similarities to fractures (Meakin 1998).

Which model one uses depends strongly on the scale of observation and the aims.

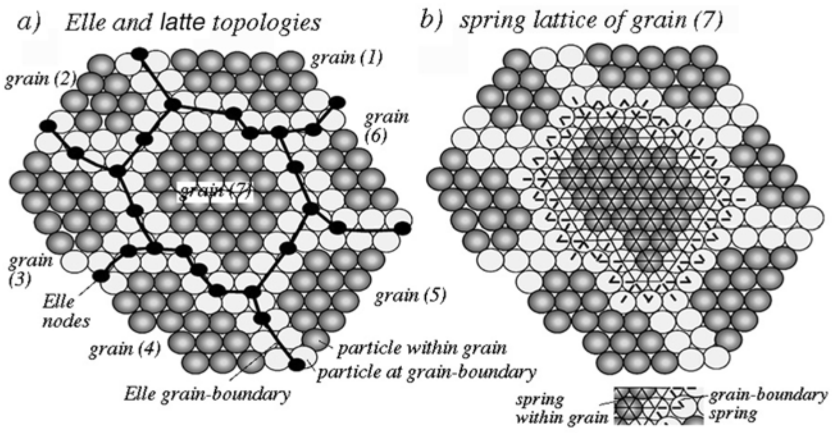


Fig. 3.11.3 Merging of the *Elle* topology and the Lattice-Spring model with a triangular lattice. **(a)** Grain number seven is surrounded by six neighbouring grains. Dark grey particles lie totally within grains whereas the light grey particles lie along grain boundaries. *Elle* flynn boundaries and bnodes are drawn in black. **(b)** spring lattice of grain number 7. Particles are connected by six springs with their neighbours. Broken lines are springs across grain boundaries whereas through-going lines lie within the grain. Grain boundary springs and internal springs of different grains may have different elastic properties

3.11.6 Implementation in *Elle*

Currently a Lattice-Spring model called *Latte* is implemented in *Elle*. It is a normal force model with a hexagonal grid, based on the work of Malthe-Sørenssen et al. (1998b). Linear elastic behaviour and fracturing can be simulated with the full description of the strain and stress field. The initial microstructure can be read from an *Elle* file and the grain boundaries can be translated into the particle code. Particles that lie within one grain can then have different properties than those of other grains and grain boundaries may also be different (Fig. 3.11.3a). Studies with these kinds of particle models have been able to reproduce the scaling behaviour and statistics of experimental and natural fracture patterns (Walmann et al., 1996; Malthe-Sørenssen et al., 1998a,b, 1999) and more complex systems (Jamtveit et al., 2000; Flekkøy et al., 2002; Koehn et al., 2003a,b, 2005; Koehn and Arnold 2003).

In the model particles are connected with their direct neighbours via linear elastic springs (Fig. 3.11.3b). The force acting on a particle from a neighbour is proportional to the extension or compression of a connecting spring. Compressive stresses in the model are defined as being negative and tensile stresses positive. An over-relaxation algorithm is used to solve the equilibrium configuration of particles. A relaxation threshold determines the accuracy of the solution. Particles are in an equilibrium position once all forces acting on particle centres cancel out. If a particle is not in equilibrium it is moved towards equilibrium depending on the resulting force on the particle. In order to attain an effective solution particles are moved an over-relaxation step beyond their equilibrium position (Allen 1954). Note that a hexagonal spring lattice with only normal force interactions always has a Poisson ratio of 0.3333.

Initially the *Elle* deformation box is filled with particles. Particles along boundaries are defined to be stationary perpendicular to boundaries, but they may move parallel to a boundary. In order to apply a deformation, walls are moved inwards or outwards respectively. Then a relaxation algorithm resolves the new equilibrium position of particles within the box. Once the equilibrium defined by the relaxation threshold is reached the tensile stress felt by all springs are checked. If the local tensile stress of a spring exceeds its tensile strength, the spring may break. Once all springs are checked for breaking potential, the spring that is most likely to break (the one that most oversteps its tensile strength) will do so and a new relaxation cycle will start. This means that springs only break one at a time. Once all springs are stable and are not breaking a new deformation step can be applied. Fracturing in this case is highly non-linear. The whole lattice may fail at once after a deformation step. Note that the model's boundary particles are not allowed to fail in the current version. Broken particles still retain a repulsive force so that they build up forces once they are pushed into each other. Large-scale deformation of the lattice is achieved not only by moving outer walls but also by moving all particles assuming a perfectly homogeneous deformation of the lattice. After this deformation average the relaxation starts. The deformation average speeds up the relaxation and gets rid of artificial gradients in the model.

Latte is build up by two main classes, the particle and the lattice class (lattice.cc, lattice.h, particle.cc and particle.h). More classes are used for the phase transformations described in Chaps. 3.12 and 3.13 (base_phase.cc, base_phase.h, phase_lattice.cc, phase_lattice.h, min_trans_lattice.cc, min_trans_lattice.h, heat_lattice.cc, heat_lattice.h) that inherit the lattice class. The user does not have to be concerned with these classes directly but calls initialization and run functions in the lattice class (for fracturing) from a normal *Elle* process file (experiment.cc) that is used like a script. The explanation of how to use the script and the different functions is described in detail in Appendix G. Functions are basically separated into initialization functions that give different flynns, single particles or defined layers in the model different properties and run functions that define the proposed deformation or internal changes and dump statistics to plot, for example, stress-strain curves. A few examples where values can be changed from the interface are given below and in the interface.

3.11.7 Example *Elle* Runs

Example 1: fracturing

This example illustrates fractures that develop in a rock that is subject to a small amount of uniaxial shortening followed by a pure shear deformation. The initial microstructure of the specimen consists of grains that have different Young's moduli drawn from a Gaussian distribution. Spring constants of all particles also have a distribution of strengths. Grain boundaries are assumed to fracture more easily and their fracture toughness is on average 50% of the fracture toughness of intra-granular bonds. An example with a 400 particle wide lattice is shown in Fig. 3.11.4. Progressive deformation leads to the development of small local fractures whose location is partly dependent on the initial noise in the system. These then develop into larger-scale conjugate shear fractures. Figure 3.11.4a and c show particles that have broken bonds as dark lines. The corresponding stress fields are shown in Fig. 3.11.4b and d where dark is relatively low mean stress and light relatively high stress. Fractures release the stress and grow as conjugate shear fractures with extensional parts. They nucleate in Fig. 3.11.4a as a function of the differential and mean stress between grains and the noise on breaking thresholds. The underlying microstructure of *Elle* is shown in Fig. 3.11.4b. The initial small-scale fractures grow in length and connect to become larger fracture sets in Fig. 3.11.4c.

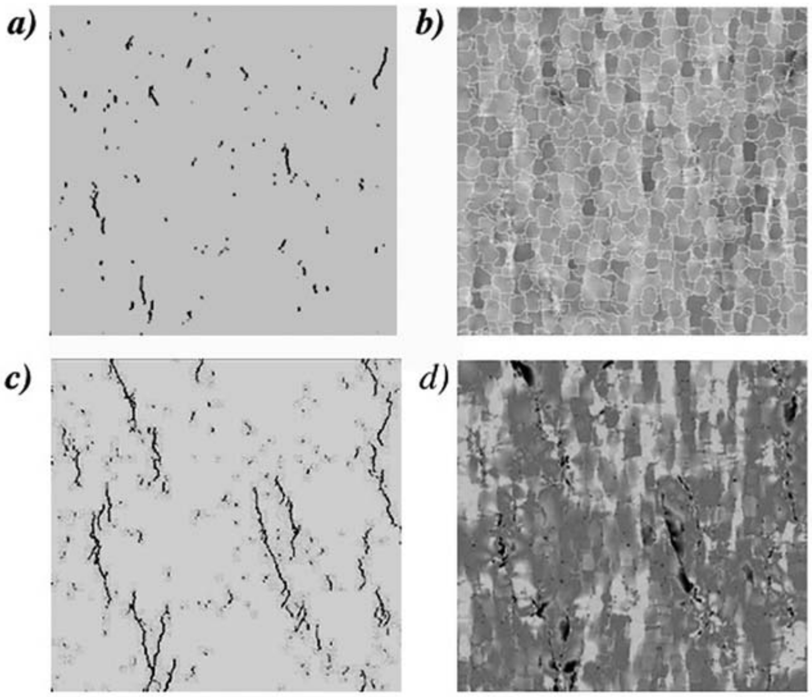


Fig. 3.11.4 Fracture patterns during pure shear deformation of a polycrystalline solid. **(a)** and **(c)** show fracture patterns (dark are particles with broken bonds) and **(b)** and **(d)** show the mean stress corresponding to a) and c). The grain boundaries are shown in b) Grains have initially different Young's moduli picked from a Gaussian distribution. Black is relatively low and white relatively high stress. The original lattice is 400 particles wide. a) and b) are time step 10 whereas c) and d) are time step 20. EXPERIMENT 4

Example 2: fracture boudinage

This example illustrates the development of fracture boudinage in a layer with a higher Young's modulus than the surrounding matrix. The underlying lattice has no clusters of particles that define grains. All particles between given maximum and minimum y-coordinates are defined as part of the competent layer and have different assigned elastic properties than the matrix particles. The springs have a distribution of breaking strengths drawn from a Gaussian distribution with a mean of 2.0 (corresponding to a mean Young's modulus of for example 20 GPa in a scaled model) and a deviation of 1.2. The applied deformation is pure shear with vertical shortening and horizontal extension.

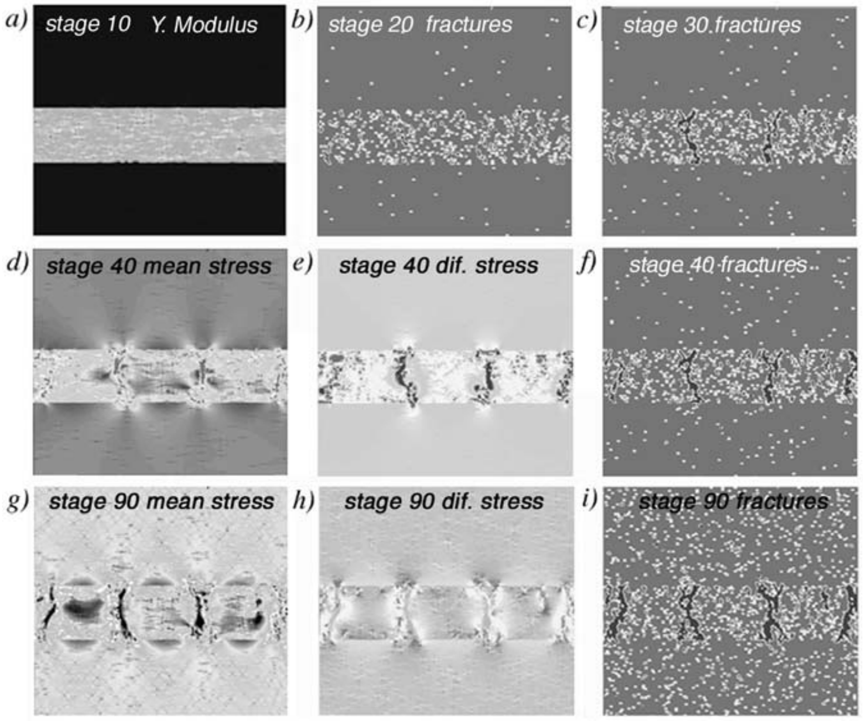


Fig. 3.11.5 Fracture boudinage in a competent layer embedded in a less competent matrix. The competent layer has a higher Young's modulus than the matrix (factor of 10). Breaking strengths of bonds have a Gaussian variation with a mean of 2.0 and a deviation of 1.2 **(a)** Stage 10 shows the initial configuration with the competent layer in the middle. First fractures develop in the competent layer. **(b)** Stage 20 shows the successive development of fractures in the competent layer, minor fractures in the matrix (light grey shows small fractures). **(c)** Stage 30 shows the amalgamation of fractures in the competent layer into two distinct mode I fractures (black shows fracture clusters, light greys small fractures). **(d-f)** show stage 40. **(d)** Mean stress in the competent layer and matrix. Two well-developed fractures in the competent layer are opening and produce stress in the weak matrix. Stress field between the two fractures shows "bone-type" geometry. **(e)** Differential stress in the competent layer. Stresses are relaxed around the large fractures. At the fracture tips stresses are high in the matrix. **(f)** Corresponding fracture pattern. Two distinct fractures developed in the middle, two additional fractures start to develop at the layer-matrix interface. **(g-i)** show stage 90, four well-developed fractures are present in the competent layer with a well-defined spacing. **(g)** Mean stress, **(h)** differential stress and **(i)** fracture pattern. EXPERIMENT 11

Figure 3.11.5 shows the progressive development of the fracture pattern and the differential and mean stress at different stages. The initial geometry of the setup is shown in Fig. 3.11.5a where the brighter areas in the competent layer represent stresses around small fractures. The relatively wide distribution of breaking strengths leads to the development of numerous small-scale fractures that later on develop into larger macroscopic fractures. Successive fracture patterns are shown in Fig. 3.11.5b,c,f and i where small fractures are light grey and large fractures black. In Fig. 3.11.5c two distinct macroscopic fractures developed out of the initial fracture nuclei. In Fig. 3.11.5i four distinct fractures developed and show a regular final spacing. The mean stress corresponding to Fig. 3.11.5f and i are shown in Fig. 3.11.5d and g respectively and the differential stress in Fig. 3.11.5e and h. Darker areas in the stress field represents low stress and light colour high stress. The most instructive stress field is represented by the mean stress in Fig. 3.11.5g. The stress is low between two neighbouring fractures in the middle of the competent layer. In contrast, at the interface between this layer and the matrix, the stress is still high due to shear forces that are transmitted from the adjacent matrix. This will lead to progressive development of more fractures at a higher strain. These additional fractures will nucleate at the layer-matrix interface, where stresses are high. The stress in the adjacent matrix is highest at the fracture tips of the fractures in the competent layer where extensional strain in the competent layer is concentrated and fractures are slowly opening.

Symbols used in Chap. 3.11

C_0	Cohesion [Pa]
l	Half length of an elliptical hole [m]
T_0	Tensile strength [Pa]
μ	Friction coefficient
ρ	Radius of curvature of the tip of an elliptical hole [m]
σ_{loc}	Local stress [Pa]
σ_1, σ_2	Maximum and minimum principal stress, respectively [Pa, with compressive stress negative]
σ_n	Normal stress [Pa]
τ	Critical shear stress [Pa]

3.12 Fluid solid reactions

Daniel Koehn

3.12.1 Phenomenological observations

Reactions between solids and fluids are very common in the Earth's crust. This chapter is mainly concerned with the reaction between a single-phase solid and a pure fluid that only contains solvents of that solid. Therefore we are mainly dealing with structures that develop during the so-called dissolution-precipitation creep or pervasive pressure solution transfer (Paterson 1973; Rutter 1976; Revil 2001). During dissolution-precipitation creep material dissolves at regions of high stresses, is transported along the fluid and precipitates at sites of low stresses. These driving forces will lead to a viscous or plastic deformation of the solid and the grains will become shorter in the direction of largest compressive stresses (dissolution) and elongate in areas of low stresses (precipitation). This scenario is described in detail in Chap. 3.10. In this chapter we are mainly dealing with small-scale structures and large scale roughening during dissolution-precipitation processes.

Dissolution-precipitation creep produces a range of typical structures at various scales. At the small scale (grain-size and smaller) two different reactive sites exist along grains, the confined grain boundaries and the free surfaces that are open to pore space. Three different models exist for the structure of grain boundaries during dissolution-precipitation creep ranging from a thin-film model where the grain boundary is flat and has a continuous fluid phase between grain, to an island channel model where solid-solid contacts make up islands to a grain boundary model that is more or less a combination of the first two (den Brok 1998; Alcantar et al., 2003). In the grain-boundary model a thin-film exists in between islands and channels develop in areas where small micro-cracks run into the crystals. Recent experiments and models show however that solid-solid contacts are relatively unstable so that the island-channel model develops into a grain-boundary model. The grain-boundary model may then become flat due to surface energy effects and develop into a thin film model. Impurities of crystals including micro-cracks will however induce new roughness so that a thin film model can develop into a grain-boundary model (Koehn et al., 2006). Therefore a characteristic more or less rough grain-boundary structure will probably exist depending on time scales and impurities of the involved crystals.

Large-scale structures during dissolution-precipitation creep are so called stylolites that are rough surfaces that are up to several meters long. They typically show a roughness on different scales.

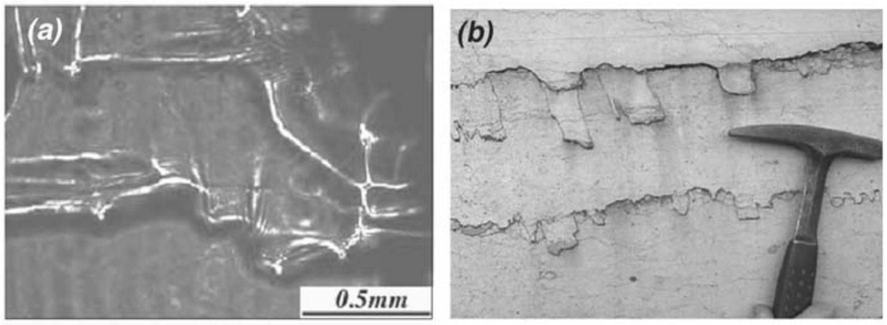


Fig. 3.12.1 Two natural examples of rough structures that develop during solid-fluid interactions. **(a)** Dissolution grooves, looking onto a liquid-crystal surface of a stressed crystal. The crystal surface is oriented parallel to the main, vertical compressive stress. **(b)** Rough surface of a stylolite, with roughness on multiple scales. The stylolite surface is oriented perpendicular to the main, vertical compressive stress. Courtesy of Francois Renard

3.12.2 Natural examples

Figure 3.12.1 shows two 'natural' examples of structures that develop during dissolution-precipitation creep. Figure 1a shows grooves that develop on free surfaces of stressed salt crystals. These grooves start with a well-defined characteristic wavelength. However, with time the wavelength increases and finally flat crystal surface remains (Koehn et al., 2004). Figure 3.12.1b shows a stylolite with a roughness that has wavelengths on several scales. Stylolites are thought to be self-affine so that they have characteristic scaling properties that are independent of the size of the system within a certain range of scales (Schmittbuhl et al., 2004; Renard et al., 2004). They have self-affine and not self-similar properties since they scale differently in the direction along the stylolite versus the direction perpendicular to it.

3.12.3 Inferred processes

Dissolution-precipitation creep is at least governed by three processes: the dissolution of material, the transport of material and the precipitation of material. Dissolution and precipitation of material may be controlled by similar rate laws. However this is not always the case. Transport of material may be by diffusion through a fluid phase or by advection with a moving fluid. As long as the system is not flushed with new fluid, transport will probably be via diffusion. Diffusion and reaction equations do not have to have the same scaling properties with time. Diffusion is often assumed to be the limiting process when the system size increases, since diffusion of material scales with the square root of time whereas reactions often scale linearly with time.

3.12.4 Driving forces, controls and governing equations

The driving forces for dissolution-precipitation creep depend on the processes involved. Diffusion is driven by chemical gradients within the fluid. Reactions are driven by the concentration of dissolved matter within the fluid, the surface and elastic energy of the solid and the normal stress on the fluid-solid interface. We will not deal with diffusion in this section, only with reactions. Diffusion is described in Chaps. 3.2, 3.10 and 3.13.

The chemical potential of a non-hydrostatically stressed solid in contact with a fluid can be described by a Gibbs' type equation of the form (Gibbs 1906; Paterson 1973)

$$\Delta\mu = \Delta H_s + V_s \Delta\sigma_n, \tag{3.12.1}$$

where $\Delta\mu$ is the change in the chemical potential, ΔH_S is the change in Helmholtz free energy of the solid, V_s is the molecular volume of the solid and $\Delta\sigma_n$ the change in the normal stress on the surface. The Helmholtz free energy contains the elastic and the surface energy. The effect of the normal stress is about three orders of magnitude larger than the effect of the elastic energy. The surface energy scales with the size of the system so that it becomes important when the system size is small. Dissolution of material in a grain boundary that has a thin-film structure will be controlled by the normal stress of Eq. 3.12.1 (see also Chap. 3.10). The elastic and surface energies become important at free crystal surfaces, at rough surfaces or when no fluid pressure and thus no normal stress gradients exist along the surface.

The chemical potential (Eq. 3.12.1) is related to the equilibrium constant (K_{eq}) of a reaction by the following expression

$$\Delta\mu = RT\ln\left(\frac{K_{eq}}{K_0}\right), \tag{3.12.2}$$

where T is the temperature, R is the universal gas constant and K_0 is the equilibrium constant for the unstressed solid (Renard et al., 1999). Eqs. 3.12.1 and 3.12.2 lead to

$$K_{eq} = K_0 \exp\left(\frac{\Delta\psi_s}{RT}\right)\exp\left(\frac{V_s\Delta\sigma_n}{RT}\right). \tag{3.12.3}$$

The equilibrium constant can be related to the concentration of dissolved matter in the solution by the expression (Alkattan et al., 1997)

$$K_{eq} = \left(a_{NaCl}c_{NaCl}\right)^2, \tag{3.12.4}$$

for NaCl, where a_{NaCl} is the activity coefficient of sodium chloride in the solution, which is dependent on temperature and c_{NaCl} is the solubility of sodium chloride at a given temperature. For quartz, Eq. 3.12.4 reads

$$K_{eq} = a_{quartz}c_{quartz}, \tag{3.12.5}$$

because there is only one species in the fluid in contrast to two ions for NaCl. The dissolution rate can be determined by a general first order rate law from Transition State Theory (TST, Lasaga 1998)

$$D_r = k_r V_s\left(1 - \frac{c_A}{c_A^{eq}}\right), \tag{3.12.6}$$

where D_r is the rate at which the surface will dissolve (or grow if growth is governed by the same linear rate law), k_r is a rate constant depending on temperature (can also be expressed by an Arrhenius law), c_A is the concentration of solute A in the fluid and c_A^{eq} the equilibrium concentration of solute A in the fluid. Combining Eqs. 3.12.3-4 and 3.12.6 leads to

$$D_r = k_r V_s\left(1 - \left(\frac{c_A}{c_A^i \exp\left(\left(\Delta\sigma_n V_s + \Delta H_s\right)\big/ RT\right)}\right)^{\frac{1}{2}}\right) \tag{3.12.7}$$

for NaCl where c_A^i is the initial equilibrium concentration of the fluid at a given temperature in an unstressed system. Eq. 3.12.7 also holds for quartz if the square root of the second parenthesis is taken out of the equation (see Eq. 3.12.5 in comparison with Eq. 3.12.4). Equilibrium is reached when D_r becomes zero, which happens when the term in the second parenthesis becomes one.

3.12.5 Possible and actual simulation techniques

Dissolution-precipitation creep has been modelled using a number of different continuum approaches (e.g. Ch. 3.10; Renard et al., 1999). These approaches use continuum formulations. Structuring of solid fluid interfaces at free surfaces and even in confined films has been studied recently using linear stability analysis as well as Finite Element approaches and Phase Fields (Yang and Srolovitz 1993; Norris and Vemula 1998; Goussoub and Leroy 2001; Kassner et al., 2001). Groove development in solid liquid systems and other interfaces is a common problem in Physics and Material Sciences. This phenomenon is often called the ATG-instability (Asaro-Tiller-Grinfeld) if it develops so-called anti-cracks and is important in thin-film overgrowths.

Structuring of solid-liquid interfaces as well as the development of stylolites can also be studied using Cellular Automata or discrete element models.

3.12.6 Implementation in *Elle*

In *Elle* the discrete element code *Latte* described in Chap. 3.11 is used for solid-liquid reactions. *Latte* is a Lattice-Spring code where discrete elements are connected with each other via linear elastic springs. In the Lattice-Spring code rows of particles can be defined to be in contact with a fluid, these particle then act as the fluid-solid interface. In order to account for the fluid pressure on the solid a force is added to interface particles. This force is oriented perpendicular to the fluid-solid interface and can change if the fluid pressure changes. In order to calculate the reaction rates of particles along the interface according to Eq. 3.12.7 we need the normal stress at the interface, the concentration of dissolved material in the fluid at a given time step and the Helmholtz free energy. Helmholtz free energy is a function of the elastic energy of single particles and the surface energy along the interface. The elastic energy (E_{el}) can be calculated from the strain tensor (ε_{ij}) following Landau and Lifshitz (1959).

$$E_{el} = V_s \left[\frac{1}{2} \lambda_1 \left(\sum_i \varepsilon_{ii} \right)^2 + \lambda_2 \sum_{i,j} \left(\varepsilon_{i,j} \right)^2 \right], \tag{3.12.8}$$

where λ_1 and λ_2 are the Lamé constants. The surface energy for a particle is calculated using the curvature of the interface. The curvature is a function of how many springs of a single particle are open to the fluid. The surface energy for a single particle is

$$\gamma_i = \gamma_{surf} \frac{1}{r_{surf}} V_s, \tag{3.12.9}$$

where γ_{surf} is the interfacial free energy between the solid and the fluid and r_{surf} is the local radius of curvature of the surface. The curvature drives dissolution in the rate law if it is convex towards the fluid and precipitation if it is concave. In order to account for long-range effects, the surface energy of the particle plus its right and left neighbours along the interface are averaged and added to the surface energy of the particle. The averaged surface energy along the interface then scales with 1/distance from the initial particle since the radius of the surface curvature increases (Eq. 3.12.9). This is done in sequential steps for 20 neighbours, by which time the long-range effects become very small compared to the surface energy of the particle itself.

In order to dissolve single particles along the interface, Eqs. 3.12.8-9 are used to calculate the Helmholtz free energy ($E_{el} + \gamma_i$) change relative to an unstressed straight interface and are used in Eq. 3.12.7 in addition to the fluid pressure (fluid pressure change is always relative to zero fluid pressure) and the concentration of dissolved material in the fluid to get a dissolution rate. We can then determine how many particles (N_p) will dissolve in the direction perpendicular to an interface in a given time step (dt) according to

$$N_p = D_r \frac{1}{r_i} dt \left(1 - \frac{N_{sp_i}}{6} \right), \tag{3.12.10}$$

where r_i is the radius of a particle i and N_{sp_i} is the number of springs that are still attached to a particle (not in contact with the fluid). Only one particle is dissolved during one internal time step of the dissolution routine. The particle that can dissolve the most is dissolved and the internal time step used is the amount of time that it took to dissolve that particle. Two different approaches are now taken. In the first approach only one particle is dissolved at a time step. In the second approach the internal time step is used to calculate how many particles will dissolve during a time step and the particles change their mass accordingly.

3.12.7 Example *Elle* Runs

We show two examples of fluid-solid interactions. The first example shows dissolution of a stressed crystal at its free surface whereas the second example shows the development of a stylolite from an initially smooth dissolution surface. Both examples show the formation of a roughness. The first example shows the development of a roughness in the form of grooves on the crystal surface out of an initial perturbation. This roughening is initially controlled by the perturbation but it develops progressively into a pattern that shows grooves on a relatively flat surface. These grooves have a well-defined wavelength or distance between them. This distance can be represented analytically as the development of an ATG-instability (Srolovitz 1989). The second type of roughness also develops on a flat surface and is initiated by a perturbation. In this case two crystals are pressed together. The roughness progressively grows on a wide range of scales in contrast to the grooves of the ATG-instability that have a well-defined length scale.

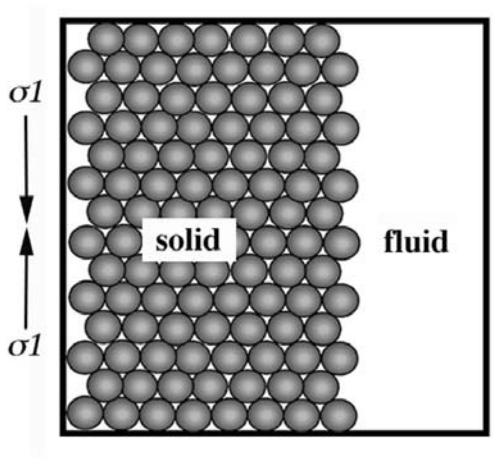


Fig. 3.12.2 Initial setup for the simulation of grooves on free surfaces of stressed crystals. The crystal is stressed vertically. All particles in contact with the fluid can dissolve as a function of elastic energy of particles and surface energy of the interface

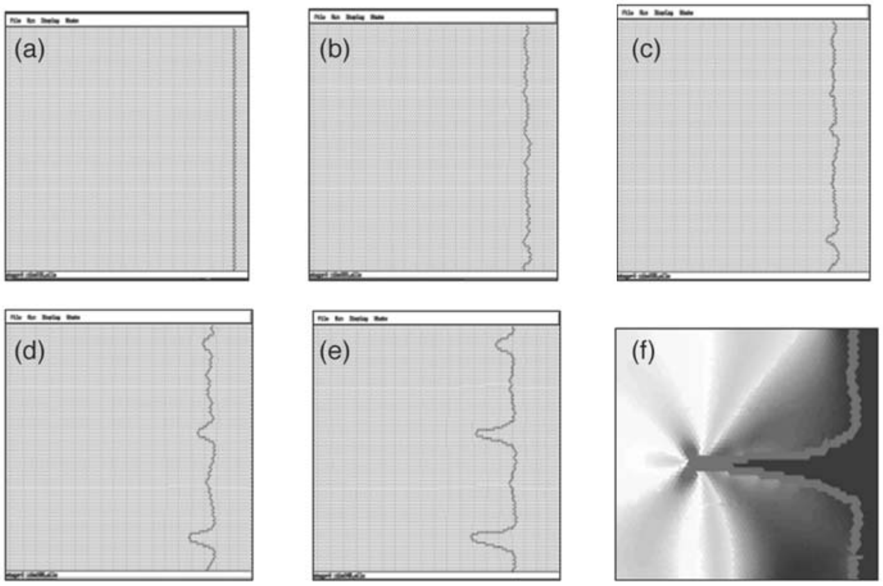


Fig. 3.12.3 Development of grooves and anti-cracks on the free surface of a stressed crystal. Horizontal resolution of the lattice is 100 particles. Fluid is on the right hand-side, the interface is marked by darker particles. **(a)** Initial setup of the model. A quenched noise is induced by a Gaussian variation of dissolution constants of particles in order to initiate the instability. **(b)** and **(c)** show the development of an initial roughness that depends on the quenched noise. In (c) the amplitude of the roughness starts to become large enough to induce the instability. **(d)** Development of three well-defined grooves with a clear spacing out of the initial random roughening. **(e)** Grooves become deeper and grow into anti-cracks. Coarsening processes lead to the deepening of the two lower grooves whereas the upper one is slowing down. **(f)** Differential stress field at the tip of an anti-crack. EXPERIMENT 12

Example 1: Growth of grooves and anti-cracks during dissolution

The first example of fluid-solid interactions shows the dissolution of the free surface of a stressed crystal in contact with fluid (Fig. 3.12.2).

The crystal is thought to be immersed in a large fluid volume so that the concentration in the fluid does not change. In the simulation the crystal is in contact with the fluid on its right hand side (Fig. 3.12.2). The fluid pressure is kept constant so that pattern formation along the interface of the crystal and the fluid is governed by gradients in elastic and surface energy. These energies are contained within the Helmholtz free energy (Eq. 3.12.1). Dissolution constants of the crystals have a Gaussian distribution with a variation of 0.3. This variation will produce an initial small scale roughening when the crystal dissolves (Fig. 3.12.3a-b).

The interface roughness increases elastic energy concentrations that drive dissolution. Surface energy will smooth the interface. The effect of the surface energy scales with the size of the system so that small-scale roughness disappears faster than larger scale roughness. Because of this scaling the roughness has to develop a critical size where elastic energy concentrations are larger than the effects of the surface energy. Once this size is reached stable grooves start to grow on the interface that have a well-defined wavelength. The interface between the grooves becomes increasingly smooth so that the final picture has great similarity to grooves that can be found on stressed crystals of salt in experiments (Fig. 3.12.1b). If the virtual experiment lasts longer these grooves develop into anti-cracks that run into the crystal perpendicular to the main compressive stress. The anti-cracks represent the ATG-instability. Note however that there are serious doubts in the literature that the solid behaves linear-elastic at the tip of anti-cracks (Grinfeld 1991). Since the presented model assumes linear elasticity the results concerning anti-cracks should be taken with great care. Differential stress concentrations at the anti-crack tip are shown in Fig. 3.12.3f.

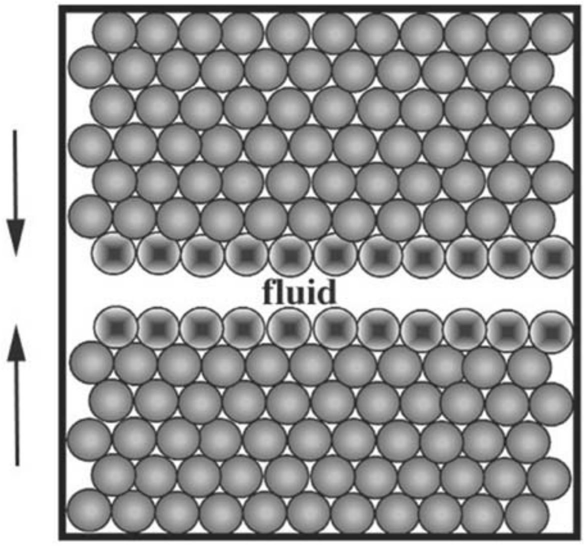


Fig. 3.12.4 Setup for the simulation of stylolite roughening. Two solids are pressed together vertically (see arrows). The crystals dissolve as a function of elastic and surface energies as well as normal stress

Example 2: Stylolite roughening

Example 2 shows the development of stylolites from flat dissolution surfaces. In this simulation two crystals are pushed together where particles can dissolve along the interface between the two crystals (Fig. 3.12.4).

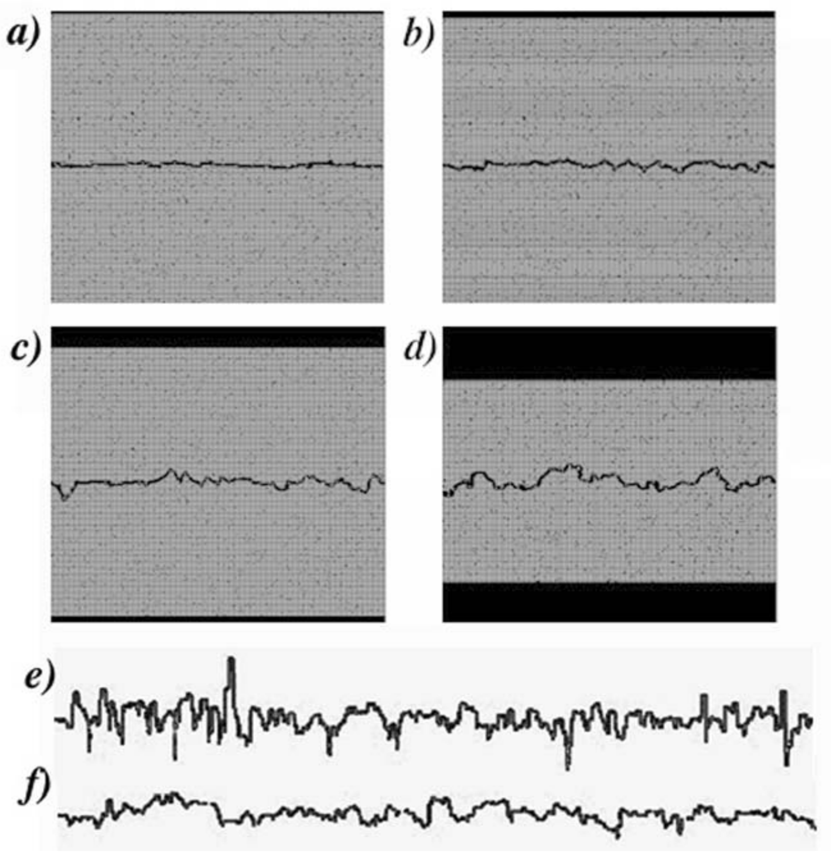


Fig. 3.12.5 Numerical simulation of stylolite formation in a lattice that is 200 particles wide. Roughness after **(a)** 200 time steps, **(b)** 500 time steps, **(c)** 1100 time steps and **(d)** 3500 time steps. **(e)** and **(f)** show mature stylolites (41000 time steps) in a lattice that is 800 particles wide. **(e)** Initial noise is a Gaussian distribution of dissolution constants, and **(f)** a bimodal distribution with about 3 percent of particles that dissolve slower. EXPERIMENT 13

We assume that the solid can only dissolve (and not re-precipitate) and the concentration in the fluid is kept constant assuming that the system is constantly flushed with fluid. Dissolution is now a function of elastic and surface energies as well as normal stress on the interface. The dissolution constants of particles have a Gaussian distribution (quenched noise) with a variation of 0.3. Figure 3.12.5a-d shows the development of a rough interface in a 200 particle wide lattice whereas Figs. 3.12.5e-d show two mature stylolites in an 800 particle wide lattice with two different kinds of noise. The quenched noise in the dissolution constants enhances the roughening whereas elastic and surface energies tend to smoothen the interface.

Symbols used in Chap. 3.12

c	Solubility at a given temperature [mol/m^3]
c^{eq}	Equilibrium concentration [mol m^{-3}]
c^i	Initial equilibrium concentration in an unstressed system [mol m^{-3}]
D_r	Dissolution/precipitation rate [m/s]
E_{el}	Elastic energy [J mol^{-1}]
ΔH_s	Change in Helmholtz free energy of the solid [J mol^{-1}]
K_{eq} (K_0)	Equilibrium constant of a reaction (for the unstressed solid)
k_r	Rate constant [mol s^{-1} m^{-2}]
N_{sp_i}	Number of springs that are attached to particle *i*
N_p	Number of particles that will dissolve in a given time step
R	Universal gas constant [8.3144 J mol^{-1} K^{-1}]
r_i	Radius of particle *i* [m]
r_{surf}	Local radius of curvature of a surface [m]
V_s	Molecular volume of the solid [m^3 mol^{-1}]
α	Activity coefficient of dissolved solid in a solution
γ	Interfacial free energy between solid and fluid [J m^{-2}]
γ_i	Surface energy of a single particle [J mol^{-1}]
$\Delta\mu$	Change in chemical potential [J mol^{-1}]
$\Delta\sigma_n$	Change in normal stress on a surface [Pa]
ε_{ij}	Strain tensor
λ_1, λ_2	Lamé constants [Pa]

3.13 Solid-solid phase transitions and heat conduction

Till Sachau and Daniel Koehn

3.13.1 Phenomenological observations

This chapter is written from the particular perspective of the olivine to spinel reaction in the transition-zone of the mantle, and is therefore concerned with polymorphic solid-solid phase-transitions under high pressure conditions in general, but with a certain emphasis on the properties of this reaction. The reaction is exothermal so that heat is released, which again influences the reaction. Therefore we are also including the description of heat conduction and its relation to the phase change in this chapter.

The kinetics of pressure-induced transitions have not been studied as extensively as the kinetics of temperature induced transformations, mainly because of technical difficulties in high-pressure experiments.

In contrast to cation exchange reactions, which additionally include large-scale diffusion between separate domains of a rock, polymorphic reactions are characterized by different structures of the same chemical compound. The olivine-spinel reaction is reconstructive in nature, involving breaking of bonds and building new structures within the crystal structure, which leads to large activation energies and a relative sluggishness of the reaction. Other polymorphic reactions (as for instance the transition from high to low quartz) rather involve a bending of bonds, thus having lower activation energies and larger transformation velocities. While the latter type is usually realized in temperature-driven transitions, the former, high-energy type is in generally related to pressure-driven transitions, and is the domain of high-pressure physics. A second difference between high and low pressure transitions is the role of strain energies, which can be usually neglected for temperature driven reactions.

The commonly proposed mechanism for the phase transition is that of nucleation and growth, in which spinel crystals nucleate on grain boundaries and grow into the olivine phase. The kinetics is controlled by diffusion of oxygen atoms across the surface, yet diffusion is restricted to the olivine-spinel surface and doesn't require large-scale diffusion. A particular topotactic relationship of the phases (i.e. the dependence on the orientation of the host) is not required by such a mechanism.

The olivine structure can be described as a hexagonal close-packed array of oxygen atoms, while spinel has a cubic close packing of the oxygen atoms. Cations have the same coordination in both minerals. This is unusual in high-pressure transitions, and it leads to a large volume-decrease of 8 percent due to the transition.

The progressive volume decrease during the transition combined with different velocities for grain boundary migration leads to a complicated interplay between the overall and local stress fields and consequently with the pressure driven transformation. This interplay leads - with certain limits - to a self-controlled process. The reaction occurs far from equilibrium, which is true under mantle conditions as well as in experiments. Under these conditions one cannot expect the resulting grain boundaries to have a smooth surface, but to have a very rough, fractal-like topography (Nordmeier 1999). Such structures have been observed in experiments with germanates (see Fig. 3.13.1, Riggs and Green 2001) where the structures have a size on the scale of several μm.

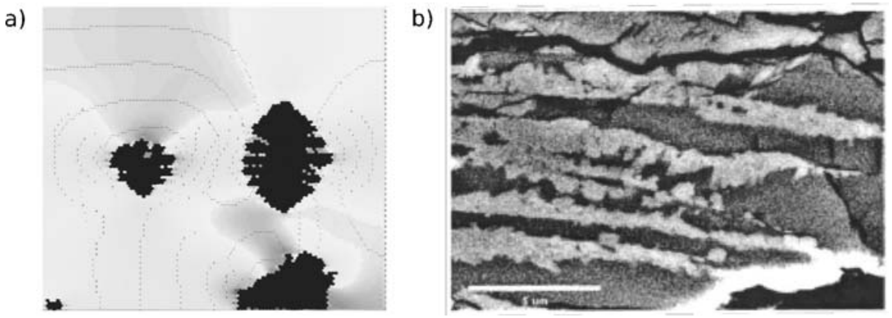


Fig. 3.13.1 Output of a simulation **(a)** in comparison with experimental results **(b)**. The experimental results were obtained in germanates, using a Griggs apparatus (Riggs and Green 2001; reproduced with kind permission of Elsevier). The simulation-run applies the above-described system, including latent heat release, and is based on an activation energy of 450 kJ. Despite the application of different types of deformation - uniaxial compression in (a) and shearing in (b) - similar patterns develop. Spinel is black and olivine grey in (a), whereas the high-pressure phase is light grey in (b)

3.13.2 Natural examples

General examples

Polymorphic phase transitions are common in metamorphic rocks of the crust as well as in the mantle. Examples range from the high/low quartz to the aragonite/calcite transition and the Al_2SiO_5 polymorphs.

However, natural examples of purely pressure driven transitions are rare in the research area of a common field geologist. Examples for accessible pure high-pressure polymorphs are the numerous varieties of carbon in the form of graphite, fullerenes or mantle-derived diamonds. In the vicinity of meteorite impacts, stishovite forms as a high-pressure variety of quartz.

High-pressure transformations in the mantle

The mantle has major density discontinuities at 400 km and 670 km, which are observed by transitions in calculated seismic velocities. These discontinuities divide the mantle into upper mantle, transition zone and lower mantle and could be due to abrupt changes in composition, but are generally assumed to be the result of high-pressure phase transitions in the constituent minerals of a uniform-composition mantle.

Petrological studies have shown that the upper mantle is mostly composed of Mg-rich olivine, which will be transformed in the given depth-range from the hexagonal close-packed array of oxygen-atoms to the cubic close-packed structure of spinel, thus providing a structural phase-transformation, which involves just the diffusion of an oxygen-atom from one site to another and involves a large increase in density at the 400 km discontinuity.

Strong evidence for the important role of the phase-transition is given by the occurrence of deep-seated earthquakes in the mentioned depth-range, which cannot be explained by simple brittle failure, but could develop due the changing mechanical properties of the mantle material and the dynamics of the phase-change.

3.13.3 Driving forces, equations describing the system

A sufficient pressure is a necessary condition, however the driving force for the phase transition and hence the rate of reaction is the change in Gibbs free energy before and after the phase transition. This again depends on the change of elastic and surface energies as well as the volume decrease due to the transition. Thus the interplay between local reaction rates plays a vital role for the structure of the phase boundaries at a specific point in time and space. The following equations were derived for the stochastic behaviour of a given system, and not for grain boundary structures in particular. The equations for time-independent growth and nucleation where originally derived by Johnson and Mehl (1939), Avrami (1939, 1940, 1941) and Turnbull (1956).

The Gibbs free energy is composed of two different contributions: the volume change of the phase and the change in free Helmholtz energy, where the latter is made up of the elastic energies and the surface energies within the system, thus the basic relation is (Shimizu 1992)

$$\Delta G = \sigma \Delta v + \Delta f, \tag{3.13.1}$$

where ΔG is the Gibbs free energy decrease, σ is the mean stress, ΔV the volume change due to the transition and Δf is the change in free Helmholtz energy of the system, which is commonly defined as the change in elastic and surface energies (Langbein 1999).

In addition, in the context of pressure-induced reactions, some energies may be released while others will be absorbed, which will expand Eq. 3.13.1 to

$$\Delta G = \sigma \Delta V + E_D + E_S + E_{PM} + \gamma_{NP}, \tag{3.13.2}$$

where E_D is the strain energy introduced by high pressure, E_{PM} is the strain energy created by the phase mismatch, γ_{NP} is the surface energy created by the new phase and E_S is the energy of heterogeneity (i.e. the energy of grain surfaces, point defects and others).

An important feature for the *direction* of pure grain boundary migration is the dependence of grain boundary reactions on the normal stress on the grain boundary (σ_n), which is (Gibbs 1906)

$$\Delta G = \sigma_n \Delta V + \Delta f, \tag{3.13.3}$$

A rate for a reaction is in principal the probability of a single or a number of molecules to change from one state to another. Assuming the growth rate to be time independent, which is generally the case unless growth is controlled by diffusion, the following kinetic laws can be derived.

For grain growth, the net rate (Y) can be calculated by the difference of the positive and the negative rates. With respect to the positive and negative energy contributions this gives (Sung 1976)

$$Y = Y_0 T\left[\exp\left(\frac{-(Q + E_D + E_S)}{RT}\right) - \exp\left(\frac{-(Q - \Delta G - E_{PM} - \gamma_{NP})}{RT}\right)\right], \tag{3.13.4}$$

where Y_0 is a pre-exponential factor, Q is the activation energy and R is the universal gas constant.

In case of nucleation, the rate (N) has to include a second activation energy ΔQ_{hom}, which is controlled by the surface and the strain energy of a nucleus smaller than a specific critical radius. A negative rate does not exist, since the nucleus is considered to form spontaneously. Therefore the nucleation rate is

$$N = N_0 T \exp\left(\frac{-\Delta G}{RT}\right) \exp\left(\frac{-\Delta Q_{hom}}{kT}\right). \qquad (3.13.5)$$

In case of spherical nucleation, ΔQ_{hom} is defined as

$$\Delta Q_{hom} = \frac{16\pi(\gamma_{NP})^3}{3(\Delta G_v + E_\varepsilon)^2}, \qquad (3.13.6)$$

where ρ is again the surface energy and E_ε is the strain energy of the nucleus. A standard-simplification is to ignore E_ε in the calculation. For a more complete review see e.g. Shekar and Rajan (2000).

3.13.4 Possible simulation techniques

Phase transitions

Various methods to simulate grain growth or molecular growth exist, though most are based on more or less purely probabilistic techniques (Monte Carlo methods), applying a probability to every particle to react. These models are purely interested in the shape of the evolving structures due to a single process, and do not (usually) involve a physically meaningful scaling in time or space. However, scaling in time and space are necessary in order to couple independent processes, which is necessary in order to study the full dynamics of a pressure driven phase transition. In the current context various techniques were tested - random techniques as well as non-random techniques - though for the sake of simplicity the following section is concerned with non-probabilistic number techniques only.

Heat conduction

The basic formula for heat-flow is the parabolic differential Fourier-equation, which can be solved by finite differencing. To solve the problem of heat conduction, several techniques can be applied, in particular finite differencing and Finite Element approaches. Relaxation-methods, as used for the stress-relaxation in *Latte*, can also be applied.

3.13.5 Implementation in *Elle*

The implementation is based on the discrete element code *Latte*, which is described in detail in Chap. 3.11. Discrete elements are referred to as "particles" that map to unodes in the *Elle* data structure. The code uses an underlying triangular lattice of linear elastic springs to allow the calculation of stress and strain tensors for every node, based on the normal stresses of the connecting springs. An additional rectangular lattice in the background allows the calculation of thermal conduction due to latent heat release. In the simulations particles are treated as independent kinetic subsystems, which allow the application of the mentioned stochastic kinetic laws. This setting treats the interaction of particles due to volume change and changing rheological properties.

The process, as implemented in *Latte*, aims to combine probabilistic techniques – in order to include chaotic, self-controlled grain growth – as well as non-random elements, to provide physical reliability.

Modelling grain-growth as a single process involves basically two steps within *Latte*: the calculation of the actual Gibbs free energy, combined with an internal phase-transition of particles sitting on grain boundaries and a subsequent calculation of the new energy state of the concerned particles. This allows the calculation of the difference in free energy, which can be used to calculate a growth-rate. This step is performed for each side of a hexagonal particle – using the normal stress in the affected direction – and thus allows the calculation of the time it takes to overgrow a complete particle.

Time steps are set dynamically, by calculating the transformation time for the fastest reacting particle, which will define the new t. Particles, which are not completely transformed in the given time step are partially transformed. In that respect the method is similar to phase fields.

Nucleation of spinel grains is not included in the current release of Latte.

Normal stresses on the grain boundary, as used in the context of grain boundary migration, can be calculated the following way (Ranalli 1995):

$$\sigma_n = \frac{1}{2}(\sigma_{11} + \sigma_{22}) + \frac{1}{2}(\sigma_{11} - \sigma_{22})\cos(2\Theta) + \sigma_{12}\sin(2\Theta), \qquad (3.13.7)$$

where Θ is the angle of the grain-boundary relative to the principal stress.

Positive and negative contributions to the elastic energy change can be approximated by subdividing the total strain into a spherical and a deviatoric component by

$$E_s = \frac{K(e_{ii})^2}{2}, \qquad (3.13.8)$$

and

$$E_d = \mu\left(e_{ij}e_{ij}\right), \tag{3.13.9}$$

where E_s is the energy of dilatation, E_d the energy of distortion (Saada 1993).

The expansion/contraction of the molar volume due to temperature has to be included into the calculation of the stress tensor under the extreme conditions of the olivine→spinel phase change, where the expansion coefficient κ can be calculated as

$$\kappa(T) = a + bT + cT^{-2}, \tag{3.13.10}$$

where a, b and c are parameters given by Akaogie (1989) and the actual molar volume by

$$V = V_0 \times \exp^{\left(\int_{298}^{T} \kappa(T)dT\right)}. \tag{3.13.11}$$

In order to solve the heat-flow equation, *Latte* uses a Finite Difference technique based on a square lattice. The solution is derived using an ADI (Alternating Direct Implicit) algorithm. The heat-flow solution is an extra class within *Latte* that needs an (primitive) interface to communicate with the triangular elastic lattice. Currently it can solve matrices for a single heat-capacity and heat-conductivity, thus neglecting differing parameters for different materials. The great advantage of using unified parameters is the considerably accelerated computation time, since the matrices have to be solved only once instead of having to be solved for every single vector.

A general difficulty in the Finite Difference-approach is how to handle deformation of the underlying lattice. In many well-known Finite Difference methods the solution is based on a fixed spacing in the x- and y-directions, in which case the system of equations cannot be solved for varying parameters. One of the strengths of the ADI-method, beside its stability for every Δt, Δx, Δy, is its solvability in spite of changing positions of lattice-boundaries.

During an ADI-run every time-step is split into two equal parts. In the first part, which means integrating from $k\Delta t$ to $(k+1/2)\Delta t$, x is taken to be implicit while y is treated explicitly. In a second step, from $(k+1/2)\Delta t$ to $(k+1)\Delta t$, the method is made explicit in x and implicit in y. In matrix form, Douglas and Peaceman (1955) proposed

$$\begin{aligned}(a)(I+B_1)\tilde{u}^{n+1/2} &= (I-B_2)u^n + \frac{k}{2}f^{n+1/2}(x-sweep),\\ (b)(I+B_2)u^{n+1} &= (I-B_1)\tilde{u}^{n+1} + \frac{k}{2}f^{n+1/2}(y-sweep),\end{aligned} \tag{3.13.12}$$

where f is a source term and will be ignored. This algorithm can be economized into a form, in which B_1 does not appear explicitly (Wachspresse and Habetler 1960). The terms 'explicit' and 'implicit' in the context of numerical mathematics refer to different schemes of the finite differencing (Chapt. 2.4, and e.g. Press et al., 1992).

Beside the advantage of being solvable for different spacing in x and y, the ADI method only involves solving a number of equations that are essentially tridiagonal and therefore computationally inexpensive.

Diffusion in the 2-dimensional case is generally governed by a parabolic partial differential equation of the form

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(D\frac{\partial u}{\partial y}\right), \tag{3.13.13}$$

where D is the diffusion coefficient and u may be for example concentration or temperature.

Application to thermal conduction yields the classical Fourier-equation

$$\rho c\frac{\partial T}{\partial t} = \frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right) + \dot{f}, \tag{3.13.14}$$

with ρ the density, c the specific heat and k the heat diffusivity. $\dot{f}$ is the remainder of the Taylor expansion and represents a source term, which can be ignored if no constant heat-sources exist, leading to a Laplace-term (Hering et al., 1992). Implicit finite differencing (i.e. approximating infinite terms by Finite Differences, compare e.g. Press et al., 1992) of Eq. 3.13.2 with $\dot{f} = 0$ now yields:

$$\rho c\frac{T_{i,k}^{n+1} - T_{i,k}^{n}}{\underset{\smile}{\Delta t}} = k\frac{T_{i+1,j}^{n+1} - 2\mathrm{T}_{i,j}^{n+t} + T_{i-1,j}^{n+1}}{(\Delta x)^2} + \frac{T_{i,j+1}^{n+1} - 2\mathrm{T}_{i,j}^{n+1} + T_{i,j-1}^{n+1}}{(\Delta y)^2}.. \tag{3.13.15}$$

In Eq. 3.13.15, n is the time-step, while j and k represent the spatial mesh. Transformed to the ADI-method, this gives

$$T_{j,k}^{n+1/2} = \alpha\left(\delta_x^2 T_{j,k}^{n+1/2}\right) + \alpha\left(\delta_y^2 T_{j,k}^{n}\right) + T_{j,k}^{n}, \text{ and} \tag{3.13.16a}$$

$$T_{j,k}^{n+1} = \alpha\left(\delta_x^2 T_{j,k}^{n+1/2}\right) + \alpha\left(\delta_y^2 T_{j,k}^{n+1}\right) + T_{j,k}^{n+1/2}. \tag{3.13.16b}$$

Including the definitions

$$\alpha = \frac{k\Delta t}{2\rho c(\Delta x)^2}; \quad \beta = \frac{k\Delta t}{2\rho c(\Delta y)^2}$$
$$x = k\Delta x; y = j\Delta y; k, j = 1,2,3,\ldots \tag{3.13.17}$$
$$\delta_x^2 T_{j,k}^n = T_{j+1,k}^{n+1} - 2\mathrm{T}_{j,k}^{n+1} + T_{j-1,k}^{n+1}; \quad \delta_y^2 T_{j,k}^n = T_{j,k+1}^{n+1} - 2\mathrm{T}_{j,k}^{n+1} + T_{j,k-1}^{n+1}$$

this idea expands to matrix-equations of the following type

$$\begin{pmatrix} 1 & 0 & \ldots & \ldots & 0 & 0 \\ -\alpha & 1+2\alpha & -\alpha & 0 & \ldots & 0 \\ 0 & \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & 0 & -\alpha & 1+2\alpha & -\alpha \\ 0 & \ldots & \ldots & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} T_{1,k}^{n+1/2} \\ T_{2,k}^{n+1/2} \\ \ldots \\ \ldots \\ T_{N,k}^{n+1/2} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \ldots & \ldots & 0 & 0 \\ \beta & 1-2\beta & \beta & \ldots & 0 & 0 \\ 0 & \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & 0 & \beta & 1-2\beta & -\beta \\ 0 & \ldots & \ldots & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} T_{j,1}^n \\ T_{j,2}^n \\ \ldots \\ \ldots \\ T_{j,N}^n \end{pmatrix} \tag{3.13.18}$$

and

$$\begin{pmatrix} 1 & 0 & 0 & \ldots & 0 & 0 \\ -\beta & 1+2\beta & -\beta & 0 & \ldots & 0 \\ 0 & \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & 0 & -\beta & 1+2\beta & -\beta \\ 0 & \ldots & \ldots & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} T_{j,1}^{n+1} \\ T_{j,2}^{n+1} \\ \ldots \\ \ldots \\ T_{j,N}^{n+1} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & \ldots & 0 & 0 \\ -\alpha & 1+2\alpha & -\alpha & 0 & \ldots & 0 \\ 0 & \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & 0 & -\alpha & 1+2\alpha & -\alpha \\ 0 & \ldots & \ldots & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} T_{1,k}^{n+1/2} \\ T_{2,k}^{n+1/2} \\ \ldots \\ \ldots \\ T_{N,k}^{n+1/2} \end{pmatrix} \tag{3.13.19}$$

These matrices are tridiagonal (i.e. they have non-zero elements only on the diagonal plus/minus one row) and can be solved by applying the Gauss-algorithm (compare e.g. Gershenfeld 1999; Press et al., 1992) or by an algorithm optimized for matrices of that form.

One can also solve it for the total system instead of solving it for single vectors, which is probably the easiest and – more important – fastest method, by simply uniting the temperature-vectors to a matrix.

Such an algorithm cannot be solved without the assumption of boundary conditions (i.e. setting boundary nodes to certain independently defined values), due to the missing parameters in the edges of the matrices. The boundary condition can be set for instance according to a Dirichlet condition, which governs the temperature of boundary nodes as a function of time. In case of the algorithm as implemented in *Latte*, T as a boundary condition is currently kept constant at a reasonable temperature (for instance 1000 K).

3.13.6 Examples

Heat conduction

An example for a combined phase-change and heat-flow experiment is given in its own chapter (Chapt. 4.5). In Fig. 3.13.2 a simple example is shown in case where four grains were initially at1500 K, and the surrounding grains at 1000 K. Boundary conditions were set at 1000 K and the system was let to cool to this temperature. Note that the conditions for the deformation and the temperature-distribution are extreme and serve only to visualize the concept, although they are based on realistic physical properties for olivine. The size of the box is set to 1 mm, the time-step to 0.1 s.

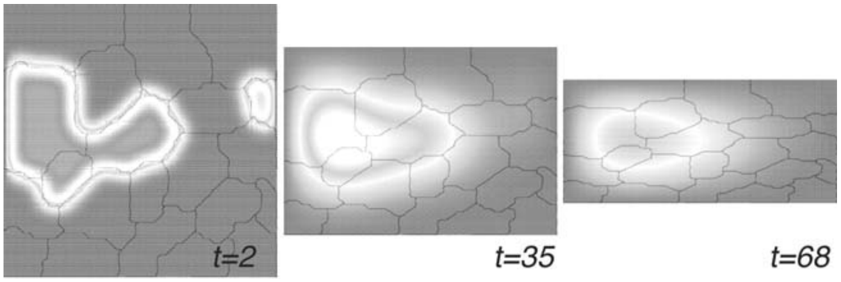


Fig. 3.13.2 Heat conduction under vertical shortening of the material. Boundaries and grains are initially at 1000 K (dark grey), except for four grains that started at 1500 K (light grey). The size of the model is 1 mm, time step t = 0.1 s, the heat capacity c = 1005 J/(kg K), the density is $4.2{\cdot}10^3$ kg/m^3 and the thermal diffusivity is $1{\cdot}10^{-6}$ m^2/s. These are typical values for olivine

Phase Transition

This example illustrates the developing structures during a high pressure phase transition including latent heat release. Figure 3.13.3 shows the growing spinel in a matrix of olivine where spinel is black. Shading in the olivine matrix indicates the differential stress. The main compressive stress is vertical. The growing aggregates have an elongation parallel to the main compressive stress. On their sides little anti-crack like structures grow perpendicular to the main compressive stress.

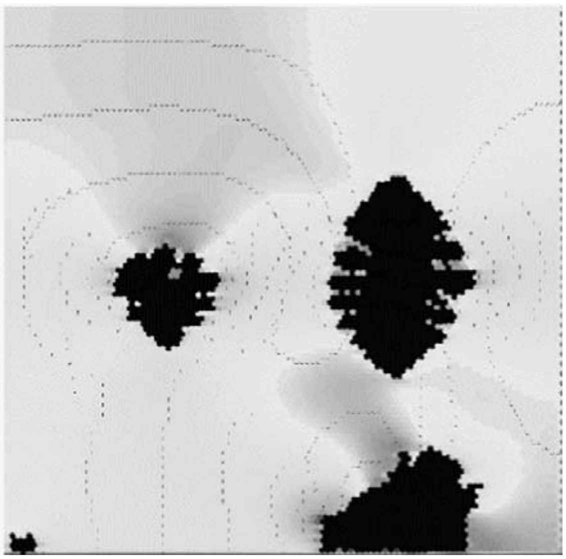


Fig. 3.13.3 Simulation of spinel growth in an olivine matrix. Spinel is black and olivine grey. Differential stresses are shown as grey scales in the olivine matrix

Symbols used in Chap. 3.13

B	Coefficient matrix
c	Specific heat [J kg^{-1} K^{-1}]
D	Diffusion coefficient [m^2 s^{-1}]
E_D	Strain energy introduced by high pressure [J mol^{-1}]
E_d	Energy of dilatation [J mol^{-1}]
E_{PM}	Strain energy created by phase mismatch [J mol^{-1}]
E_S	Energy of heterogeneity [J mol^{-1}]
E_ε	Strain energy of nucleus [J mol^{-1}]
E_s	Energy of distortion [J mol^{-1}]
e	Strain tensor
$\dot{f}$	Remainder of Taylor expansion series in Eq. 3.13.14
Δf	Change in Helmholtz energy of the system [J mol^{-1}]
ΔG	Change in Gibbs free energy [J mol^{-1}]
ΔQ_{hom}	Activation energy associated with homogeneous nucleation [J mol^{-1}]
ΔG_v	Free energy change due to volume change
K	Bulk modulus [Pa]
i,j	Spatial mesh indices
k	Heat diffusivity [m^2 s^{-1}]
l_1, l_2, l_3	Parameters used in Eq. 3.13.10
N, N_0	Nucleation rate; associated pre-exponential factor [m^{-3} s^{-1}]
Q	Activation energy [J m^{-3}]
s	Source term in Eq. 3.13.12
$\tilde{u}, u$	temperature vector (in y and x respectively)
V, V_0	Molar volume; associated pre-exponential factor in Eq. 3.13.11
ΔV	Volume change due to phase transition [m^3 mol^{-1}]
Y, Y_0	Net rate for grain boundary migration [m s^{-1}], and pre-exponential factor
α, β, δ	Mathematical variable
γ_{NP}	Surface energy created by new phase [J m^{-2}]
Θ	Angle between grain boundary and principal stress
κ	Expansion coefficient for phase change.
μ	Shear modulus [Pa]
ρ	Density [kg m^{-3}]
σ, σ_n	Mean stress [Pa], Normal stress acting on grain boundary [Pa]

4 Case studies and coupling of processes

Editors: Mark W. Jessell, Daniel Koehn and Paul D. Bons

This chapter with eight authored sections presents a selection of possible application of microdynamic simulation to address geological questions. The various processes that have been introduced in the previous chapter were used, sometimes with minor additions or modifications. Because processes in rocks never operate in isolation, the reader will see that the various authors in this chapter have combined two or more processes to simulate the microstructural development under investigation. As such the authors have fully taken advantage of the possibility of the *Elle* software to couple processes.

4.1 Grain shapes and sizes during static grain coarsening in salt

Mark W. Jessell, Janos L. Urai and Oliver Schenk

4.1.1 Anisotropic grain growth

Grain boundary energy anisotropy (the anisotropy of grain boundary energy (γ) as a function of grain boundary-lattice orientation) plays a demonstrable role in determining grain shapes in salt (NaCl) polycrystals undergoing grain growth (Fig. 4.1.1). The influence this anisotropy has on grain-growth kinetics (Rohrer 2005), and the nature of the grain boundary networks that develop have been investigated using the *Elle* grain boundary migration module.

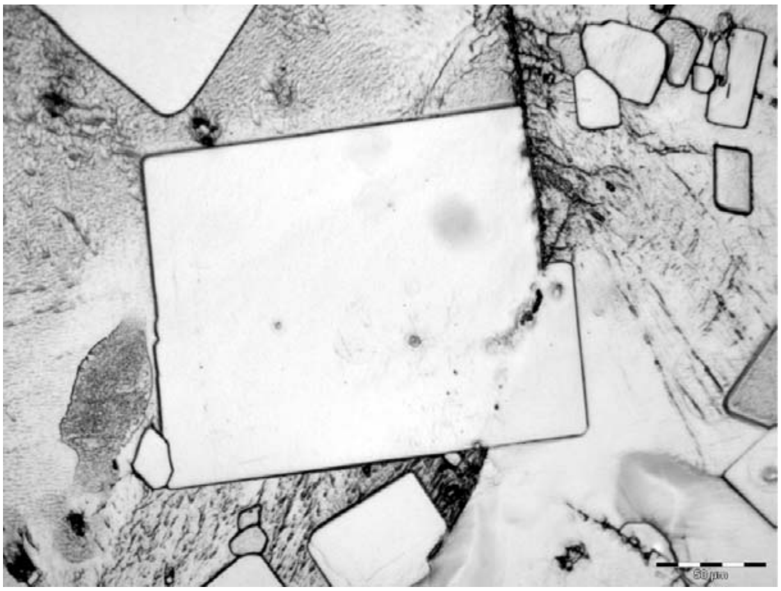


Fig. 4.1.1 Grain growth in salt (NaCl). Reflected light image of a polished and etched surface of a sample of compacted polycrystalline salt showing the growth of new euhedral grains (Schenk 2004)

4.1.2 Implementation in *Elle*

This experiment uses the code described in (Bons et al., 2001) to simulate boundary energy driven grain growth in a material with an anisotropic boundary energy. Unfortunately, the actual anisotropy of grain boundary energy for NaCl is not known, so instead we draw upon data from other face-centred cubic (fcc) systems, which are isostructural with NaCl (such as MgO) for some help. In these simulations we ignore any independent anisotropy of grain boundary mobility, so that the patterns of anisotropy could in fact be equally well interpreted in terms of the mobility term. The anisotropy patterns for some isostructural fcc systems have been calculated experimentally (Saylor et al., 2000) and from first principles (Braun et al., 1997). The two approaches give similar patterns and magnitudes of anisotropy. For the experimental results, the maximum of the best-fit surface energy function is at (111) and the minimum is at (100). The relative surface energies of the low index planes are $\gamma_{(110)}/\gamma_{(100)} = 1.040{\pm}0.008$ and $\gamma_{(111)}/\gamma_{(100)} = 1.072{\pm}0.010$ (Fig. 4.1.2).

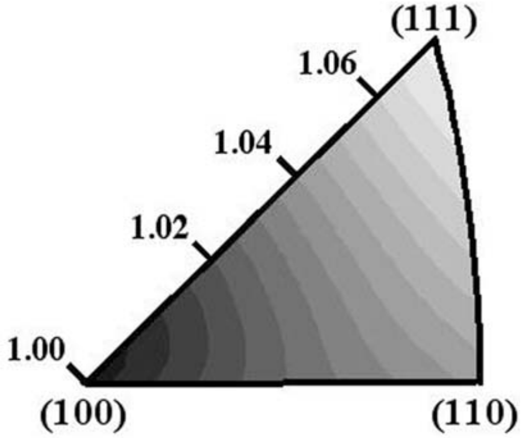


Fig. 4.1.2 The variation of normalised interfacial energy for MgO at 1400°C, derived from thermal groove data (Saylor et al., 2000)

In order to reduce the complexity of the system, we have only considered a single-axis model for boundary energy variations, and have used the range of anisotropies along the (100)-(111) plane, which is in any case displays the greatest anisotropy variation (Fig. 4.1.3)

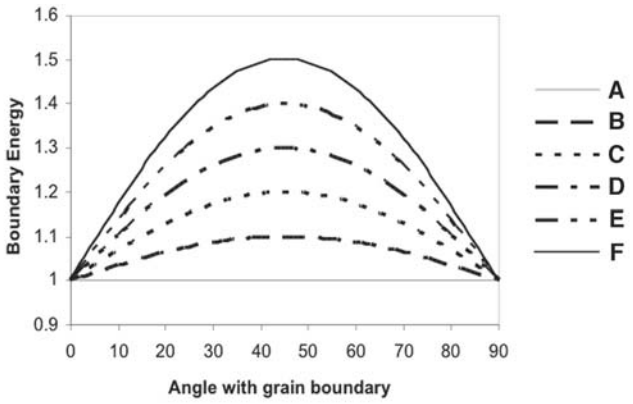


Fig. 4.1.3 Variations in normalised boundary energy as a function of the angle of the boundary with the 100-axis, for the six experimental runs

An initially randomly oriented aggregate consisting of 315 grains (and hence an average grain area of 3.17 x 10^{-3} (the entire model has unit area) with a foam texture, was allowed to grow for between 100,000 and 300,000 time steps (this may take several hours). Experiment A, which has an isotropic surface energy distribution, produces a foam texture with triple junction angles of about 120°. At a first glance, the microstructure in the other experiments looks like a foam texture, but most triple junction angles are in fact far from 120 degrees (Fig. 4.1.4).

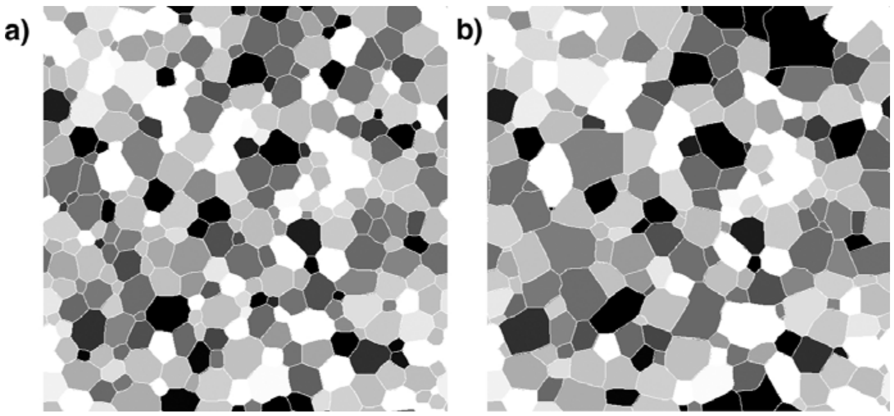


Fig. 4.1.4 Experiment F. **(a)** Time step 0, shading reflect lattice orientation. **(b)** Time step 100,000. Most triple junction angles reflect surface energy anisotropy by being far from 120 degrees. EXPERIMENT 15

The rate of grain growth in all experiments with a surface energy anisotropy is vastly decreased relative to an identical experiment where surface energies were assumed to be isotropic (A), and experiments with a high boundary energy anisotropy quickly reach an apparently stagnant grain size (Fig. 4.1.5). After a considerable time in the experiments with higher surface energy anisotropies, a second phase of grain growth commences, dominated by the exaggerated growth of just one or two grains (Fig. 4.1.6).

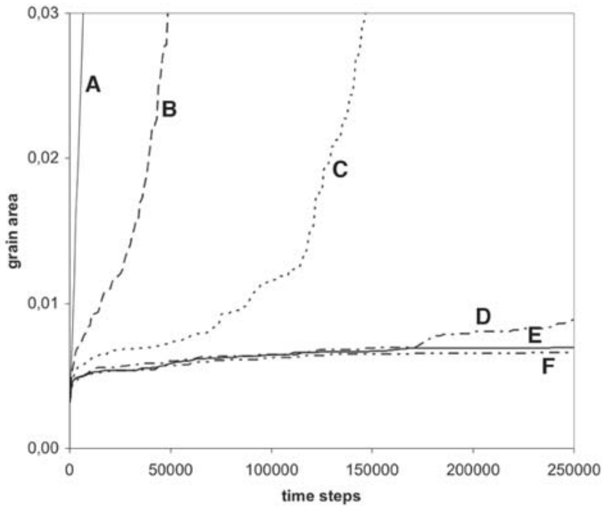


Fig. 4.1.5 Evolution of average grain area as a function of time, for a range of boundary energy anisotropies. Note the extended period of slow grain growth that occurs at higher anisotropies

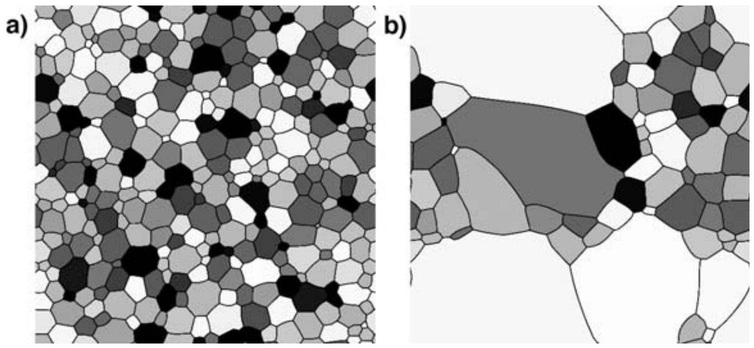


Fig. 4.1.6 Experiment C. **(a)** Time step 0, colours reflect lattice orientation **(b)** Time step 200,000. EXPERIMENT 15

4.1.3 Discussion

These experiments demonstrate that increasing boundary energy anisotropy initially modifies the effective grain boundary mobilities. For isotropic grain growth, the rate of grain area increase is linear with time. However, for the non-anisotropic cases the rate of growth undergoes a three stage evolution, with an initial period of rapid growth, followed by a period of slower growth whose length increases with increasing anisotropy, and finally a second period of rapid growth, which shows a similar rate of grain area increase as the isotropic case. This three-stage behaviour renders predictions of growth rate in natural polycrystals extremely difficult, and can considerably extend the predicted time needed to modify the grain size of a sample.

As the anisotropy increases, the final rapid grain growth stage is in fact a period of exaggerated grain growth, where only a few grains are increasing in size, and the ‘matrix’ grains remain effectively frozen.

Perhaps surprisingly, even with high anisotropies (and this has been confirmed with even more extreme models than those presented here) the grain shapes themselves are still relatively isotropic, and there is no evidence for the macroscopic faceting seen in Fig. 4.1.1. It appears that to get macroscopically facetted grains you need the boundary shape to be dominated by the surface energy anisotropy of the growing grain.

We can imagine two principal explanations for this:

1. Dislocations or impurities in the matrix grains reduce the boundary anisotropy, but the growing, and hence clean, grain retains its anisotropic properties. In this case we would not expect faceting in pure systems, and it would also be suppressed when grain boundaries between two growing merge (as appears to be the case for the impinging grains at the top right of Fig. 4.1.1)
2. There is an inherent asymmetry in the dissolution and precipitation processes at the atomic scale, such that the dissolution process is isotropic, but the precipitation processes is anisotropic. Evidence for this comes from resorption microstructures in partial melts, which are typically rounded, in comparison to the euhedral crystal forms of crystallising grains (Lowenstern 1995).

Clearly these simulations themselves are unable to resolve the real causes for faceting in salt polycrystals undergoing grain growth, however they do demonstrate that we need to go beyond the simple inference that boundary energy anisotropy is in itself sufficient to cause this effect.

4.1.4 Conclusions

1. Anisotropic boundary energies can considerably slow down the rate of grain growth, and materials with anisotropic boundary energies can undergo a three-stage growth behaviour.
2. Boundary energy anisotropies cannot in themselves produce macroscopically facetted grains during grain growth, and an asymmetry of anisotropies across grain boundaries is suggested.

4.2 Dynamic recrystallisation and crystalline plasticity

Sandra Piazolo

4.2.1 Introduction

Dynamic recrystallisation occurs within a polycrystalline material during solid state deformation. Through dynamic recrystallisation a crystalline aggregate lowers its free energy by establishment of an array of grain boundaries in new material positions (Means 1983) and formation and/or migration of grain boundaries (Vernon 1981). Dynamic recrystallisation is characterized by a number of concurrent processes that act upon the polycrystalline material. As such it is useful to use a numerical simulations system that allows simulation of concurrent micro-scale processes.

The study of the effect of dynamic recrystallisation on the microstructure is of major interest to the metals industry to intelligently gear processing techniques towards desired microstructures, but also for geologists to interpret the microstructures seen in geological samples. Microstructural characteristics induced by dynamic recrystallisation are used to evaluate the dominance of different processes active at the grain scale (e.g. Doherty et al., 1997; Urai et al., 1986; Hirth and Tullis 1992). In order to interpret microstructures that developed during dynamic recrystallisation, it is vital to know how they form. Questions to be answered involve the way and extent at which specific processes and combinations of processes affect the microstructure and the rheology of a rock during progressive deformation.

In this study I am particularly interested in the effect of the variable dominance of grain boundary migration, subgrain formation, rotational recrystallisation, and nucleation of new grains from strained grains on the microstructural development of a polycrystalline aggregate.

4.2.2 Theoretical background and implementation in *Elle*

In the model presented here, I simulate dynamic recrystallisation during plane strain simple shear deformation of a polycrystal, involving crystal lattice rotation, formation of subgrains, recrystallisation by nucleation, recrystallisation by subgrain rotation (also called rotational recrystallisation), grain boundary migration and recovery. Figure 4.2.1 shows a schematic flow diagram of the *Elle* modeling system used in the simulations of dynamic recrystallisation. In the following, the theoretical background and implementation in *Elle* is briefly summarized. For more details, such as the complete list of parameters used and reasoning behind those the reader is referred to Piazolo (2001) and Piazolo et al. (2002).

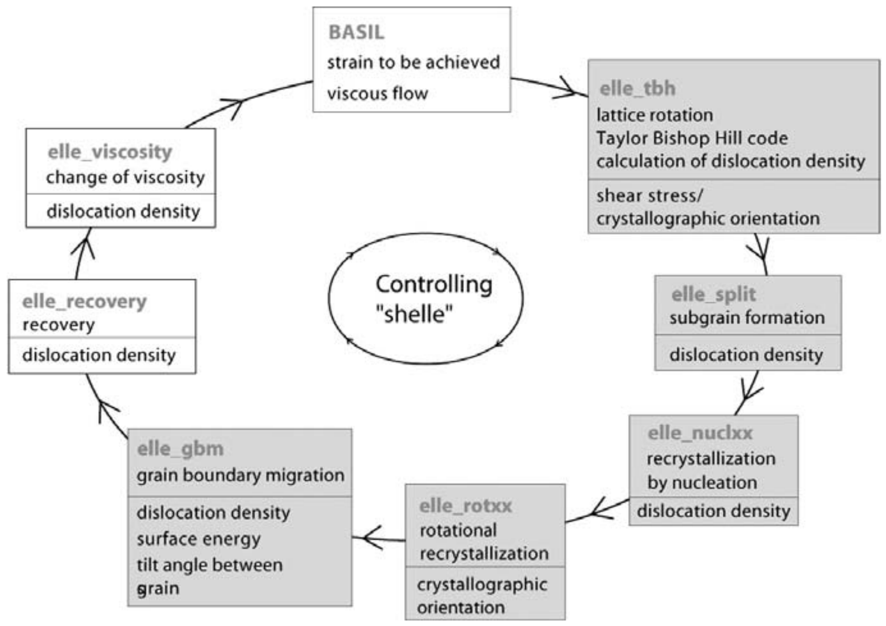


Fig. 4.2.1 Schematic flow diagram of the dynamic recrystallisation model. In the white boxes, processes (upper part of box) are given which have no true driving force (e.g. calculation of viscosity). The parameters used for their calculations are shown in the lower part of box. Grey boxes signify processes (top of box) with driving forces (lower part of box) (modified from Piazolo 2001)

Basil - Viscous Deformation

The plane strain simple shear deformation is modelled by the two-dimensional Finite Element code *Basil* (Barr and Houseman 1992, 1996, 1999; see Chapt. 3.8) that can solve for linear and non-linear rheologies by the constitutive laws. In simulations presented here, a plane strain approximation with cyclic boundaries was used to simulate simple shear deformation. The material properties were chosen to be Newtonian viscous and the strain per time step is 0.05. Runs were performed up to a finite simple shear strain of two.

elle_tbh - lattice rotation and dislocation density calculation

Lattice rotation of grains and subgrains and accumulation of dislocations within these are two phenomena arising from dislocation glide during progressive deformation. Rotation of the crystal lattice is required to accommodate an arbitrary deformation because deformation by slip is possible only along a limited number of slip systems. The resistance to dislocation movement increases as dislocation pile-ups or tangles form, and additional stress is needed to further move dislocations. The crystal is said to "work-harden" (Kocks 1976, 1985; Mecking and Kocks 1981; Barber 1985).

The routine that models the lattice rotations and the accumulation of dislocations consists of two parts. The first part is the calculation of the crystal lattice rotation and the associated work using the Taylor-Bishop Hill-calculation method (for details see Chap. 3.9; Taylor 1938; Bishop and Hill 1951a, 1951b; Lister and Paterson 1979) and the second part is the calculation of the accumulation of dislocations. The amount of accumulated dislocations is calculated by using the work term from the *elle_tbh* calculation. Theory (Argon 1970; Kohlstedt and Weathers 1980) and observations of experiments (e.g. Durham et al., 1977; Beemann and Kohlstedt 1988; De Bresser 1996) suggest a general positive correlation between dislocation density and differential stress. Therefore, the dislocation density at time step t, $\rho_{(t)}$, of a flynn undergoing work is calculated according to the following equation

$$\rho_{(t)} = a\rho_{(t-1)} + bW \, , \qquad (4.2.1)$$

where $\rho_{(t-1)}$ corresponds to the previous dislocation density of the flynn and the dimensionless parameter W is the calculated work for deformation of the flynn according to the Taylor-Bishop-Hill code. a and b are chosen to be 0.4 and 0.7, respectively. The sum of the constants a and b is chosen to be larger than unity to ensure that there is an increase of dislocation density with progressive deformation.

elle_split - subgrain formation

Subgrain formation is simulated as described in section 3.4, where the new subgrain boundaries are chosen to be preferentially parallel or perpendicular to the crystallographic c-axis of the parent grain. The threshold dislocation energy density E_{sg} for subgrain formation is calculated according to

$$E_{sg} = \rho E_{\rho} \rho_{scale} , \qquad (4.2.2)$$

where E_ρ the energy of dislocations and ρ_{scale} the dislocation density scaling factor. If $E_{sg} > \tau_{split}$ (energy threshold value for splitting of a polygon), the grain has a probability to “split” into subgrains. The probability for “splitting” increases with increasing E_{sg}. τ_{split} is chosen to be $2.4 \cdot 10^5$ J/m^2 (Gottstein and Shvindlermann 1999).

elle_nucl_xx – recrystallisation by nucleation

The theoretical background and general implementation of this process is described in Chap. 3.4. For a nucleus to be successful, i.e. to develop into a grain, the total free energy must decrease during expansion of the nucleus. For this (a) a critical nucleus size has to be exceeded, (b) there must be an instability of the microstructure, like differences in dislocation density, and (c) the boundary must be a mobile, high angle grain boundary (Gottstein and Mecking 1985) on at least one side of the nucleus.

The driving force for recrystallisation by nucleation is the strain energy of the grain or subgrain, E_{sg} (Eq. 4.2.2). The critical threshold value τ'_{nucl} of a specific mineral species at a specific temperature is calculated from a base nucleation threshold value τ_0 specific to a mineral species

$$\tau_{nucl} = \frac{\tau_0}{0.001T + 1}, \tag{4.2.3}$$

where T is the temperature. If $E_\gamma > \tau_{nucl}$ there is a certain probability that a nucleus which fulfils the three requirements for recrystallisation by nucleation is present in a polygon. In simulations $\tau_0 = 4.8 \cdot 10^5$ JK/m^2.

elle_angle_rx - Rotational Recrystallisation

A subgrain undergoes recrystallisation by rotation if all of its subgrain/low angle boundaries develop into grain/high angle boundaries.

For each time step, the routine calculates the misorientation angle between adjacent polygons. If a polygon is completely surrounded by high angle boundaries the subgrain is promoted to grain status, without changes to any other attributes e.g. dislocation density or crystallographic orientation. In the model, the critical misorientation angle between a high and low angle boundary was 10° according to values given for quartz (e.g. White 1979; Lloyd et al., 1992; Trimby et al., 1998).

elle_gbm – grain boundary migration

Grain boundary migration occurs due to the difference in stored internal strain energy and in chemical potential due to curvature between adjacent grains.

In the model grain boundary migration is simulated according to Chap. 3.5. The following two driving forces are taken into account:

1. change in surface energy and
2. change in strain energy stored in adjacent polygons.

In the model the surface energy is a function of the c-axis misorientation angle between two adjacent grains or subgrains. The influence of the c-axis misorientation angle is chosen so that the calculated surface energy of a specific boundary is zero at a c-axis misorientation angle of 0°, 0.1 × surface energy E_Γ at 10° and 0.99 × E_Γ at 20° and above. E_Γ is chosen to be $7\cdot10^{-2}$ Jm^{-2} according to measurements of surface energies provided by Urai et al. (1986), Gottstein and Shvindlermann (1999) and Stöckert and Dyster (2000).

elle_recovery – recovery of dislocations

Thermodynamically unfavourable lattice defects are removed during recovery. This process is simulated by a reduction of the dislocation density of each grain or subgrain per time step according to

$$\rho_{(t)} = R_{recovery}\rho_{(t-1)}, \tag{4.2.4}$$

where $R_{recovery}$ is a temperature dependent recovery factor. In our model (at $T = 450$ °C) $R_{recovery}$ is 0.94.

elle_viscosity - viscosity calculation

The rheology of the polycrystal is assumed to be viscous. In the model, it is assumed that the activation energy required to overcome an obstacle in the crystal lattice is the rate-limiting factor for the deformation of the material and therefore the viscosity (η) is related directly to the square root of the dislocation density (Argon 1970; Frost and Ashby 1983):

$$\eta = \eta_0 + v\rho^{1/2}, \tag{4.2.5}$$

where η_0 is a base viscosity specific to the mineral species and v a mineral-specific parameter (set to one in all applications of this routine in this book). η_0 is chosen to be $3.15\cdot10^{18}$ Pa s assuring that the viscosities in our simulations correspond to the range of viscosities suggested for greenschist to amphibolite facies conditions (e.g. Carter and Tsenn 1987; Clark and Royden 2000).

4.2.3 Experimental Runs

To investigate the effect of the relative rates of the different microscale processes active during dynamic recrystallisation experiments were performed in which one parameter was varied. The grain boundary mobility (M_{gb}) was chosen to be either $1 \cdot 10^{-12}$, $20 \cdot 10^{-12}$ or $40 \cdot 10^{-12}$ $m^2s^{-1}J^{-1}$. The interested reader is referred to Piazolo (2001) and Piazolo et al. (2002) for further simulations the energy threshold value for recrystallisation by nucleation was also varied. Two different starting microstructures were used: one fine-grained and one-coarse grained aggregate (Fig. 4.2.2). Figure 4.2.3 shows the development of the microstructure in simple shear strain intervals of 0.25, showing both the grains as well as their dislocations densities. Movies of the experiments are provided on the CD and a variety of image sequences of developing microstructures during progressive deformation are provided on
www.uni-mainz.de/FB/Geo/Geologie/tecto/elle_movies.

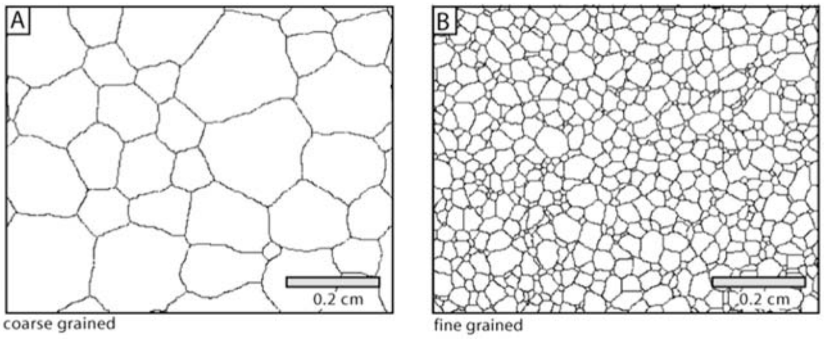


Fig. 4.2.2 Starting microstructures where only high angle boundaries are shown; **(a)** coarse grained, **(b)** fine grained (modified from Piazolo 2001)

4.2.4 Results

Simulated microstructures (Fig. 4.2.3) correspond to microstructures that are commonly attributed to dynamic recrystallisation (e.g. Poirier and Nicolas 1975; Guillopé and Poirier 1979; FitzGerald et al., 1983; Hirth and Tullis 1992; Takeshita et al., 1999). These can be divided in those that develop in initially coarse-grained aggregates and those in initially fine-grained aggregates. In initially coarse-grained aggregates, the following features are seen:

- bimodal grain size distribution, where large, strongly elongated grains that are surrounded by small grains remain even at a strain of 2 (Fig. 4.2.4a),
- small recrystallised grains develop predominately at the rim of large, old grains (Fig. 4.2.4a),
- strong shape fabrics develop (Fig. 4.2.3)
- and domains of similarly oriented grains are observed at low recrystallisation rates and low grain boundary mobility (Fig. 4.2.4b).

In contrast, a uni-modal grain-size distribution develops (Fig. 4.2.4c) and the grain shape fabric is weaker (Fig. 4.2.3) in initially fine-grained aggregates.

If we analyse results in terms of increasing grain boundary mobility, then the following can be observed: the irregularity of grain boundaries increases (Fig. 4.2.3), the mean grain size increases, the grain size of recrystallised grains increases, the aspect ratios decreases, the overall dislocation density decreases, the distribution of dislocations becomes more homogeneous, the total amount of recrystallised grains (that is grains that are formed by recrystallisation by nucleation and rotational recrystallisation) decreases with increasing grain boundary mobility and the total length of both high and low angle boundaries decreases with increasing grain boundary mobility (see Piazolo (2001) and Piazolo et al. (2002) for a quantitative analysis).

Table 4.2.1. Settings and abbreviations for simulations

Grain boundary mobility M_{gb} [$m^2s^{-1}J^{-1}$]	Coarse-grained	Fine-grained
$1 \cdot 10^{-12}$	**cg1-1**	**fg1-1**
$20 \cdot 10^{-12}$	**cg1-2**	**fg1-2**
$40 \cdot 10^{-12}$	**cg1-3**	**fg1-3**

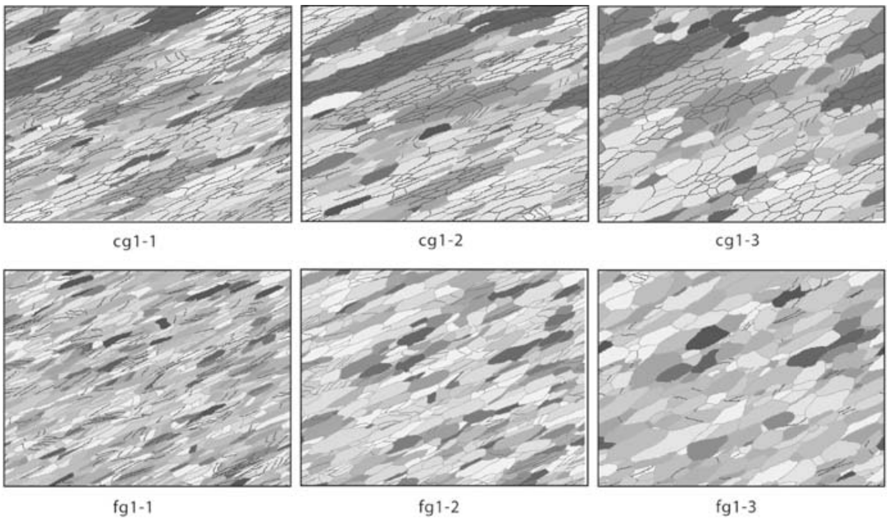


Fig. 4.2.3 Results of the 6 simulation runs. Different grey colours signify different grains, dark boundaries low angle boundaries and light grey coloured boundaries high angle boundaries. For the nomenclature of simulation runs refer to Table 4.2.1 (modified from Piazolo 2001). EXPERIMENT 16

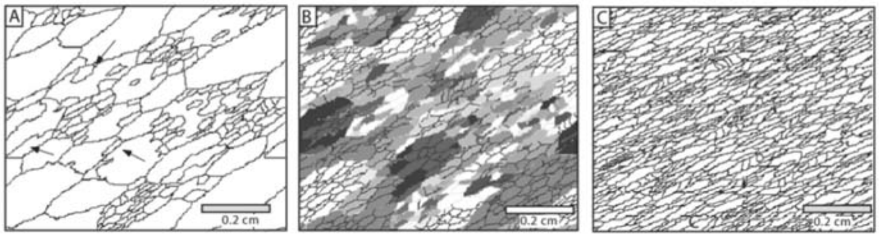


Fig. 4.2.4 Selected simulated microstructures. **(a)** High-angle grain boundaries in an initially coarse grained aggregate (cg1-1) showing a bimodal grain size distribution due to recrystallised grains at rims of large grains (core and mantle structures; arrows). **(b)** Crystallographic orientation in grey scales in the same simulation. Note clusters of similarly oriented recrystallised grains; different shading of grain boundaries signify different misorientation angles (dark < 10°, light > 10°). **(c)** High-angle grain boundaries in an initially fine-grained aggregate (fg1-1). Scale bars given assume a unit cell width of 1 cm. Shear strain is two. Modified from Piazolo (2001)

4.2.5 Discussion

The fabric development is significantly influenced by its initial grain size in the simulations. A strong shape fabrics and a bimodal gain size distribution develops in simulations of initially coarse-grained aggregates, whereas the shape fabric is weaker and the grain size distribution is uni-modal in initially fine-grained aggregates (Fig. 4.2.3). In addition, the recrystallised grain size in initially coarse-grained aggregates is larger than the recrystallised grain size in the fine-grained counterpart.

My results show that a change in the parameter value such as grain boundary mobility is sufficient to produce microstructures that are commonly interpreted to be the result of the strong dominance of one single process. The same processes operate in each of the simulations. The important difference between the different simulations is that the relative rates of the simulated processes vary due to the change in the value of grain boundary mobility. Although all processes are active and have some effect on the microstructure at all times, the rate at which one process affects the microstructure may be significantly higher than that of the other processes. Accordingly, changes in parameters that are related to different conditions with respect to e.g. temperature, strain rate and fluid activity result in differences in developing microstructures caused by different relative rates of concurrent processes. Once it is know how the absolute and relative rates of concurrent processes vary with parameters such as temperature, stress, strain rate, finite strain, fluid absence/presence and mineral species, we can use the resultant microstructures to interpret conditions prevailing during the deformation of the rock analysed.

4.2.6 Discussion of technique

Numerical simulations

The general correspondence of results of the simulations with microstructures observed in rocks suggests that the method of discretisation of a complex system of interacting, concurrent processes to a set of successive processes active in small time and length intervals is indeed a valid numerical technique. In addition, algorithms describing the effect of an individual process on a microstructure seem to be a good approximation to effects of the corresponding process active in nature and values of parameters seem to be in the range of physical values.

Nevertheless, the model can and should be improved significantly. Main improvements that could be envisaged for the current *Elle* system are:

1. Multiscale modelling: e.g. linking of the Potts model for subgrain growth and formation with the boundary model of subgrain formation.
2. More rigorous calculation and bookkeeping of the types of dislocations that are generated, recovered and used in subgrain formation (dislocation simulations). This would give better constraints on the types of subgrain walls forming and their crystallographic orientation.
3. Extension of simulations to finite shear strains of 20 or higher.

4.2.7 Conclusions

Numerical simulations of microstructures that undergo progressive dynamic recrystallisation show that different values of grain boundary mobility result in microstructures that resemble those observed in natural examples and in experimentally deformed samples, if deformed at low versus high temperature or low versus high fluid content.

A microstructure develops due to the effect of a number of concurrent processes. There is no switch between different processes at different conditions, but a change in relative rates at which the different processes affect the resulting microstructure. Combining numerical simulations, experimental data and observations in natural examples, the relationship of microstructures and conditions during their formation can be established. Once this is done, the geologist can interpret observed microstructures in terms of the interaction and dominance of different processes and with this understanding derive conditions of deformation. The latter use will be significantly enhanced by extending the numerical simulation presented here to multiscale modelling taking into account local variations in dislocation densities and types of dislocations that are generated, removed and moved within a grain and subgrain.

Symbols used in Chap. 4.2

E_{sg}	Threshold dislocation energy density to start subgrain formation [J m-3]
E_{Γ}	Surface energy of high angle grain boundary [J m^{-2}]
E_{ρ}	Energy of dislocation [J m^{-1}]
v	Mineral-specific parameter [Pa s m]
η, η_0	Viscosity; base viscosity at zero dislocation density [Pa s]
M_{gb}	Grain boundary mobility [m^3 J^{-1} s^{-1}]
$R_{recovery}$	Recovery factor
ρ, ρ_{scale}	Dislocation density [m m^{-3}], Dislocation density scaling factor
τ_{split}	Energy threshold for splitting of a polygon [J m^{-3}]
τ_{nucl}, τ_0	Energy threshold for nucletion [J K m^{-3}];
W	Work done to deform a polygon [J m^{-3}]

4.3 Localisation of deformation

Mark W. Jessell, Esteban Siebert, Paul D. Bons, Lynn Evans
and Sandra Piazolo

4.3.1 Introduction

The localization of deformation in geological materials is a widely studied phenomenon at all scales, from the formation of deformation bands within single grains, up to crustal-scale shear zones (White et al., 1980; Poirier 1980; Hobbs et al., 1986). There is widespread evidence (in the form of faults, shear zones, fold hinges, etc.) that localization plays a major role in determining the mechanical response of rocks, and results in tectonically and economically important structures. At each scale, and sometimes for each material, there is considerable speculation as to which processes dominate the observed localization behaviour. In the mantle, localization of deformation is necessarily much harder to document. It may, however, play a major role in controlling continental and mantle dynamics and the observable seismic anisotropies (Ellis et al., 2001; McNamara et al., 2001). The purpose of this section is to demonstrate a simple simulation scheme capable of investigating spatial and temporal patterns of instantaneous and cumulative localization of deformation at the grain scale, using a two-dimensional numerical modelling scheme.

4.3.2 Implementation in *Elle*

We simulate a simple system in which three processes are active. We stress that these are not supposed to accurately represent the detailed behaviour of any single deformation process, but rather act as proxies for three classes of processes, namely a grain-size and strain dependant rheology, a process that increases the grain size, and a process that reduces the grain size (see Jessell et al. (2005) for a complete set of the experimental parameters). The three processes are:

1. We use the non-linear viscous Finite Element code *Basil* (Barr and Houseman 1996; see Chap. 3.8) to calculate the change in shape resulting from the application of a uniform translation of the upper surface of the model with respect to the lower surface, thus simulating bulk dextral simple shear. (Fig. 4.3.1b)

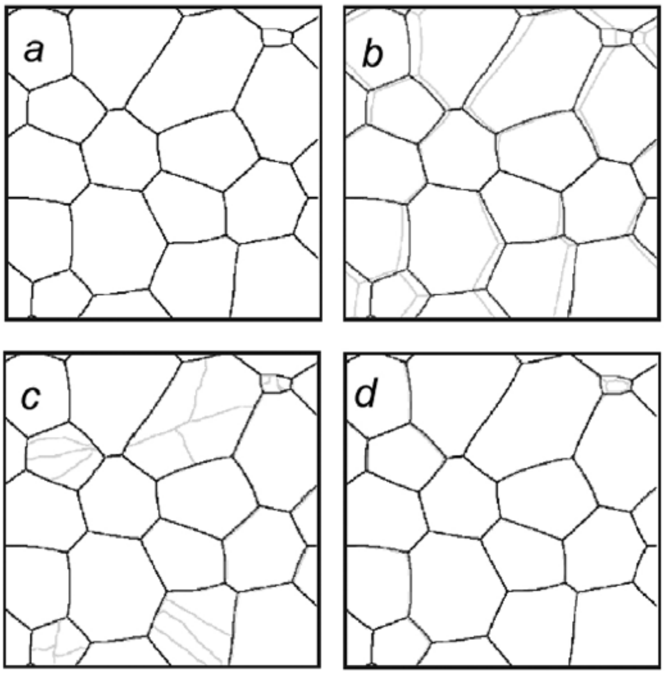


Fig. 4.3.1 The effect of individually applying the three processes to the same initial microstructure. **(a)** The starting microstructure, showing a foam texture with smoothly curved grain boundaries (individual nodes not shown). **(b)** Grey grain boundaries show the effect of deforming this microstructure with the Finite Element code *Basil*, to a bulk dextral simple shear of 0.05, the deformation applied during one time step of the experiments described in this paper. The model imposes wrapping vertical boundaries, so with further strain the small grain in the top-right of the model would re-appear on the top-left hand side. **(c)** Grey grain boundaries show the effect of several steps of the grain size reduction process, resulting in the production of several new high angle boundaries. **(d)** Grey grain boundaries show the effect of several steps of grain boundary migration, driven by boundary curvature reduction (commonly known as grain growth). The grain in the top right hand corner is visibly smaller than in the starting microstructure, and with further application of this process on its own, this grain would disappear (thus increasing the average grain size of the whole system)

2. There are several processes that could result in a decrease in grain size in a rock including fracturing, twinning, kinking, metamorphic re-equilibration and rotation recrystallisation, all of which are replaced in these experiments by a single process that inserts an approximately straight high angle grain boundary in a grain (Fig. 4.3.1c and see Chap. 3.4).

3. We have used a simple Front-Tracking algorithm which calculates the new positions of the boundary nodes resulting from a minimization of local grain boundary curvature to describe a generic grain size increase process (Fig. 4.3.1d and see Chap. 3.6). Although this type of algorithm quite well describes normal grain growth (Maurice 2001) it is obviously a simplification of the more general case during deformation when intercrystalline defect energy contrasts and/or surface energy contrasts may provide a larger and more complexly distributed driving force (Urai et al., 1986; Bons et al., 2001).

A single experiment consists of defining a starting microstructure, then cycling this microstructure through the three processes, so that the microstructure is slightly modified by each process in turn. In this sequence of experiments we chose to change only two parameters. The first of the two parameters is the stress exponent (*n* in Eq. 3.8.6) of the grain size dependent flow law, which could have values of 1 or 3. The second variable is the activation or non-activation of the Grain Size Reduction and Grain Size Increase processes. We thus define four Experiments A, B, C and D (Table 4.3.1).

Table 4.3.1 Characteristics of the four experiments described in this section

	Stress exponent = 1	Stress exponent = 3
Without grain-size modifying processes	**A**	**B**
With grain-size modifying processes	**C**	**D**

4.3.3 Experimental Results

These four simple experiments demonstrate a range of behaviours (Fig. 4.3.2), a monotonic evolution for Experiments A and B versus a more complex path for Experiments C & D. As one might expect, in these experiments, the more non-linear the system, either in terms of the rheology or the process coupling, the more complex the behaviour.

The localization behaviour observed for experiment D is time dependent, with a slow progression towards a broad zone of high strain (Fig. 4.3.3). In contrast the location of the zone of instantaneous high strain rate is narrower and varies more quickly. These instantaneous high strain rate zones are mostly associated with either grain boundaries (as demonstrated in Jessell et al., 2005) or trains of fine-grained material. As these boundaries migrate, and as new high angle boundaries form, these zones of localization shift accordingly.

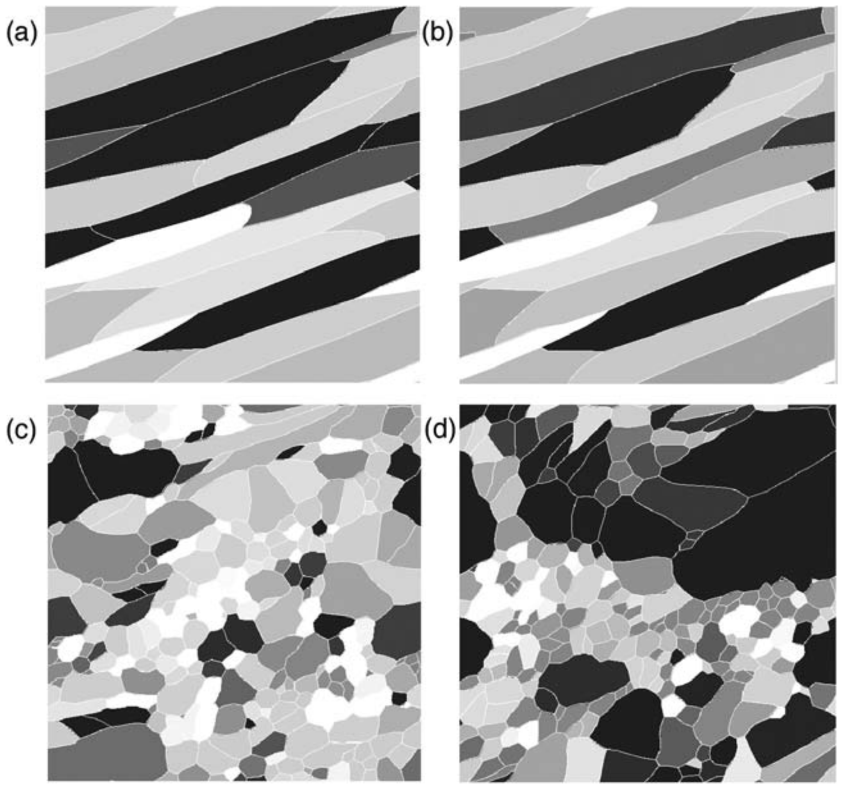


Fig. 4.3.2 Maps of the average first principal strain rate for each grain during time step 50 for each of the four experiments. The shading in each map is proportional to grain averaged strain rate, overlaid by the grain boundary network as white lines. The same look up table is used for all four maps. **(a)** Experiment A, showing a relatively homogeneous strain rate distribution, with a concentration of deformation in the small grain in the top-right. **(b)** Experiment B, showing a relatively homogeneous strain rate distribution, but with a concentration of deformation in the small grain in the top-right. **(c)** Experiment C, showing several subtle narrow horizontal zones of higher strain rate, which coincide with zones of finer grained material. **(d)** Experiment D, showing a broad sub-horizontal zone of high strain rate, which is in turn divided into narrow intense zones of deformation. These narrow zones coincide with zones of finer grained material. Bright pixels reflect high strain rates, dark pixels reflect low strain rates. EXPERIMENT 17

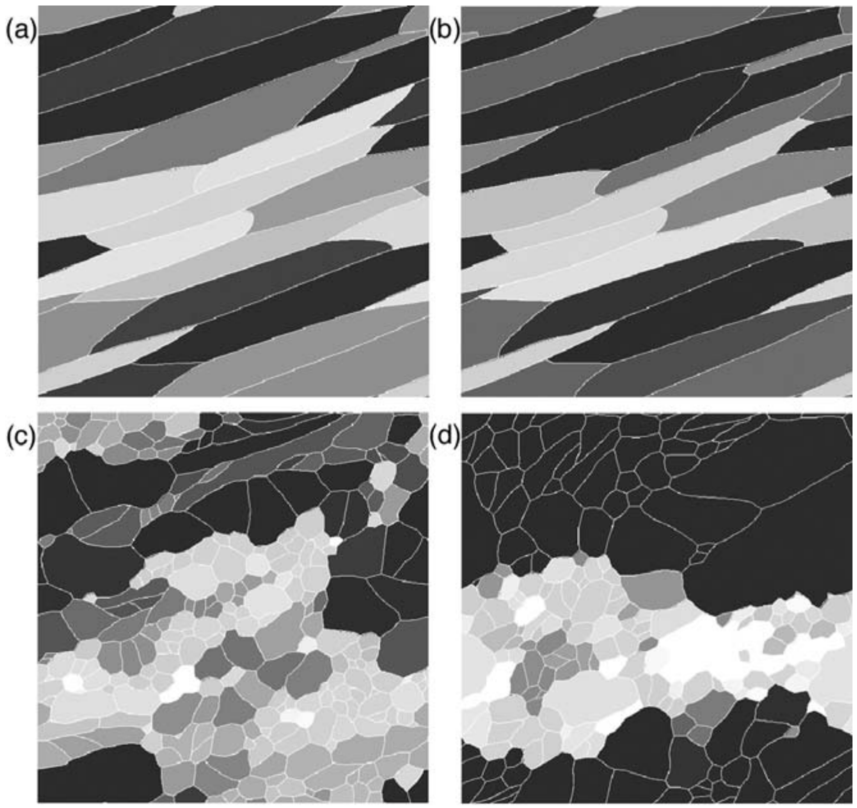


Fig. 4.3.3 Maps of the average finite first principal strain for each grain during time step 50 for each of the four experiments. The shading in each map is proportional to accumulated strain, overlaid by the grain boundary network as white lines. The same look up table is used for all four maps. **(a)** Experiment A, showing a relatively homogeneous strain distribution, with a concentration of deformation in the small grain in the top-right. **(b)** Experiment B, showing a relatively homogeneous strain distribution, but with a concentration of deformation in the small grain in the top-right. **(c)** Experiment C, showing a broad zone of higher strain. **(d)** Experiment D, showing a broad sub-horizontal zone of high strain. Notice that the high strain rate zones in Fig. 4.3.2d are not all high finite strain zones in this figure, and that the zone of bulk finite strain is much more clearly defined that the equivalent instantaneous high strain rate zone. Bright pixels reflect high bulk strain, dark pixels reflect low bulk strain. EXPERIMENT 17

4.3.4 Conclusions

1. Numerical simulations of coupled mechanical and microstructural evolution during dynamic recrystallisation display two length and time scales of deformation localization behaviour. There is no significant large-scale localization of deformation unless the grain size modifying processes are active and there is a non-linear relationship between grain size and rheology.
2. The intense zone of localization is neither temporally nor spatially stable, but leaves behind a broader zone of high finite strain.

4.4 Reactions and fracturing

Daniel Koehn

4.4.1 Overview of phenomenon

Reactions in the Earth are often associated with the development of fractures. Most of these reaction-related fractures form due to a volume reduction or volume increase during the reaction. Fractures often enhance the permeability of a rock so that fluids that may speed up reactions can penetrate into otherwise closed systems (Jamtveit et al., 2000). This may lead to a feedback where infiltrating fluids lead to further reaction with volume change, which will then initiate new fractures, enhance permeability and so on. In addition to mineral reactions, volume change may also be initiated during cooling or loss of water (Fig. 4.4.1).

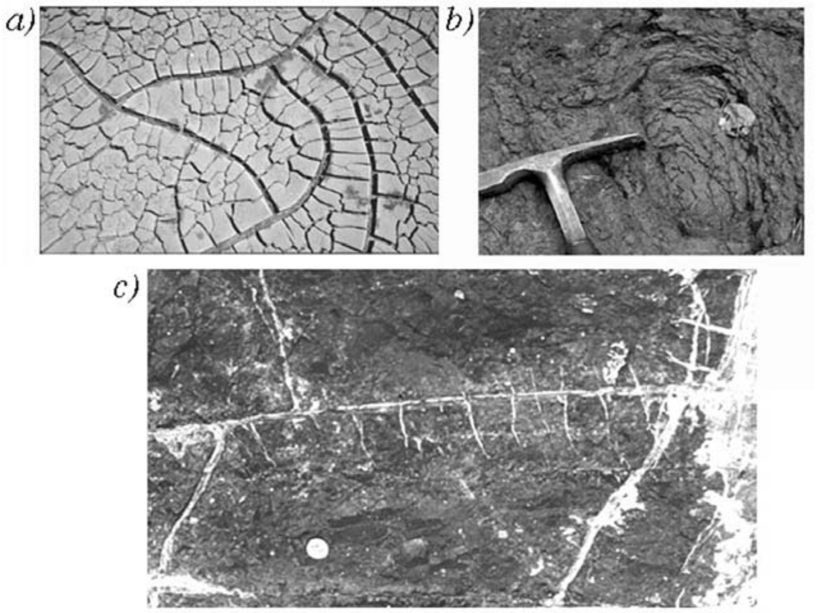


Fig. 4.4.1 (a) Mud cracks in dehydrating sediments. Picture from Namibia, courtesy Cees W. Passchier. **(b)** Exfoliation cleavage in a basalt, near Kirn, Germany. **(c)** Extension veins around a reaction vein in retrograde Ultramafics from an ophiolite in the Liguride Units near Bonasola, Italy

A natural example of a mineral reaction that leads to volume change and fracturing is the reaction of stishovite to normal quartz. If the stishovite (the high pressure phase of quartz) forms inclusions in a garnet and the pressure is then released (for example during uplift) the stishovite may go through a phase-change to normal quartz with an increase in volume. This increase can lead to fracturing of the surrounding garnet. Another example of the combination of reactions and fracturing is the development of stylolites (dissolution seams, see Chapt. 3.12) and associated fractures or veins that form perpendicular to the stylolite. Natural examples of fracturing during loss of water or cooling processes are shrinkage cracks in drying muds or exfoliation cleavage in basalts (Fig. 4.4.1).

4.4.2 Implementation in *Elle*

In order to implement the combination of reactions that dissolve material or change the volume of phases in *Elle* we use two different approaches. The basis for the simulations is the *Elle Latte* fracture code described in detail in Chap. 3.11. It handles the initiation and propagation of fracture once the local breaking strength is exceeded. The reactions or volume change processes are handled by the following rules: in the case of expanding inclusions and shrinkage patterns we change the volume of particles using a linear rate law independent of local stress or other factors. Each given time step the volume of a number of particles within certain grains or the volume of each particle in the simulation is changed. Afterwards the stress field is determined and bonds may break. Once all possible bonds are broken a new volume change is induced. The second approach treats the dissolution of material into a fluid along an interface coupled with fracturing of the solid. The dissolution routine is identical to the one described in Chap. 3.12. Each time a particle is dissolved or the lattice is stressed a new stress distribution is determined and bonds can possibly break. Examples of three processes (expanding inclusions, shrinkage patterns and stylolites in combination with fractures) are given below.

4.4.3 Experimental runs

Example 1: Expanding inclusions

In this example single grains expand and develop fracture patterns around them. Two examples are shown in Fig. 4.4.2. The underlying microstructure is a polycrystal where grain-boundaries break more easily (with half the failure limit). One or several grains are set to expand. The grains expand by an increase in the radius of all particles within the specified grains. During each time step the specified grains expand by a given amount, the elastic stresses are calculated, and springs may fracture. The fracture pattern in this case grows subcritically and depends on the expansion rate of the expanding grains.

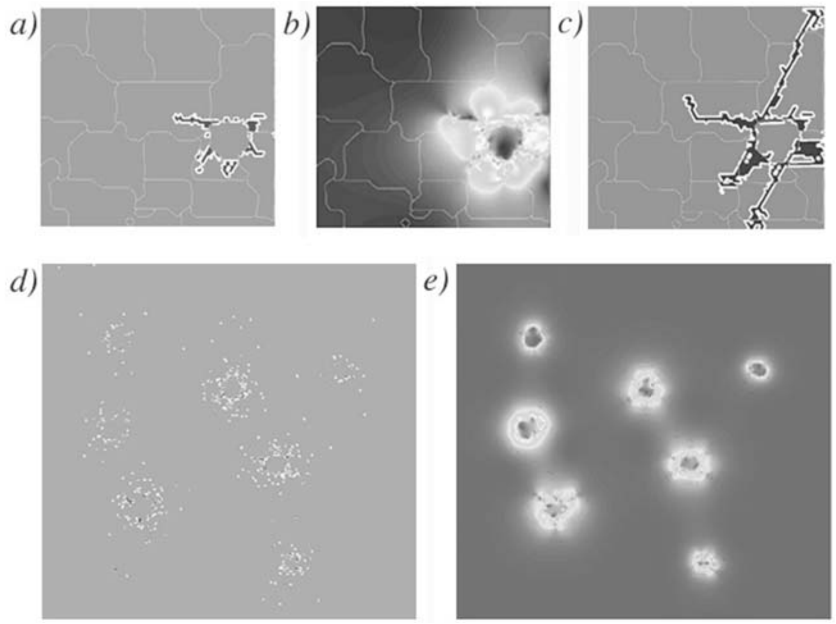


Fig. 4.4.2 Fracture patterns around expanding grains. **(a)** Fracture pattern around an expanding grain, and **(b)** corresponding mean stress field. Dark is low stress and light is high stress. **(c)** large fractures around the expanding grain reach the boundaries of the model. Fractures nucleate at corners of the expanding grain. **(d)** Fracture pattern, and **(e)** mean stress around seven expanding grains. Resolution in the models is 100 particles (a-c) and 400 particles across (d-e). EXPERIMENT 18

Figures 4.4.2a-c show the fracture pattern and stress field in a simulation that is 100 particles wide with a linear distribution of breaking strengths of springs. The initial grains are shown in the background. Figure 4.4.2a shows the fracture pattern around the expanding grain after 10 time steps. Fractures develop around the expanding grain and grow into the matrix from corners where stresses concentrate. The corresponding stress field is shown in Fig. 4.4.2b where dark shades represent lower stresses and light shades higher stresses. The stress field has a flower structure and is strongly influenced by the developing fractures. The fractures partly separate stressed from unstressed regions and this gradient is probably responsible for the development of the fractures themselves. Figure 4.4.2c shows the fracture pattern after 20 time steps. The fractures have now reached the border of the model. Note that due to the low resolution of the model the underlying lattice is now visible, which means that it partly controls the direction of the fractures. Figures 4.4.2d-e show the fractures and mean stress around seven expanding grains in a model that is 400 particles across. The fracture pattern is more dispersed since the underlying noise in breaking strengths is drawn from a Gaussian distribution and has a much wider variation than that shown in Figs. 4.4.2a-c. The stress fields are similar to the low-resolution model. Expanding grains where the fractures are not yet well developed show a circular stress field, whereas the stress around grains where larger fractures have developed show a more flower like geometry similar to the stress field in Fig. 4.4.2b.

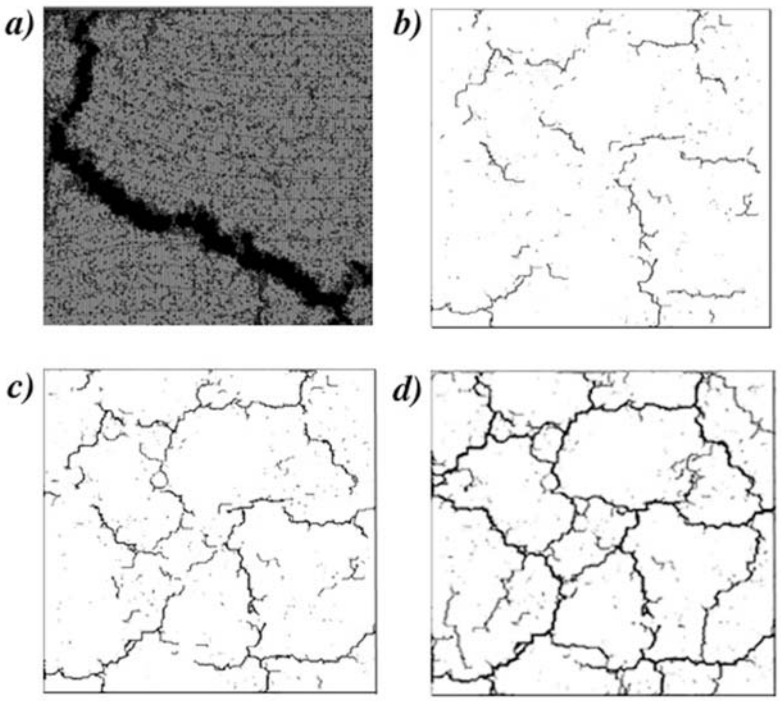


Fig. 4.4.3 Fracture patterns that develop due to a reduction in volume. Particles in the model shrink by a constant amount. **(a)** large shrinkage crack in a 200 particle wide lattice. Progressive shrinkage patterns in a 800 particle wide lattice after **(a)** 100, **(b)** 200 and **(c)** 300 time steps. The fractures are interconnected in (d) so that the model is fragmented and the fractures open to accommodate any further area reduction. EXPERIMENT 19

Example 2: Shrinkage patterns

In this example all particles in the model shrink. This simulation mimics for example the development of mud-cracks in shrinking sediments. An initial Gaussian distribution of breaking strengths is used. Grain boundaries have lower breaking strengths and thus fracture easier. Figure 4.4.3 shows fracture patterns that develop during the shrinkage process. Figure 4.4.3 shows a 200 particle wide lattice with a large curved fracture that has opened significantly. Fractures are shown in dark shades. The fractures show no preferred orientation with respect to the boundaries of the model or underlying lattice directions. Figures 4.4.3b-d show the development of fracture in a shrinking 800 particle wide model. Fractures nucleate at relatively random locations but start to grow and organize into a well-defined network with a regular spacing. The whole network is finished in Fig. 4.4.3d where darker colours indicate fractures that are opening to accommodate the strain due to the shrinking process. Polygons that are surrounded by fractures have four to six edges.

Example 3: Stylolites and cracks

The two following examples show the development of two stylolites and cracks (Fig. 4.4.4). The initial setup is similar to the example of stylolite development given in Chap. 3.12, where a horizontal row of particles is dissolved at the beginning. The surrounding particles are defined as interfaces and can react with the fluid. Depending on the speed of the reaction, stresses may build up in the solid and fractures develop.

In the example shown in Figs. 4.4.4a-b the model is compacted vertically and stretched by a much smaller amount horizontally. Dissolution is slow so that stresses start to build up in the solid. This leads to the development of conjugate shear fractures around the very slowly growing stylolite. In the example shown in Figs. 4.4.4c-f the compaction is again vertical but the box is also extended by a significant amount horizontally. Dissolution is faster than in the first example so that the dissolution of the solid can relax the compressive stress and the stylolite grows and develops a roughness. However the growing stylolite cannot relax tensile stresses due to the horizontal shortening component so that extensional fractures develop that are oriented perpendicular to the stylolite. These fractures develop a well-defined spacing.

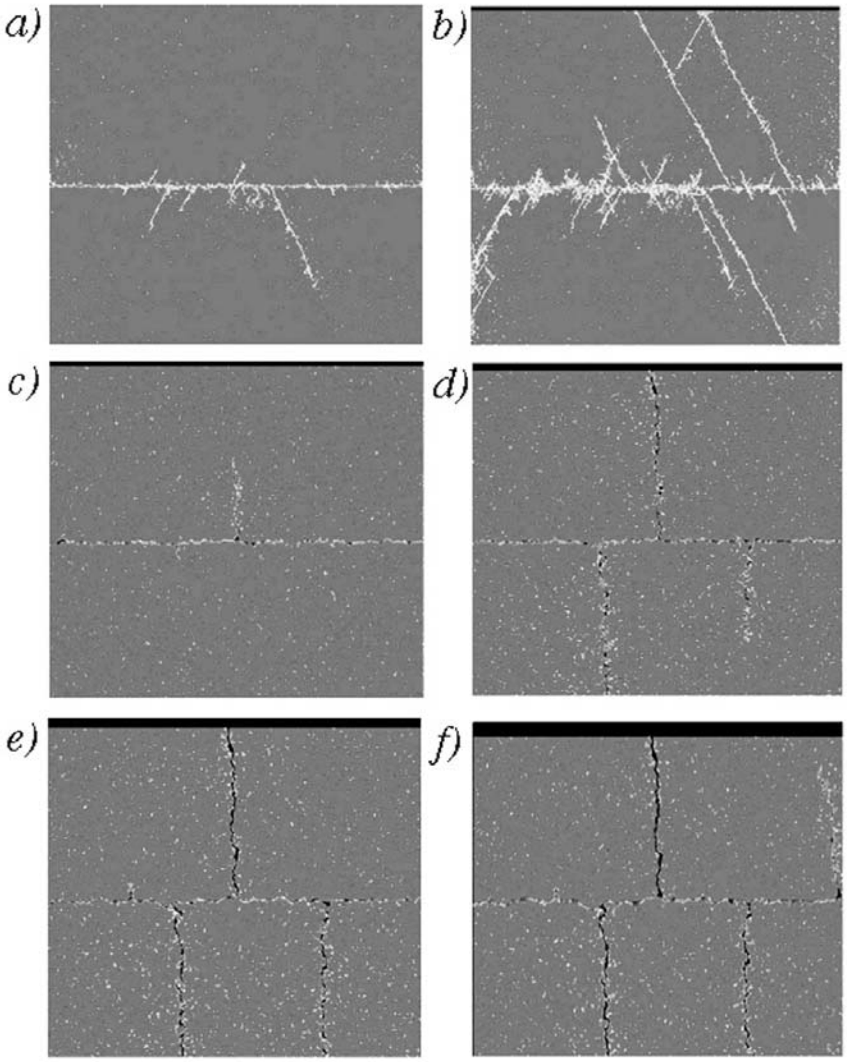


Fig. 4.4.4 Stylolite and fracture development. **(a-b)** Two stages of the development of shear fractures around a stylolite where dissolution is so slow that high differential stresses can build up. These induce the development of conjugate shear fractures. **(c-f)** Four stages in the development of extension fractures perpendicular to the stylolite. In this example the dissolution at the stylolite reduces the compressive stress but cannot reduce the extension component parallel to the dissolving interface. Therefore tensile stresses build up that lead to the development of extension fractures with a well-defined spacing

4.5 Combined phase transition and heat diffusion

Till Sachau and Daniel Koehn

4.5.1 Introduction

The algorithms to simulate olivine-spinel phase transformations and to simulate heat-conduction are included in the *Latte* process of *Elle*, and are already described in Chap. 3.13. We discuss a combination of these processes below.

Most phase-transitions involve the release or consumption of heat-energy. At the same time the transition depends on the local temperature. Therefore the role of the latent heat release during a transition on the rate of the reaction itself becomes important and leads to a certain self-control of the transformation.

The influence of temperature on reaction-rates is frequently discussed in the context of the proposed rapid olivine-spinel transition in subducted oceanic lithosphere. This transition is one important idea to explain deep-seated earthquakes in the upper mantle along the Wadati-Benioff zone. The assumption is that relatively cool former lithosphere enters the stability-field of spinel, and will thus lead to a rapid transition and an equally rapid loss of volume of the material.

In the context of this chapter a different approach is used: by applying the idea of a temperature-driven reaction directly on evolving grain-surfaces of newly-formed spinel-grains. What can be expected is a certain smoothing of grain-surfaces compared to algorithms that do not involve latent-heat release, since the heat halo around transformed material will increase local reaction-rates. As a side effect, the reaction is now controlled by a time-dependent parameter, the heat flow, and not only by time-independent elastic and surface energies. In addition to the direct influence on reaction-rates, the temperature also influences several parameters of a material, e.g. the viscosity and the volume, which can be included into the overall algorithm. In the current state of the program only the change of the molar volume is taken into account.

Apparently the influence of heat energy on surface structures of a new phase depends on the coordination of the velocity of heat-conduction and the rate of the reaction. If thermal equilibrium is reached prior to a significant continuation of the reaction the effect will be minor. For this reason the activation energy of the reaction, which controls the velocity of the transition, plays an important role in simulations. In case of the olivine-spinel reaction previous work offers a relatively broad band of supposed activation energies, ranging from 300 to 450 kJ.

4.5.2 Implementation

The algorithms involved are already discussed in the chapters of the particular processes (Chaps. 3.11 to 3.13) and shall not be repeated here. The general law for the release of latent heat energy (ΔQ), depending on the volume change ΔV is

$$\Delta Q = \Delta H_{T,p_{equ}} + \int \Delta V(P',T)dP'. \tag{4.5.1}$$

In the current context, the above formula is ignored and Q just replaced by H, the difference in enthalpy between the two phases, which is assumed to be 39050 ± 2620 J/mol (Akaogie 1989). The temperature difference before and after the transition is calculated on the basis of the molar heat capacity (C_m, J/molK)

$$C_m = a + bT^{-0.5} + cT^{-2} + dT^{-3}, \tag{4.5.2}$$

$$\Delta Q = NC_m(T_1 - T_2), \tag{4.5.3}$$

where a, b, c and d are parameters given for instance by Akaogie et al. (1989), N is the number of moles involved and T_1, T_2 are the temperature before and after a transition.

4.5.3 Example runs

Two experiments of growing spinel grains (black) and the associated temperature fields are shown in Fig. 4.5.1. The two experiments lead to a complex pattern of the distribution of heat-energy, due to the fact that many overlapping small-scale fields exist (in particular Fig. 4.5.1b). The difference between the two experiments is the activation energy, which was either 450 kJ (Fig. 4.5.1a-b), or 380 kJ (Fig. 4.5.1c-d).

As a result, the evolving grain boundaries of the spinel phase look less uneven in Fig. 4.5.1c due to a narrow temperature halo in comparison to Fig. 4.5.1a. As a second observation, particularly Fig. 4.5.1d displays an increasing narrowing of the temperature halo with ongoing transformation, caused by an acceleration of the reaction rate as a result of rising differential stresses.

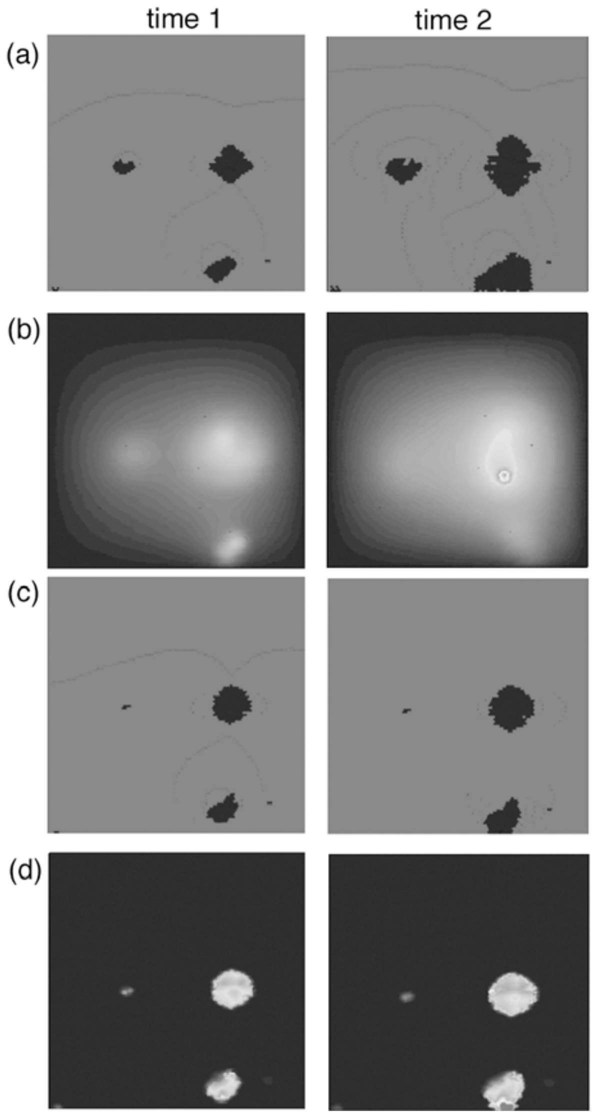


Fig. 4.5.1 (a) Two stages of growth of spinel (black) within olivine (grey). Time 1 is after 35 and time 2 after 68 steps. **(b)** Associated temperature distribution, with brightest area at 1025 K. The high activation energy (450 kJ) leads to a sluggish phase transition. **(c)** Same phase transition with an activation energy of 380 kJ, which increases the reaction rate. **(d)** Associated temperature distribution with brightest area now at 1200 K, despite constant heat loss at the boundaries that are held at the same constant temperature of 1000 K as in (b)

Symbols used in Chap. 4.5

a,b,c,d	Parameters used in Eq. 4.5.2
C_m	Molar heat capacity [Jmol^{-1}K^{-1}]
ΔH	Enthalpy-difference between two phases [J mol-^{1}]
N	Number of moles
P'	Derivative of pressure [Pa]
ΔQ	Heat energy [J mol-^{1}]
T_1, T_2	Temperature before and after a phase transition [K]
ΔV	Volume difference between two phases [m^3 mol^{-1}]

4.6 Visco-elastic and brittle deformation

Daniel Koehn and Till Sachau

4.6.1 Overview of phenomenon

Rocks that are deformed in the upper part of the Earth's crust may behave in a visco-elastic or brittle manner depending on strain rates, temperature and material properties. A rock that behaves in a brittle manner fractures once a failure strength is reached, whereas a rock that behaves in a viscous manner flows like a fluid as a function of time. Elastic strain as a function of elastic material parameters is recoverable. Whether or not a visco-elastic rock flows or reaches failure and fractures depends on the interplay between build up of elastic stress and release of this stress through ductile flow of the material. This behaviour is obvious near the brittle-ductile transition but can be found throughout the crust in rocks that contain minerals or layers with different material constants. An example is given in Fig. 4.6.1 where a more brittle layer in a mainly ductile matrix is both fractured and folded.

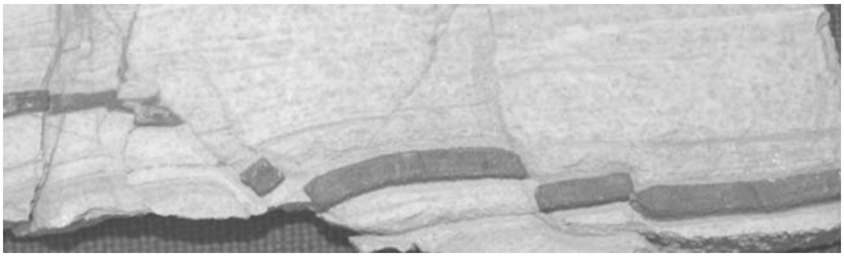


Fig. 4.6.1 Visco-elastic deformation in a rock sample from Namibia, courtesy Cees W. Passchier. Width of view is about 20 cm. The dark layer behaves in a more viscous manner than the white matrix and can build up elastic stresses so that it fractures and breaks apart. Even though the dark layer is fractured it still behaves in a viscous manner and is folded. Fractures also extend into the white matrix

Other examples for visco-elastic deformation include most boudin-types where layers are fractured and then neck during subsequent viscous behaviour. Another example are porphyroclasts that are fractured internally within a matrix that flows.

4.6.2 Implementation in *Elle*

The implementation in *Elle* is based on the spring-network *Latte* described in Chapt. 3.11. When the spring-network deforms visco-elastically the springs will change their length depending on the stress. This operation reduces the deviatoric stresses on a node, but doesn't affect the mean stress. In contrast to an elastic relaxation, which occurs immediately, viscous deformation depends not only on the stress but also on time. Therefore we calculate the viscous deformation after an applied elastic deformation for a given time.

As a basic assumption we extend the discrete-element lattice from a network of circles or spheres to a network of ellipses, where the ellipses are circular in the initial, undeformed state. Every ellipse is defined by its two main axes, which will rotate and change length due to the actual stress-tensor in direction of (in case of normal stresses) or perpendicular to (in case of shear stress) the axes. This will lead to a rotation and deformation of the particles that define the spring network.

The normal stress (σ_n) and the shear-stress (σ_s) can be calculated via:

$$\sigma_n = \frac{1}{2}(\sigma_{xx} + \sigma_{yy}) + \frac{1}{2}(\sigma_{xx} - \sigma_{yy})\cos(2\phi) + \sigma_{xy}\cos(2\phi) \tag{4.6.1}$$

and

$$\tau = \frac{1}{2}(\sigma_{xx} - \sigma_{yy})\sin(2\phi) + \sigma_{xy}\cos(2\phi), \tag{4.6.2}$$

where σ_{xx} and σ_{yy} are the stress in the x- and y-direction and ϕ the angle between the x-axis and the long axis of the ellipse. Subtraction of the mean stress from σ_n gives the deviatoric normal stress.

During the elastic relaxation spring-lengths and repulsion parameters for a particular node are constantly adapted to the shape and orientation of the superimposed particle. In a following version the spring-model may be replaced by beams in order to include elastic micro-rotations as well.

A Newtonian law of viscosity was used, where the differential stress ($\Delta\sigma$) is linearly related to the strain rate ($\dot{\varepsilon}$) through the viscosity (η): $\sigma = 2\eta\dot{\varepsilon}$. Integration with respect to time, assuming that the differential stress remains constant, allows the calculation of the strain increment $\Delta\varepsilon$ after a time increment Δt in any desired direction:

$$\frac{\sigma\Delta t}{2\eta} = \Delta\varepsilon. \tag{4.6.3}$$

By using differential stresses, the area of the particle remains constant during the viscous deformation step so that shear stresses are released and the pressure remains. Of course this formulation can only be an approach to the real strain increment, since the stress itself cannot be regarded as a time-independent constant. For the high viscosities of geomaterials, and for sufficiently small time steps, this can be neglected. If needed, the formulation can be tested or improved with an adapted Romberg-scheme algorithm, which allows the extrapolation to $\Delta t \rightarrow 0$.

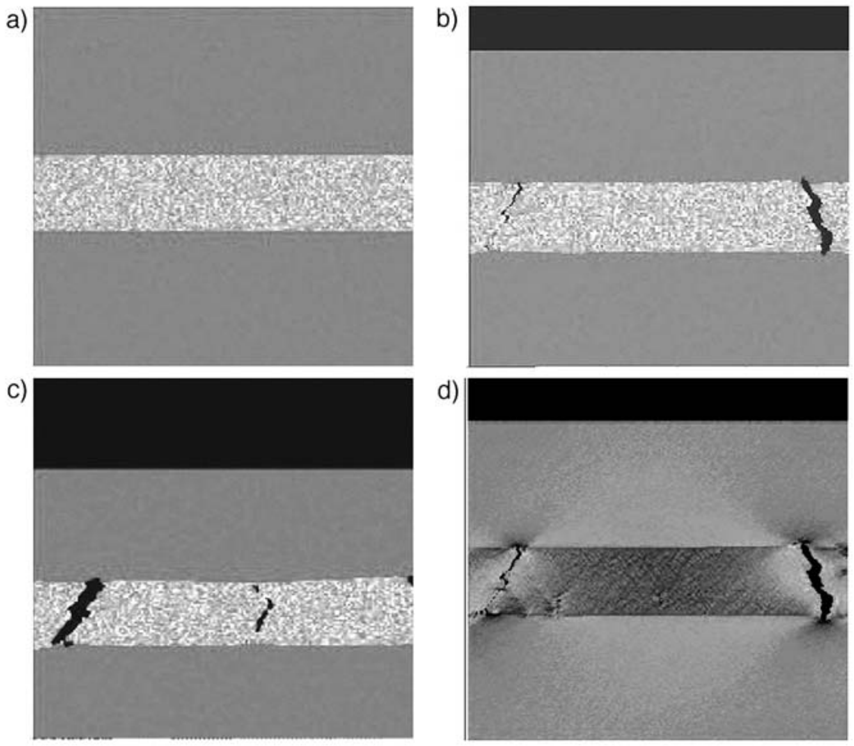


Fig. 4.6.2 Fracturing of a layer that has a viscosity that is 10 times that of the matrix. The harder layer develops two to three distinct open fractures. Images showing the Young's moduli of particles after **(a)** 0, **(b)** 100, and **(c)** 200 time steps. **(d)** Differential-stress state after 100 time steps, showing stress concentration (dark) at fracture tips and stress release next to these. EXPERIMENT 20

4.6.3 Experimental runs

Simulation 1: Fracturing and necking of boudins

In the first simulation, we place a horizontal layer within a weaker matrix. All particles have a Gaussian distribution of Young's moduli. The layer in the centre has a Young's modulus that is on average 10 times that of the matrix and a viscosity that is also 10 times that of the matrix. Breaking strengths of all springs have a linear distribution. The initial lattice has a resolution of 100 particles in the x-direction. Deformation is pure shear with a vertical compression and a horizontal extension.

Figure 4.6.2a shows the initial setup before deformation. Grey scales show the Young's moduli of single particles. The dark particles have a lower Young's modulus than brighter particles. Figures 4.6.2b-c show the layer after 100 and 200 steps. Two to three distinct fractures develop in the layer. The layer necks around the fractures. Figure 4.6.2d shows the differential stress in the layer and adjacent matrix. Stresses concentrate at fracture tips (dark) and are released next to the fracture (bright).

Simulation 2: Rotation of boudins

The second simulation (Fig. 4.6.3) shows the deformation of a layer in a visco-elastic matrix where the layer has a viscosity that is 5 times that of the matrix. The layer with the higher viscosity fractures and develops boudins that show some necking, rotation and folding. The initial shape of the deformed box was a square, and deformation is pure shear with a vertical compression. The hard layer fractures relatively early during the experiment because it is fixed at the right and left hand side of the deforming box and concentrates tensile stresses. After fracturing stresses are concentrated at the contact of the fractured hard layer and the weaker matrix (see also fig. 4.6.2d). This stress concentration leads to necking of the hard layer near fractures. If the fractures are not horizontal in the hard layer but curve or have an angle towards the main compressive stress and the hard layer is folded.

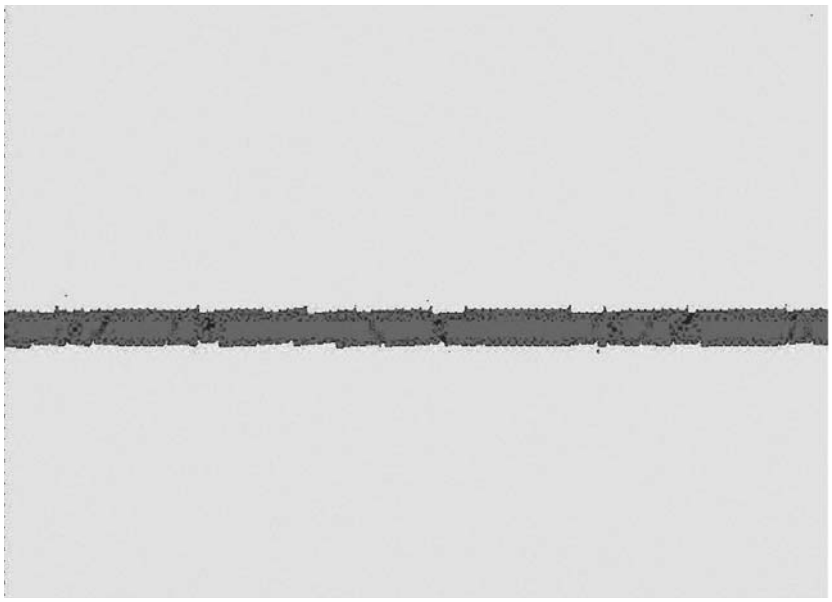


Fig. 4.6.3 Fracturing of a layer that has a viscosity that is 5 times that of the matrix. The matrix flows whereas the layer fractures, shows rotation of small boudins and folding. The initial box shape before deformation was square

Simulation 3: Fracturing of a hard grain in a viscous matrix

Figure 4.6.4 shows a grain that has a viscosity that is 10 times that of the surrounding matrix. The deformation is pure shear with vertical compression and horizontal extension. The model was initially square. Brightness represent the distribution of the mean stress where bright represents a high compressive stress and dark a low compressive or tensile stress. The more viscous grain fractures internally whereas the matrix flows around it without fracturing.

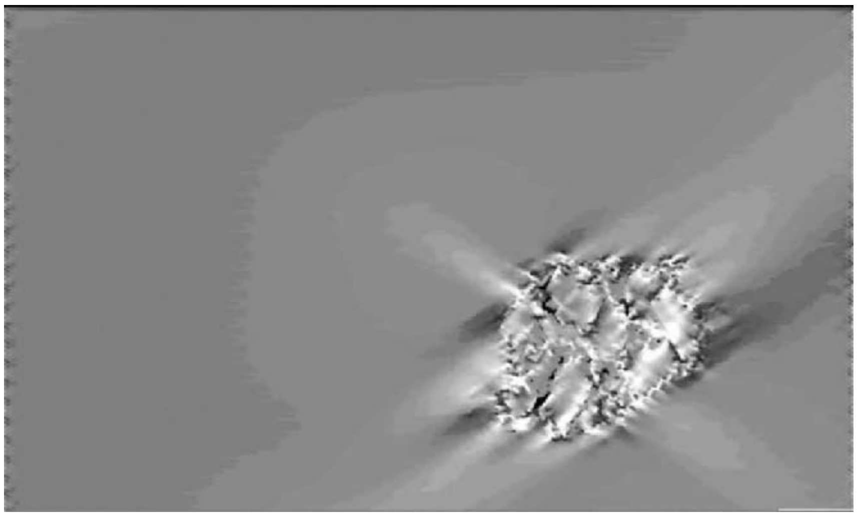


Fig. 4.6.4 Deformation of a more viscous grain (10x) in a less viscous matrix. The matrix flows around the grain whereas the grain itself deforms mainly by the development of internal fractures. Brightness represents the mean stress from compressive (dark) to extensional (bright). EXPERIMENT 20

Symbols used in Chap. 4.6

ε	Strain
$\dot{\varepsilon}$	Strain rate [sec^{-1}]
η	Viscosity [Pa s]
σ_N	Normal stress [Pa]
σ_s	Shear stress [Pa]
$\Delta\sigma$	Differential stress [Pa]
τ	Deviatoric stress tensor [Pa]
ϕ	Angle between x-axis and long axis of ellipse [deg]

4.7 Strain localisation and rigid-object kinematics

Scott E. Johnson

4.7.1 Introduction

The kinematic behaviour of rigid particles in flowing viscous fluids is a fundamental problem in various branches of science and mathematics. In the Earth sciences, there are several primary areas of interest. For example, we are interested in how the bulk viscosity of flowing materials, such as magma, changes when rigid particles are introduced. Early investigations along these lines include that of Einstein (1906), in which he considered spherical particles. Einstein's work, and that of Roscoe (1952), form the basis of what we generally now know as the Roscoe-Einstein equation in which the viscosity of a fluid/particle system is expressed in terms of the fluid or particle fraction. Another area where this is important in the Earth sciences involves the rotational behaviour of rigid particles in deforming rocks. Jeffery (1922) published the seminal work on the topic, deriving the equations of motion for rigid triaxial ellipsoids in a homogeneous, isotropic, Newtonian viscous fluid. Jeffery's work has formed the basis of numerous theoretical, analogue and numerical investigations of rigid object kinematics in deforming rocks (e.g., Gay 1968; Rosenfeld 1970; Ghosh and Ramberg 1976; Willis 1977; Ferguson 1979; Fernandez et al., 1983; Passchier 1987; Fernandez and Laporte 1991; Ildefonse and Mancktelow 1993; Jezek 1994; Tikoff and Teysier 1994; Masuda et al., 1995; ten Brink and Passchier 1995; Arbaret et al., 2001; Marques and Coelho 2001, 2003; ten Grotenhuis et al., 2002; Mancktelow et al., 2002; Samanta et al., 2002; Stallard et al., 2002; Biermeier and Stüwe 2003; Ceriani et al., 2003; Ghosh et al., 2003; Schmidt and Podladchikov 2005).

Some observations in naturally deformed rocks are not well described by the equations of Jeffery (1922). For example, Pennacchioni et al. (2001), ten Grotenhuis et al. (2002), Mancktelow et al. (2002) and Ceriani et al. (2003) noted that elongate mineral grains in very-high strain rocks (mylonites) are typically oriented with long axes oblique to the mylonitic foliation, which at very high strains is assumed to track the flow plane. These objects appear to align in a stable position in the flow that is not predicted by Jeffery's equations, and which conflicts with analogue and numerical models of rigid objects in homogeneous viscous media (e.g. Ghosh and Ramberg 1976). Ildefonse and Mancktelow (1993), Mancktelow et al. (2002) and Ceriani et al. (2003) conducted analogue experiments in viscous media, but used soap to lubricate the surfaces of the rigid objects. Their results suggest that strain localization and slip at the object interface may provide an explanation for the oblique stable positions of elongate particles. Ten Grotenhuis et al. (2002) conducted analogue experiments of rigid objects in a Mohr Coulomb material (tapioca pearls), and their results suggest that, in addition to slip at the object interface, slip along discrete shear zones in the matrix away from the object may promote the development of oblique stable positions of elongate particles.

In addition to the above studies in mylonites, a large number of papers over the past 15 years have used inclusion-trail orientation data to argue that metamorphic porphyroblasts have undergone little or no rotation during deformation relative to one another and a fixed external reference frame (Fig. 4.7.1).

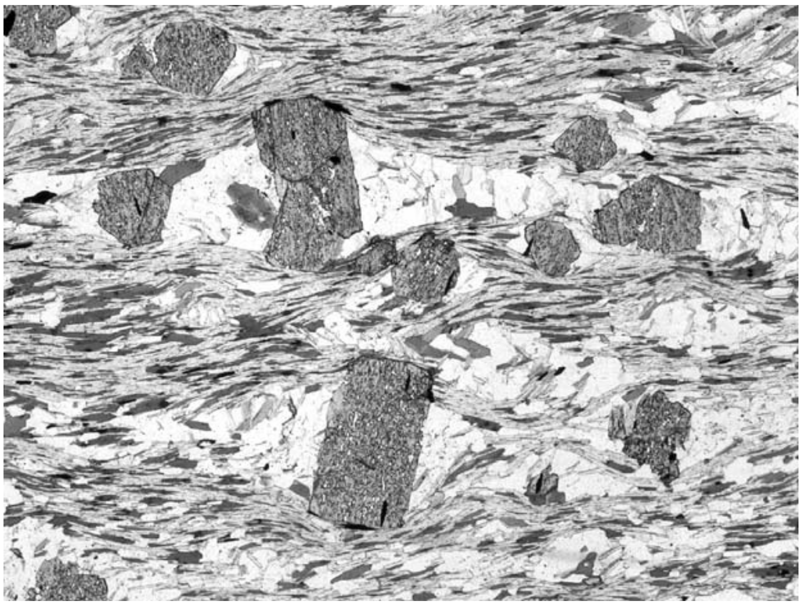


Fig. 4.7.1 Photomicrograph of staurolite porphyroblasts in a matrix of muscovite, biotite, quartz and plagioclase. The staurolite grains have trails of inclusions that represent a foliation present at the time of their growth. These trails are highly oblique to the matrix foliation, which raises the question of the kinematic behaviour of the porphyroblasts during deformation. Plane light, long axis of photomicrograph 5.9 mm

These studies have shown that the statistical peaks of the data remain consistently oriented over areas ranging in size from outcrop-scale folds (e.g. Steinhardt 1989; Jung et al., 1999) to tens or hundreds of square kilometres (e.g., Fyson 1980; Johnson 1992; Aerden 1995; Ilg and Karlstrom 2000). Such consistency in inclusion-trail orientation across large areas seems counter intuitive, given the highly heterogeneous distribution of strain in deforming orogens. Although these data are impressive, the spread in the data from individual samples is commonly 40 degrees or more (Johnson et al., 2006), and this has been attributed either to primary heterogeneity in foliation orientation prior to porphyroblast growth (e.g., Bell et al., 1992), or to differential rotation of porphyroblasts relative to one another during deformation (e.g., Passchier et al., 1992, Johnson et al., 2006). Johnson (1990) suggested that the boundaries of porphyroblasts in deforming mid-crustal rocks may be partially lined by phases that are relatively weak in shear, such as phyllosilicates and fluids, and that these phases may allow strain localization at the object interface leading to kinematic behaviour that is not fully predicted by the equations of motion derived by Jeffery (1922).

In this contribution, we show how numerical modelling can be used to simulate the effects of “slip” at the interface of a rigid particle and a viscous medium. Although we have results from a variety of object shapes and aspect ratios, we focus here on sub-circular objects owing to the ease with which the physics of their kinematic behaviour can be understood.

4.7.2 Numerical experiments

Description of the model

To establish and keep track of the evolving model geometry, we used the *Elle* microstructure simulation system (Jessell et al., 2001; Jessell and Bons 2002).

The physical model is shown in Fig. 4.7.2. A central, hard inclusion is intended to simulate a garnet porphyroblast. This grain is surrounded above and below by “caps” that are intended to simulate material that is relatively weak in shear, such as micas with or without thin fluid layers. The surrounding polygonal framework is intended to simulate a background matrix with a homogenous, isotropic viscosity structure. The matrix and central inclusion are assigned constant viscosities, whereas the viscosity of the caps was varied from one experiment to another (Fig. 4.7.2).

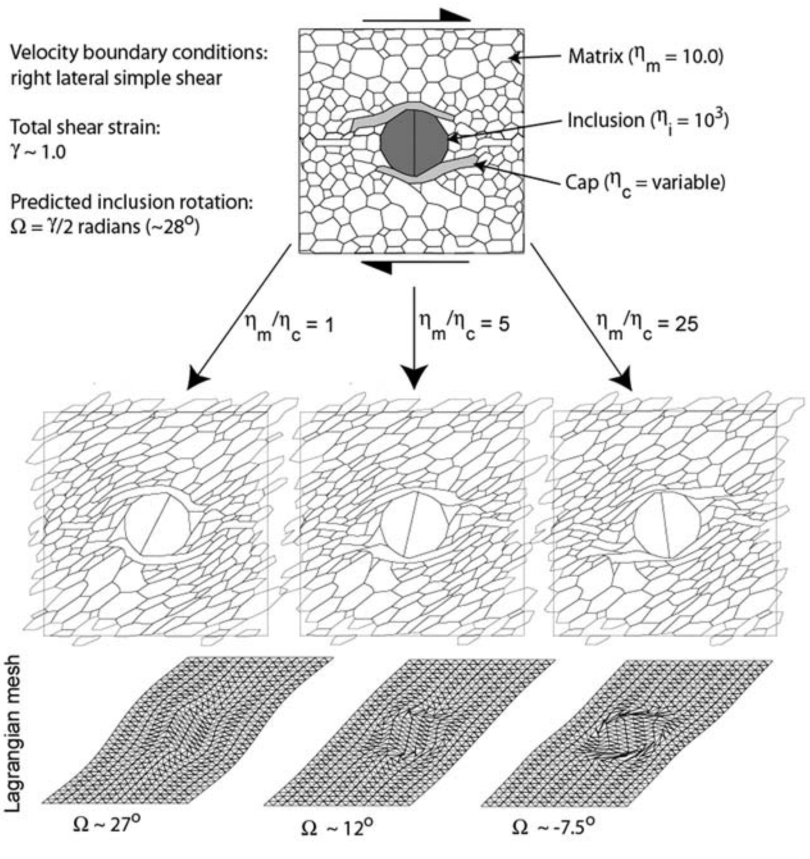

Velocity boundary conditions: right lateral simple shear
Total shear strain: γ ~ 1.0
Predicted inclusion rotation: Ω = γ/2 radians (~28°)
Matrix (η_m = 10.0)
Inclusion (η_i = 10³)
Cap (η_c = variable)
η_m/η_c = 1
η_m/η_c = 5
η_m/η_c = 25
Lagrangian mesh
Ω ~ 27°
Ω ~ 12°
Ω ~ -7.5°

Fig. 4.7.2 Diagram showing the starting physical model at top, the final state for three experiments with different cap viscosities, and the deformed Lagrangian grid for each experiment. All three experiments were conducted under right-lateral simple shear to a shear strain of 1.0. Under these conditions, an inclusion with axial ratio of 1.0 is expected to rotate at half the bulk shear strain rate giving a total rotation of 28° for a bulk shear strain of 1.0

To evaluate the effect of relatively low-viscosity caps above and below a porphyroblast, we deformed the starting geometry under constant velocity boundary conditions of right lateral simple shear to a total shear strain of 1.0. The vertical boundaries in the model are cyclic, so grains that disappear off the upper-right part of the model, for example, reappear at the upper-left. The mechanical equations were solved using *Basil*, a 2D velocity based formulation that employs a Finite Element solution for incompressible, linear or nonlinear viscous flow (Barr and Houseman 1996; see Chap. 3.8). The *Elle* boundary nodes are used for triangulating the Finite

Element grid, and each element responds to applied velocity or stress boundary conditions by the following constitutive relation (see Eq. 3.8.4):

$$\tau_{xy} = 2\eta\dot{\varepsilon}_{xy} = B\dot{E}^{\frac{1-n}{n}}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right), \tag{4.7.1}$$

where τ is the deviatoric stress, $\dot{\varepsilon}$ is the shear strain rate, and u and v are the velocity components in the x- and y-directions, respectively. The viscosity is given by η, which is defined as follows:

$$\eta = \frac{1}{2}B\dot{E}^{\frac{1-n}{n}}, \tag{4.7.2}$$

where $\dot{E}$ is the square root of the second invariant of the strain rate tensor (the “effective strain rate”). The stress exponent n determines the power-law relation between deviatoric shear stress and shear strain rate, and B is a pre-exponential material-dependent constant. For the purposes of this paper, n was set to unity, resulting in a linear viscous constitutive relation. It was already shown by Bons et al. (1997) that varying the matrix stress exponent (up to $n = 5$) has little effect on the rotation of a circular object.

4.7.3 Results

Figures 4.7.2 and 4.7.3 show the results of three experiments with viscosity ratios of 1, 5 and 25 between the matrix and caps. In experiments with the viscosity ratio equal to 1, the rigid inclusion rotates at a rate one half of the bulk shear strain rate, and the total rotation of the central inclusion is 27° for a shear strain of 1.0, which is approximately that predicted by Jeffery (1922) for an object of aspect ratio 1. In these experiments, the distribution of the y-velocity component, vorticity and maximum deviatoric shear stress are symmetrical about the central inclusion (Fig. 4.7.3).

In experiments with viscosity the ratio equal to 5, the inclusion rotates only 12 degrees for a shear strain of 1.0, less than half that predicted by Jeffery (1922). In addition, a downward component of velocity develops on the left side of the inclusion, and an upward component develops on the right side (Fig. 4.7.3). This corresponds with a sharp decrease in deviatoric shear stress along the top and bottom of the inclusion. Interestingly, this degree of localization along the caps nearly balances the effect of the inclusion on the bulk strain, indicated by the nearly straight edges of the deformed grid (Fig. 4.7.2).

In experiments with the viscosity ratio equal to 25, the inclusion rotates -7.5°, against the bulk sense of rotation in the monoclinic flow. The deformed grid shows that strain localization in the caps has caused marked thinning of the central part of the grid owing to the increased strain rates

(Fig. 4.7.2). The y-velocity component is strongly developed on either side of the inclusion, and the deviatoric shear stress is sharply reduced above and below the inclusion (Fig. 4.7.3).

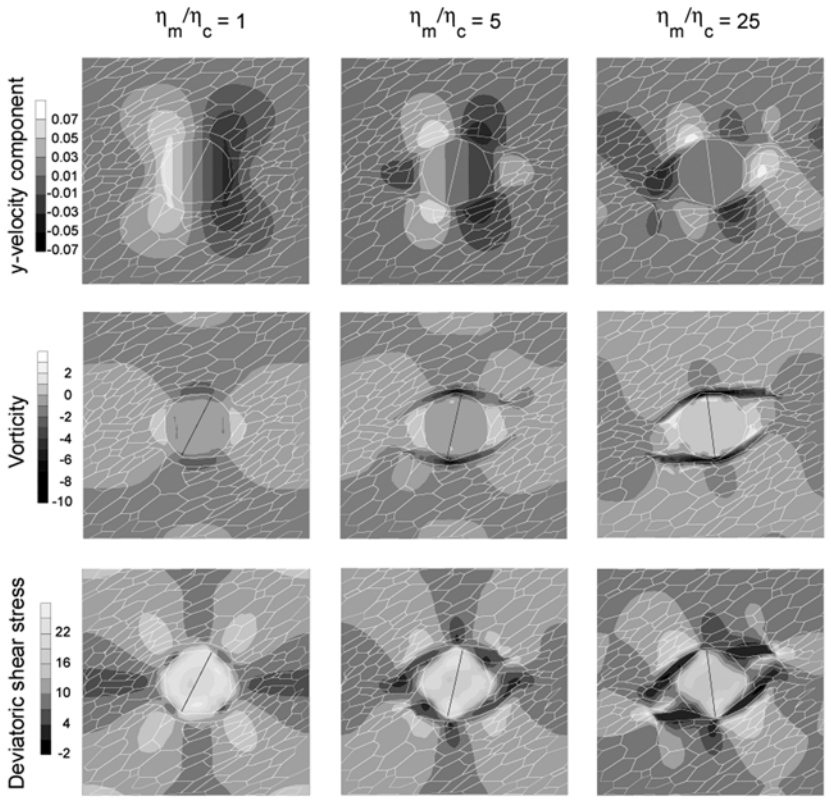


Fig. 4.7.3 Contour plots of the y-velocity component, vorticity and deviatoric shear stress for the three experiments with different viscosity caps. Note that stress and velocity are non-dimenionalised; see Chap. 3.8). Total shear strain in each is 1.0. The deformed microstructure from Fig. 4.7.2 has been overlain on the contour plots. EXPERIMENT 21

4.7.4 Discussion

A rigid inclusion embedded in a homogeneous, isotropic, viscous fluid undergoing steady-state, monoclinic flow will rotate at a rate determined by the shear strain rate, vorticity, object aspect ratio, and original orientation of the object long axis relative to the shear plane (Jeffery 1922; Ghosh and Ramberg 1976). In the simple shear experiments performed above, the vorticity is equal to the shear strain rate, and the particle aspect ratio is approximately one. Thus, the rate of inclusion rotation would depend entirely on the shear strain rate. However, in a non-homogeneous material like that investigated here, in which the viscosity of the matrix along a portion of the inclusion interface is low relative to the rest of the matrix, the rate of rotation will be determined by the balance of traction on the inclusion surface, which will vary as a function of the variable viscosity structure.

Rotation of the inclusion is caused by shear traction at the particle interface, which effectively drags the particle causing it to rotate in the direction of flow. Low viscosity caps around a particle carry less stress than the surrounding, higher viscosity matrix. At the same time, the presence of the caps causes a deflection of the flow leading to opposite and enhanced vertical velocity components along the sides of the particle. Because the viscosity of the matrix either side of the particle is higher than the viscosity of the caps, the vertical velocity components lead to a traction that competes with that along the cap-particle interfaces. With increasing viscosity contract between the caps and matrix, the y-velocity component either side of the inclusion grows in magnitude, and the antithetic traction progressively increases relative to the synthetic traction, diminishing the synthetic rotation of the inclusion. At a viscosity contract of 25, the antithetic traction is clearly stronger than the synthetic traction, causing antithetic rotation of the particle against the bulk flow.

4.8 Transient strain-rate partitioning during porphyroblast growth

Wesley G. Groome and Scott E. Johnson

4.8.1 Overview

Changing mineralogy during metamorphism can significantly affect the strength of mid-crustal rocks, and by extension the strength of relatively large volumes of the middle crust. The growth of relatively strong porphyroblasts during prograde metamorphism is a common phenomenon in metapelitic rocks and the increasing abundance of effectively rigid porphyroblasts in rocks of these bulk compositions can markedly increase their strength relative to surrounding rocks. The localization of metamorphic strengthening reactions in a stratigraphic succession leads to changes in the strain-rate partitioning within the layer undergoing the reaction and the stratigraphic succession as a whole. Changes in strain-rate partitioning during porphyroblast growth can cause further metamorphic reactions in high strain-rate zones, which can in turn lead to additional strain-rate partitioning, causing a feedback between metamorphic reaction and strain-rate partitioning (e.g. Bell and Hayward 1991, Bell et al., 2004). This feedback can cause strain-rate partitioning on the orogen scale, which can in turn cause metamorphism in other parts of the crust (e.g. Bell et al., 2004), affect the exhumation of high-grade metamorphic rocks (e.g. Jamieson et al., 2002), and potentially alter the topography of an orogen. In this contribution, we explore the evolving strain-rate partitioning behaviour within a layered succession during porphyroblast growth, and explore the feedback between strain-rate partitioning around effectively rigid porphyroblasts and metamorphic reactions.

4.8.2 Theoretical background and natural example

The strength of a polymineralic rock is largely a function of the strength, volume fraction and distribution of the constitutive minerals (e.g. Burg and Wilson 1987; Jordan 1988; Handy 1990; Tullis et al., 1991; Ji and Zhao 1993; Bons and Urai 1994; Goodwin and Tikoff 2001; Ji and Xia 2002; Treagus 2002; Johnson et al., 2004). The growth of large, effectively rigid grains (porphyroblasts) during prograde metamorphism can cause pelitic layers to strengthen relative to un-metamorphosed pelitic layers and potentially relative to interlayered lithologies that have not experienced porphyroblast growth (e.g. Groome and Johnson 2006). If porphyroblast abundance is high enough in the pelitic layer, it may become stronger than interlayered units and lead to an overall strengthening of large crustal volumes.

In un-metamorphosed turbidite sequences, psammite layers consist predominantly of quartz and feldspar, and pelite layers consist predominantly of phyllosilicates, quartz and feldspar. The predominance of relatively weak phyllosilicates in un-metamorphosed pelite units makes them weaker than interlayered psammite units during layer-parallel shearing at low to intermediate temperatures (e.g. Shea and Kronenberg 1993; Treagus 1999). However, during prograde metamorphism of pelitic rocks, the relative abundance of phyllosilicates can decrease due to the increased abundance of effectively rigid porphyroblasts, which leads to changes in the relative strengths of pelitic and psammitic rocks.

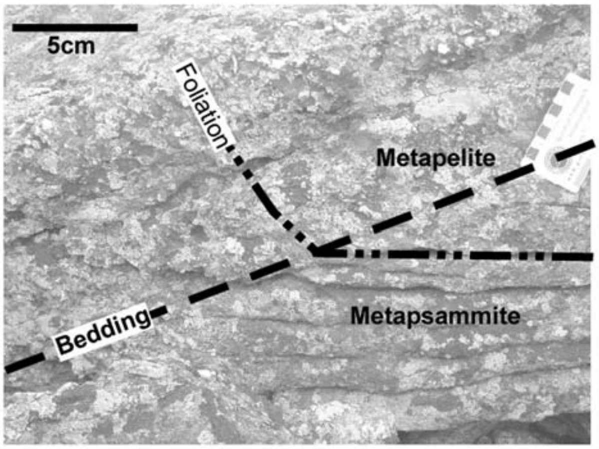


Fig. 4.8.1 Field photograph from Mt. Washington, New Hampshire, showing refracted foliation at the contact between metapsammite and metapelite. The bedding-foliation angle in the metapelite layer is larger than the bedding-foliation angle in the metapsammite layer, suggesting that the metapelite had a higher effective viscosity than the metapsammite (Eq. 4.8.1)

The White Mountains region in the northern New England Appalachians records evidence for the strengthening of pelitic layers relative to psammitic layers during prograde metamorphism. During prograde metamorphism and deformation, large andalusite grains grew in the pelitic layers, making them stronger than the psammitic layers. Andalusite grains are up to 25 cm long and constitute up to 25-30% by volume of the pelitic layers (e.g. Wall 1988; Eusden et al., 1996). By using the foliation refraction technique of Treagus (1999) to estimate effective viscosity contrasts between adjacent rock types, we determined that the pelitic layers were approximately 2 to 3 times more viscous than the interlayered porphyroblast-free psammitic layers (Groome and Johnson 2006). The use of foliation refraction angles to estimate effective viscosity contrasts is based on the relationship:

$$\frac{\tan\theta_a}{\tan\theta_b} = \frac{\eta_a}{\eta_b}, \qquad (4.8.1)$$

where subscripts *a* and *b* refer to two distinct lithologies, θ is the angle between bedding and foliation in a given bed and η is the effective viscosity of a given bed (Treagus 1999, fig. 4.8.1). This method allows for the estimation of relative viscosity contrasts between contrasting lithologies, but does not allow for the estimation of absolute viscosity values.

Using our field example from the New England Appalachians as a constraint on the strength contrast between pelitic and psammitic bulk compositions in amphibolite-facies meta-turbidites, we present a model of the evolving viscosity contrast between layers undergoing porphyroblast growth and layers without porphyroblasts. This changing viscosity contrast leads to changes in strain-rate partitioning between model pelitic layers and model psammitic layers as the pelitic layers strengthen during porphyroblast growth. Our models consist of a two layer system with contrasting viscosities (Fig. 4.8.2). The lower layer has a viscosity 2.5 times that of the upper layer, which is consistent with estimated viscosity contrasts between un-metamorphosed pelitic and psammitic bulk compositions (e.g. Treagus 1999). Isolated, fine grains within the model pelite layer were selected to be porphyroblasts, with viscosities two orders of magnitude greater than the surrounding matrix, making the porphyroblasts effectively rigid. These porphyroblasts were allowed to grow, resulting in an increase in the volume fraction of rigid porphyroblasts with each time step, which caused the viscosity structure of the model to change.

4.8.3 Implementation in *Elle*

In the experiment presented here, we use the following routines in sequence: 1) Finite Element deformation in *Basil*, 2) grain growth using *elle_gg*, and 3) porphyroblast growth using *elle_pblast*. The implementation of each process is briefly summarized below.

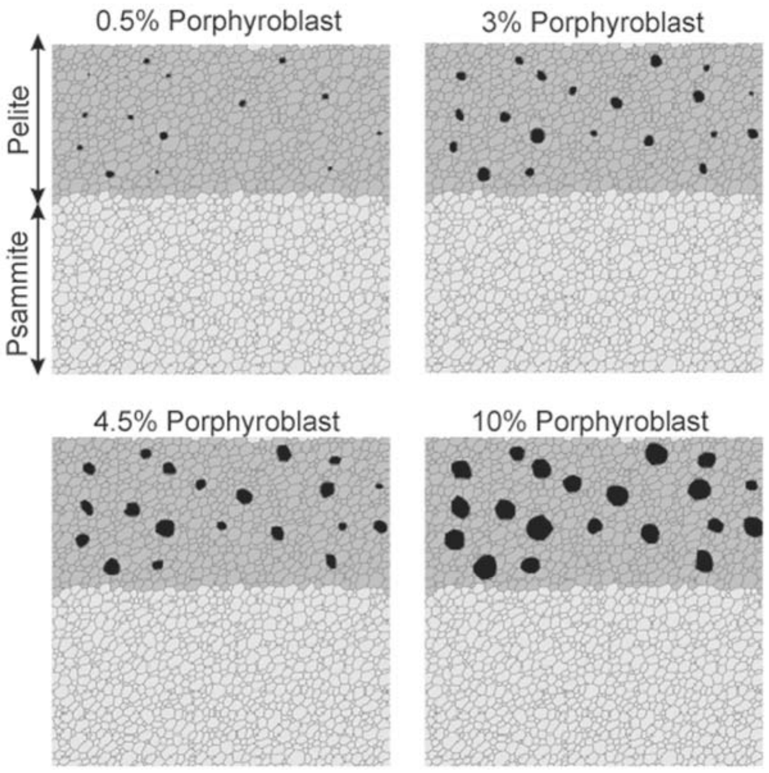


Fig. 4.8.2 Model topologies used in this paper at various porphyroblast abundances (0.5%, 3%, 4.5% and 10%). Porphyroblast abundances were calculated by dividing the area covered by porphyroblast by the area of the pelite layer (i.e. the area of the psammite layer is not included). The porphyroblasts are the darkest grains in each frame

Basil: Viscous deformation

Basil is a two-dimensional viscous Finite Element code that can solve for linear and non-linear incompressible viscous deformation (Chapt. 3.8 and 4.8). In our experiments we used a linear viscous rheology (stress exponent $n = 1$ in Eqs. 4.7.1-2). The experiments presented here use a plane strain approximation with cyclic boundaries to simulate simple shear deformation (see Chap. 3.8 for full discussion of *Basil* implementation). The models discussed below were deformed a very small increment in *Basil* (one deformation time step) in order to examine the evolving viscosity structure of the layered sequence and the corresponding evolving shear-strain-rate partitioning. Two models were deformed to shear strains of approximately 1.5 (a porphyroblast-free model and a model with a porphyroblast volume fraction of approximately 18%) to illustrate the difference in strain-rate partitioning as porphyroblast abundance increased in one layer.

elle_gg: Grain growth

The grain growth routine used for the experiments presented here calculates the grain boundary energy of each grain based on the grain boundary curvature such that high angle grain boundaries have high grain boundary energies (see Chap. 3.6). The driving force for grain growth is the energy required to minimize the grain boundary curvature. The lowest energy configuration in *Elle* is grains with 120° triple junctions. The rate of grain growth is governed by the equation:

$$R = EM, \tag{4.8.2}$$

where R is the rate of grain growth, E is the driving energy (proportional to the curvature of the grain boundary) and M is the grain boundary mobility.

elle_pblast: Porphyroblast growth

The porphyroblast growth routine used in these experiments allows selected grains to grow relative to the surrounding grains. The growth rate of the porphyroblasts is fast relative to surrounding grains and is not governed by surface energy driving forces, as in *elle_gg*. The number of porphyroblast growth stages per time step is adjustable in the input code such that porphyroblasts can grow at varying rates relative to the rate of deformation. The growing porphyroblasts obey the same governing equations as in the grain growth routine, and will attempt to form circular grains as they get larger. In these experiments, the porphyroblasts are assigned a viscosity two orders of magnitude higher than the surrounding grains, making them effectively rigid.

4.8.4 Experiments and discussion

Description of the model

The experiments shown here have porphyroblasts growing in the pelitic layer without deformation, and are referred to as static growth models. Porphyroblast abundance in the pelitic layer ranges from 0% in the initial geometry to ~18% in the final geometry. In addition, the two end-member geometries were deformed in simple shear to shear strains of approximately 1.5 to illustrate the change in strain partitioning as porphyroblast abundance increases. Figure 4.8.2 shows the geometry used at four stages of porphyroblast abundance (0.5%, 3%, 4.5% and 10% porphyroblast in the pelitic layer). The pelite layer has an assigned dimensionless viscosity of 1.0 (B in Eqs. 4.7.1-2) and the psammite layer has an assigned dimensionless viscosity of 2.5, giving a psammite to pelite viscosity ratio of 2.5, which is consistent with estimated viscosity contrasts using field data (e.g. Treagus 1999). The porphyroblasts in the model have an assigned dimensionless viscosity of 100, making them effectively rigid objects in the viscous model. Porphyroblasts in the initial stage were placed in the matrix by hand and were distributed through the pelite layer. Porphyroblast abundance at each stage was estimated by dividing the area covered by porphyroblast by the total area of the pelitic layer alone (i.e. the psammitic layer was not included in the calculation of porphyroblast abundance).

The viscosity of the system was calculated using a Gaussian integration across the x-dimension of the model to obtain an integrated viscosity for each y-value in the geometry. The bulk viscosity of the pelitic layer was calculated by integrating the area under the viscosity profile from the bedding contact to the top of the model. A single *Basil* deformation step was conducted for each model to obtain a shear-strain-rate contour map for each stage of porphyroblast growth (via Eq. 4.7.1), which provides the basis for the viscosity calculation.

Results

Figure 4.8.3 shows the evolving viscosity structure and strain-rate partitioning in our layered system as the porphyroblast abundance increases in the pelitic layer. The starting geometry, with no porphyroblasts, has a uniform viscosity structure through the pelitic layer, and strain rate is uniformly partitioned into the pelitic layer. As porphyroblast growth occurs, however, the viscosity structure and strain-rate partitioning through the pelitic layer becomes less regular. High viscosity regions correspond to areas that have high porphyroblast abundance, whereas low viscosity regions correspond to porphyroblast-free areas. As porphyroblast abundance increases, the shear-strain-rate partitions around porphyroblasts within the pelitic layer, in addition to larger-scale partitioning into porphyroblast-free psammitic layers. Figure 4.8.4 shows the evolving viscosity structure of just the pelitic layer as porphyroblast abundance increases from 0.5% to 18%. High viscosity spikes in the viscosity structure correspond to areas in the geometry where porphyroblast abundance is high, whereas low viscosity spikes correspond to areas in the geometry where porphyroblast abundance is low. The low viscosity spikes in the pelitic layer correspond with zones of high shear-strain-rate.

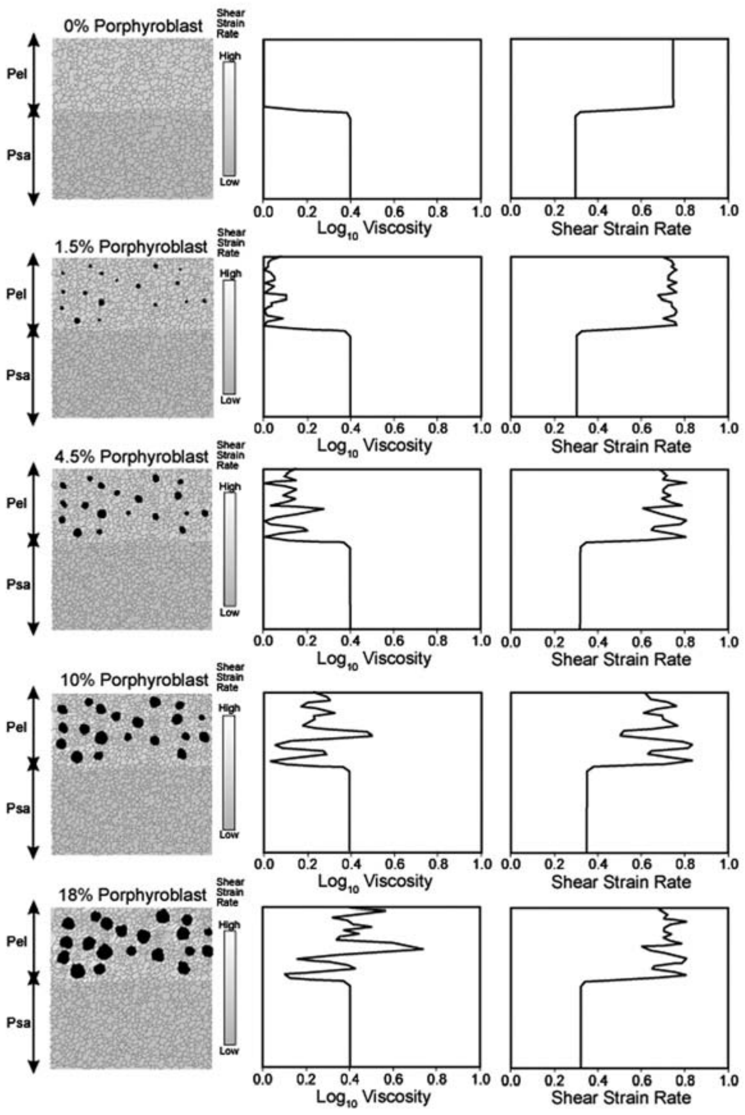


Fig. 4.8.3 Contour maps of shear strain rate and plots showing the viscosity structure and shear strain rate distribution of the model for five stages of porphyroblast growth (0%, 1%, 4.5%, 10% and 18%). Note how the shear strain rate increases in the psammitic layer as porphyroblast abundance increases, corresponding to an increase in the bulk viscosity of the pelitic layer. The grain boundary topologies for each stage are overlain on the strain-rate maps. EXPERIMENT 14

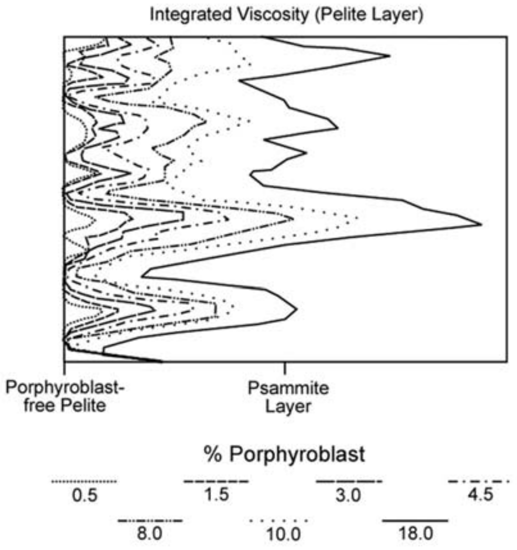


Fig. 4.8.4 Plot of the evolving viscosity structure of the pelitic layer as porphyroblast abundance increases from 0% to 18%. The location of the starting pelite viscosity and the psammite viscosity are indicated

The changing bulk viscosity of the model pelite layer as porphyroblast abundance increases is shown in Fig. 4.8.5. The trend of increasing viscosity in our model falls between two end-member theoretical bounding conditions describing the strength of poly-phase materials: the Voigt Bound and the Reuss Bound. The Voigt Bound assumes that the strain rate for each phase (and the bulk strain) is the same and that the bulk strength (σ_c) of a polyphase material increases linearly with increasing volume fraction (V_s) of the strong phase (σ_s) (e.g. Ji and Xia 2002):

$$\sigma_c = V_s\sigma_s + (1 - V_s)\sigma_w, \tag{4.8.3}$$

where σ_w is the strength of the weak matrix phase. The Reuss Bound assumes that all phases are subjected to a constant differential stress and that the bulk strain rate ($\dot{\varepsilon}_c$) is a function of the strain rate in each phase ($\dot{\varepsilon}_s, \dot{\varepsilon}_w$) (e.g. Ji and Xia 2002):

$$\dot{\varepsilon}_c = V_s\dot{\varepsilon}_s + (1 - V_s)\dot{\varepsilon}_w. \tag{4.8.4}$$

Most treatments of the rheology of natural polyphase materials assume that natural rheologic trends fall between the two theoretical end-members (e.g. Ji and Xia 2002), which is consistent with the experiments presented here.

Two geometries were deformed in simple shear to bulk strains of approximately 1.5 to illustrate the difference between strain partitioning in a porphyroblast-free layered system and a porphyroblastic layered system. In the absence of porphyroblasts, the pelitic layer in our model has a viscosity 0.4 times that of the psammitic layer. Strain is partitioned in this model such that the pelitic layer records a shear strain of approximately 2.1 and the psammitic layer records a shear strain of approximately 0.8, consistent with the viscosity contrast between the two layers (Fig. 4.8.6). A second geometry, with approximately 18% porphyroblast in the pelitic layer, was deformed to a similar bulk strain.

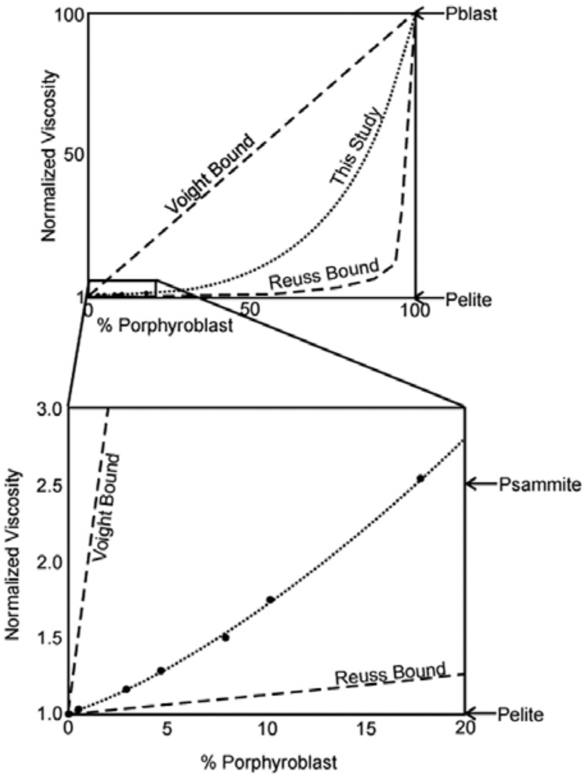


Fig. 4.8.5 Graph of the changing viscosity of the pelitic layer as porphyroblast abundance increases. The theoretical Voigt and Reuss bounds are also shown with the viscosity values used in this model as parameters. Note that the trend line for this experiment lies between the two theoretical bounds. All viscosities are normalized to the porphyroblast-free pelite layer. Also shown are the values of the psammite layer and a layer consisting of 100% porphyroblast (Pblast)

In this model, strain is not partitioned as significantly between the two layers, with the pelite layer recording a shear strain of approximately 1.6 and the psammite layer recording a shear strain of approximately 1.4 (Fig. 4.8.6), because with approximately 18% porphyroblast the pelitic layer has a bulk viscosity similar to the psammitic layer (Fig. 4.8.5). Furthermore, in the pelitic layer, shear-strain-rates are not uniformly distributed through the layer, but instead partition around the porphyroblasts.

4.8.5 Conclusions

Porphyroblast growth during metamorphism will strengthen the rock if the product porphyroblasts are stronger than the reactant phases. Within the layer experiencing porphyroblast growth, the strain rate distribution will become more heterogeneous as porphyroblast abundance increases and strain is partitioned into ever decreasing rock volumes around the porphyroblasts (Fig. 4.8.6; e.g. Bell and Hayward 1991). This increasing strain rate partitioning around the porphyroblasts leads to the development of high strain rate gradients along the margins of the porphyroblasts. In nature, this may allow for grain boundaries to dilate along the margins of the porphyroblasts, which will provide pathways for fluids in the rock that can enhance metamorphic reaction rates (e.g. Etheridge et al., 1983). If the rock is relatively anhydrous, as expected in the lower crust, the opening of grain boundaries will allow for fluids to catalyze retrograde metamorphic reactions, as reported from the Sesia Zone in the Swiss Alps (e.g. Rubie 1986; Brodie and Rutter 1985; Koons et al., 1987; Freuh-Green 1994). However, if the rock is hydrous, as would be expected during the prograde metamorphism of pelitic rocks in the middle crust, the opening of grain boundaries may allow for fluids to leave the immediate reaction site, thus allowing dehydration to progress (e.g. Etheridge et al., 1983).

High strain rates in grains along the margins of porphyroblasts may increase dislocation densities in grains undergoing intracrystalline deformation (e.g. Bell and Hayward 1991). As dislocation densities increase, the free energy available for metamorphic reactions is increased which will enhance the metamorphic reaction rate if the pressure-temperature conditions are overstepped (e.g. Porter and Easterling 1992). In nature, this may lead to rapid reaction rates if a reaction is overstepped and porphyroblasts begin to nucleate (e.g. Bell and Hayward 1991). Once a porphyroblast nucleates and begins to grow, the increasing strain rates in the surrounding grains will add strain energy to the system, which can lower the energy barrier for further progress of metamorphism.

As a rock strengthens during porphyroblast growth, strain rates will be partitioned out of the regions undergoing porphyroblast growth. The partitioning of high strain rates into surrounding crustal volumes may catalyze metamorphic reactions in parts of the crust that are not undergoing metamorphism (e.g. Bell at al. 2004). If the rocks in these crustal volumes are in disequilibrium with respect to the ambient pressure and temperature conditions, the strain energy added to these rocks as they begin to deform may be enough to overcome the energy barrier for nucleation and growth. If this energy barrier is overcome, metamorphic reactions will occur, potentially strengthening these crustal volumes and leading to further strain rate partitioning into other parts of the crust (Groome et al., 2006). In this way a feedback relation between porphyroblast growth and strain-rate partitioning can develop and metamorphic reactions can occur over large crustal volumes in a relatively short period of time.

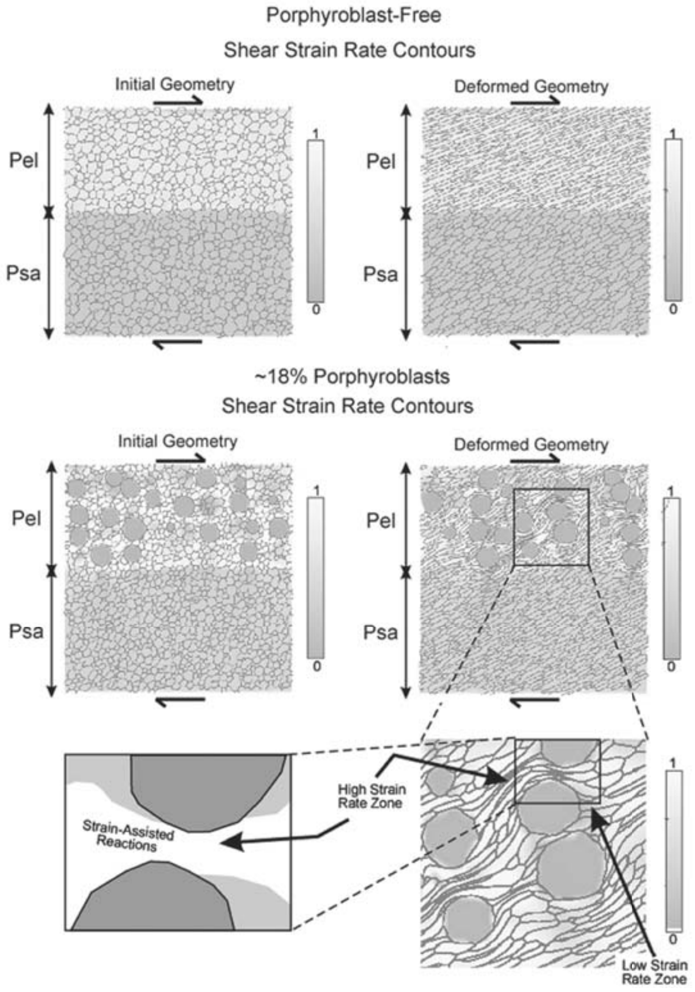


Fig. 4.8.6 Top: Shear strain rate contour maps for the porphyroblast-free simple shear deformation experiment. Note the homogeneous shear strain rate in each layer and the bulk strain rate partitioning into the pelitic layer. Middle: Shear strain rate contour maps for the 18% porphyroblast simple shear deformation experiment. Note strain rate partitioning within the pelitic layer with high strain rate zones around the porphyroblasts. Also, the strain rate is higher in the psammitic layer than in the porphyroblast-free experiment. Bottom: Close-up images of the shear strain rate distribution around porphyroblasts showing high and low shear strain rate zones. The image on the left has had the grain topology removed for simplification. The location of strain-assisted reaction between two porphyroblasts is indicated. See text for discussion

References

Aerden DGAM (1995) Porphyroblast non-rotation during crustal extension in the Variscan Lys-Caillaouas Massif, Pyrenees. J Struct Geol 17:709–725

Aharonov E, Sparks D (1999), Rigidity phase transition in granular packings. Phys Rev E 60:6890-6896

Akaogi M, Ito E, Navrotsky A (1989) Olivine-modified spinel-spinel transitions in the system Mg_2SiO_4 – Fe_2SiO_4: Calorimetric measurements, thermochemical calculation, and geophysical application. J Geophys Res 94:15671-15685

Allen DN de G (1954) Relaxation methods in engineering and science. McGraw Hill, New York

Allen TT (1992) Migmatite systematics and geology, Carter Dome-Wild River Region, White Mountains, New Hampshire. Unpubl PhD Dissertation, Dartmouth College, Hanover, NH. pp 249

Alcantar N, Israelachvili J, Boles J (2003) Forces and ionic transport between mica surfaces: Implications for pressure solution. Geochim Cosmochim Acta 67:1289-1304

Alkattan M, Oelkers EH, Dandurand JL, Schott J (1997) Experimental studies of halite dissolution kinetics, 1. The effect of saturation state and the presence of trace metals. Chem Geol 137:201-219

Altherthum H (1922) Zur Theorie der Rekristallisation. Z. Metallk. 14:417-424

Anderson MP (1988) Simulation of grain growth in two and three dimensions. In: Hansen N, Juul Jensen D, Leffers T, Ralph B (eds) Annealing Processes-Recovery, Recrystallization and Grain Growth. Risø National Laboratory, Roskilde, Denmark, pp 15-34

Anderson MP, Srolovitz DJ, Grest GS, Sahni PS (1984) Computer simulations of grain growth. I Kinetics Acta Metall 32:783-791

Arbaret L, Mancktelow NS, Burg JP (2001) Effect of shape and orientation on rigid particle rotation and matrix deformation in simple shear flow. J Struct Geol 23:113–125

Argon AS (1970) Internal stresses arising from the interaction of mobile dislocations. Scripta Metall 4:1001-1004

Avrami M (1939) Kinetics of Phase Change. I: General Theory. J Chem Phys 7:1103–1112

Avrami M (1941) Kinetics of Phase Change III: Granulation, Phase Change, and Microstructure. J Chem Phys 9:177–184

Avrami M (1940) Kinetics of Phase Change II: Transformation-Time Relations for Random Distribution of Nuclei. J Chem Phys 8:212–224

Bacon DJ (1967) A Method for Describing a Flexible Dislocation Physica status solidi 23:527-538

Bacon DJ, Kocks UF, Scattergood RO (1973) Effect of dislocation self-interaction on Orowan stress. Phil Mag 28:1241-1263

Bacroixt B, Gilorminit P (1995) Finite Element simulations of earing in polycrystalline materials using a texture-adjusted strain-rate potential. Modelling Simul Mater Sci Eng 3:1-21

Bak P, Chen K, Tang C (1990) A forest-fire model and some thoughts on turbulence. Physics letters A 147: 297-300

Barber D J (1985) Dislocations and Microstructures. In: Wenk H-R (ed) Preferred Orientation in Deformed Meals and Rocks: An Introduction to Modern Texture Analysis. Academic Press, London, pp 149-182

Barker CE, Kopp OC (1991) Luminescence Microscopy and Spectroscopy: Qualitative and Quantitative Applications. Society for Sedimentary Geology, Tulsa, Oklahoma

Barr TD, Houseman GA (1992) Distribution of deformation around a fault in a non-linear ductile medium. Geophys Res Lett 19:1145-1148

Barr TD, Houseman GA (1996) Deformation fields around a fault embedded in a non-linear ductile medium. Geophys J Int 125:473-490

Baskes MI (1999) The status role of modeling and simulation in Mat Sci Eng. Current Opinion in Solid State and Materials Science 4: 273-277

Bate P (2001) Modelling microstructure development in annealing and deformation. In: Gottstein G, Molodov DA (eds) Recrystallization and Grain Growth. Proceedingsof the First Joint International Conference, Springer Verlag, Berlin, pp 39-48

Bathurst RCG (1958) Diagenetic fabrics in some Dinantian limestones. Liverpool and Manchester Geological Journal 2:11-36

Bathurst RJ, Rothenburg L (1988) Micromechanical aspects of isotropic granular assemplies with linear contact interactions. J Appl Mech 55:17-23

Bauer P, Rosenberg C, Handy MR (2000) See-trough deformation experiments on brittle-viscous norcamphor at controlled temperature, strain rate and applied confining pressure. J Struct Geol 22:281-289

Beach A (1979) Pressure Solution As A Metamorphic Process In Deformed Terrigenous Sedimentary-Rocks. Lithos 12:51-58

Beeman M I, Kohlstedt DI (1988) Dislocation density: stress relationships in natural and synthetic sodium chloride. Tectonophysics 149:147-161

Bell TH, Hayward N (1991) Episodic metamorphic reactions during orogenesis: the control of deformation partitioning on reaction sites and reaction duration. J Metam Geol 9:619-640

Bell TH, Johnson SE, Davis B, Forde A, Hayward N, Wilkins C (1992) Porphyroblast inclusion-trail orientation data: eppure non son girate! J Metam Geol 10:295–307

Bell TH, Ham AP, Kim HS (2004) Partitioning of deformation along an orogen and its effects on porphyroblast growth during orogenesis. J Struct Geol, 26:825-845

Bestman M, Prior DJ (2003) Intragranular dynamic recrystallization in naturally deformed calcite marble: diffusion accommodated grain boundary sliding as a result of subgrain rotation recrystallization. J Struct Geol 25: 1597-1613

Biben T (2005) Phase-field models for free-boundary problems. Eur J Phys 26:S47-S55

Biermeier C, Stüwe K, Barr TD (2001) The rotation rate of cylindrical objects during simple shear. J Struct Geol 23:765-776

Biermeier C, Stüwe K (2003) Strain rates from snowball garnet. J Metam Geol 21:253-268

Billen MI, Houseman GA (2004) Lithospheric instability in obliquely convergent margins: San Gabriel Mountains, southern California. J Geophys Res 109: doi 10.1029/2003JB002605,

Bishop JFW, Hill R (1951a) A theory of plastic distortion of a polycrystalline aggregate under combined stresses. Phil Mag 42:414-427

Bishop JFW, Hill R (1951b). A theoretical derivation of the plastic properties of a polycrystalline face-centered metal. Phil Mag 42:1298-1307

Bjørnerud MG (1989) Mathematical model for folding of layering near rigid objects in shear deformation. J Struct Geol 11:245-254

Bjørnerud MG, Zhang H (1994) Rotation of porphyroblasts in non-coaxial deformation; insights from computer simulations. J Metam Geol 12:135-139

Bjørnerud MG, Zhang H (1995) Flow mixing, object-matrix coherence, mantle growth and the development of porphyroclast tails. J Struct Geol 17:1347-1350

Bons PD (2000) The formation of veins and their microstructures. In: Jessell MW, Urai JL (eds) Stress, Strain and Structure, A volume in honour of WD Means. (on-line) J Virtual Explorer 2

Bons PD (2001) Development of crystal morphology during antitaxial growth in a progressively opening fracture: I. The numerical model. J Struct Geol 23:865-872

Bons PD, den Brok SWJ (2000) Crystallographic preferred orientation development by dissolution-precipitation creep. J Struct Geol 22:1713-1722

Bons PD, Urai JL (1992) Syndeformational grain growth: Microstructures and kinetics. J Struct Geol 14:1101-1109

Bons PD, Urai JL (1994) Experimental deformation of two-phase rock analogues. Mat Sci Eng A175:221-229

Bons PD, Barr TD, ten Brink CE (1997) The development of delta–clasts in non-linear viscous materials: a numerical approach. Tectonophysics 270:29-41

Bons PD, Jessell MW, Evans L, Barr TD, Stüwe K (2001) Modelling of anisotropic grain growth in minerals. In: Koyi HA, Mancktelow NS (eds) Tectonic Modeling: A Volume in Honor of Hans Ramberg. Geol Soc America Memoir 193:45-49

Braun RJ, Cahn JW, McFadden GB, Wheeler AA (1997) Anisotropy of interfaces in an ordered alloy: a multiple-order-parameter model. Trans R Soc London A 355:1787-1833

Brodie KH, Rutter EH (1985) On the relationship between deformation and metamorphism, with special reference to the behavior of basic rocks. In: Thompson AB, Rubie DC (eds) Metamorphic Reactions: Kinetics, Textures and Deformation. Advances in Physical Geochemistry 4:138-179

Brodie KH (1995) The development of oriented symplectites during deformation. J Metam Geol 13:499-508

Brown LM (1967) A proof of Lothe's theorem, Phil Mag A 15:363-370

Bulau JR, Waff HS, Tyburczy JA (1979) Mechanical and thermodynamical constraints on fluid distribution in partial melts. J Geophys Res 84:6102-6108

Buiter SJH, Babeyko AYu, Ellis S, Gerya TV, Kaus BJP, Kellner A, Schreurs G, Yamada Y (2006) The Numerical Sandbox: Comparison of Model Results for a Shortening and an Extension Experiment. In: Buiter SJH and Schreurs G (eds) Analogue and Numerical Modelling of Crustal-Scale Processes, Geol Soc Lond Spec Publ 253:29-64

Burg JP, Wilson CJL (1987) Deformation of two phase systems with contrasting rheologies. Tectonophysics 135:199-205

Burke JE, Turnbull D (1952) Prog Metal Physics 3:220

Cahn, JW (1956) The kinetics of grain boundary nucleated reactions. Acta Metall 4:449-459

Canova GR (1994) Self-consistent methods: application to the prediction of the deformation texture of polyphase materials. Mat Sci Eng A 175:37-42

Carpenter HCH, Elam CF (1920) Crystal growth and recrystallization in metals. J Inst Met 24:83-131

Carter NL, Tsenn MC (1987) Flow properties of continental lithosphere. Tectonophysics 136:27-63

Casey M, McGrew AJ (1999) One-dimensional kinematic model of preferred orientation development. Tectonophysics 303:131-140

Chang J, Cai W, Bulatov VV, Yip S (2002) Molecular dynamics simulations of motion of edge and screw dislocations in a metal. Computational Materials Sci 23:111-115

Chrzan DC, Erdonmez CK (2001) Dislocation dynamics in Ll2 compounds displaying the yield strength anomaly: extracting scaling parameters from simulations. Mat Sci Eng A319-321:31-36

Ceriani S, Mancktelow NS, Pennacchioni G (2003) Analogue modelling of the influence of shape and particle/matrix interface lubrication on the rotational behaviour of rigid particles in simple shear. J Struct Geol 25:2005–2021

Clark MK, Royden LH (2000) Topographic ooze: Building the eastern margin of Tibet by lower crustal flow. Geology 28:703-706

Cleri F (2000) A stochastic grain growth model based on a variational principle for dissipative systems. Physica A 282:339-354

Cmíral M, Fitz Gerald JD, Faul UH (1998) A close look at dihedral angles and melt geometry in olivine-basalt aggregates: a TEM study. Contrib Mineral Petrol 130:336-345

Coble RL (1963) A model for boundary diffusion controlled creep in polycrystalline materials: J Appl Phys 34:1679-1682

Cocks ACF, Gill SPA, Pan JZ (1999) Modeling microstructure evolution in engineering materials. Advances Appl Mech 36:81-162

Connolly JAD (1990) Multivariable phase diagrams: an algorithm based on generalized thermodynamics. Am J Sci 290:666-718

Cooper RF, Kohlstedt DL (1984) Solution-precipitation enhanced diffusional creep of partially molten olivin-basalt aggregates during hot pressing. Tectonophysics 107:207–233

Cordier P (2005) Plastic deformation of minerals at high pressure: Multiscale numerical modelling. In: Miletich R (ed) Mineral behaviour at extreme conditions, EMU notes in Mineralogy 7

Cundall PA, Strack ODL (1979) A discrete numerical model for granular assemblies. Géotechnique 29:47-65

Cygan RT, Lasaga AC (1985) Self-diffusion of magnesium in garnet at 750° to 950°C. Am J Sci 285:328-350

D'Addetta GA, Kun F, Ramm E, Herrmann HJ (2001) From solids to granulates - discrete element simulations of fracture and fragmentation processes in geomaterials. In: Vermeer PA, Diebels S, Ehlers W, Herrmann HJ, Luding S, Ramm E (eds) Lecture Notes in Physics 568:231-257

Dässler R, Yuen D, Karato S, Riedl M (1996) Two-dimensional thermo-kinetic model for the olivine-spinel phase transition in subducting slabs. Phys Earth Planetary Interiors 94:217-239

Davies CHJ, Hong L (1999) The cellular automaton simulation of static recrystallization in cold-rolled AA1050. Scripta Materialia 40:1145–1150

Davis P, England P, Houseman G (1997) Comparison of shear wave splitting and finite strain from the India-Asia collision zone. J Geophys Res DOI 102:27511-27522

De Boer RB (1977) On the thermodynamics of pressure solution - interaction between chemical and mechanical forces. Geochim Cosmochim Acta 41:249-256

de Bresser JHP, Peach CJ, Reijs JPJ, Spiers CJ (1998) On dynamic recrystallization during solid state flow: Effects of stress and temperature. Geophys Res Lett 25:3457-3460

den Brok SWJ (1996) The effect of crystallographic orientation on pressure solution in quartzite. J Struct Geol 18:859-860

den Brok B (1998) Effect of microcracking on pressure-solution strain rate: The Gratz grain-boundary model. Geology 26:915-918

Devincre B, Condat M (1992) Model validation of a 3d simulation of dislocation dynamics: discretization and line tension effects. Acta metall Mater 40:2629-2637

Devincre B, Kubin LP, Lemarchand C, Madec R (2001) Mesoscopic simulations of plastic deformation. Mat Sci Eng A 309:211-219

Doherty RD, Hughes DA, Humphreys FJ, Jonas JJ, Jensen DJ, Kassner ME, King WE, McNelley TR, McQueen HJ, Rollett AD (1997) Current issues in recrystallization: a review. Mat Sci Eng A 238:219-274

Douglas Jr J (1955) On the numerical integration of $\partial u/\partial t=\partial^2 u/\partial x^2+\partial^2 u/\partial y^2$by implicit methods. J Soc Industrially Appl Math 3:42-65

Douglas Jr J, Peaceman D (1955) Numerical solution of two-dimensional heat flow problems. Am Inst Chem Eng J 1:505-512

Dowty E (1980) Crystal Growth and Nucleation Theory and the Numerical Simulation of Igneous Crystallisation. In Hargraves RB (ed) Physics of Magmatic Processes. Princeton Univ Press

Durham WB, Goetze C, Blanke B (1977) Plastic flow of oriented single crystals of olivine: 2. Observations and interpretations of the dislocation structures. J Geophys Res 82:5755-5770

Durney DW, Ramsay JG (1973) Incremental strains measured by syntectonic crystal growths, In: De Jong, KA, Scholten, K. (eds) Gravity and Tectonics. John Wiley & Sons, New York, pp 67–96

Ehlers K, Powell R, Stuwe K (1994) Cooling rate histories from garnet+biotite equilibrium. Am Miner 79:737-744

Einstein A (1906) Eine neue bestimmung der molekudimensionen. Annales de Physique 19:289–306

Ellis S, Wissing S, Pfiffner A (2001) Strain localization as a key to reconciling experimentally derived flow-law data with dynamic models of continental collision. Int J Earth Sci 90:168-180

Emmerich H (2003) The diffuse interface approach in material science. Lecture Notes in Physics 73. Springer, Berlin

Ercolessi F, Adams JB (1994) Interatomic potentials from 1st-principles calculations – the force-matching method. Europhys Lett 26 (8):583-588

Etchecopar A (1977) A plane kinematic model of progressive deformation in a polycrystalline aggregate. Tectonophysics 39:121-139

Etchecopar A, Malavieille J (1987) Computer models of pressure shadows: a method for strain measurement and shear-sense determination. J Struct Geol 9:667-677

Etchecopar A, Vasseur G (1987) A 3-D kinematic model of fabric development in polycrystalline aggregates. Comparisons with experimental and natural examples. J Struct Geol 9:705-717

Etheridge MA, Wall VJ, Vernon RH (1983) The role of the fluid phase during regional metamorphism and deformation. J Metam Geol 1:205-226

Eusden JD, Garesche JM, Johnson AH, Maconochie J, Peters SP, O'Brien JB, Widmann BL (1996) Stratigraphy and ductile structure of the Presidential Range, New Hampshire: Tectonic implications for the Acadian Orogeny. GSA Bulletin 108:417-437

Ewing JA, Rosenhain W (1900) The Crystalline Structure of Metals. Second Paper. Proceedings of the Royal Society of London 67:112-117

Fan D, Chen L-Q (1997) Topological evolution during coupled grain growth and ostwald ripening in volume-conserved 2-D two-phase polycrystals. Acta Materialia 45:4145-4154

Faul UH (1997) Permeability of partially molten upper mantle rocks from experiments and percolation theory. J Geophys Res 102:10299–10311

Ferguson CC (1979) Rotations of elongate rigid particles in slow non-Newtonian flows. Tectonophysics 60:247-262

Fernandez A, Febesse JL, Mezure JF (1983) Theoretical and experimental study of fabrics developed by different shaped markers in two-dimensional simple shear. Bulletin de la Société Géologique de France. 319-326

Fernandez A, Laporte D (1991) Significance of low symmetry fabrics in magmatic rocks. J Struct Geol 13:337-347

Ferry JM, Spear FS (1978) Experimental calibration of the partitioning of Fe and Mg between biotite and garnet. Contrib Mineral Petrol 66:113-117

Fitz Gerald JD, Etheridge M, Vernon RH (1993) Dynamic recrystallization in a naturally recrystallized albite. Textures and Microstructures 5:219-237

Flekkøy EG, Malthe-Sørenssen A, Jamtveit B (2002) Modeling hydrofracture. J Geophys Res 107:1-11

Ford JM, Wheeler J, Movchan AB (2002) Computer simulation of grain boundary creep: Acta Materialia 50:3941-3955

Ford JM, Ford NJ, Wheeler J (2004) Simulation of grain boundary diffusion creep: analysis of some new numerical techniques: Proc R Soc, London, Series A 460:2395-2413

Foreman AJE, Makin MJ (1966) Dislocation movement through random arrays of obstacles. Phil Mag 14:911-924

Foster CT (1999) Forward modeling of metamorphic textures. Canadian Mineralogist 37:415-429

Frisch U, Hasslacher B, Pomeau Y (1986) Lattice-gas automata for the Navier-Stokes Equation. Phys Rev Letts 56:1505-1508

Frost HJ, Ashby MF (1983). Deformation-Mechanism Maps: The Plasticity and Creep of Metals and Ceramics. Pergamon Press, Oxford

Frueh-Green GL (1994) Interdependence of deformation, fluid infiltration and reaction progress recorded in eclogitic metagranitoids (Sesia Zone, Western Alps). J Metam Geol 12:327-343

Fueten F (1997) A computer-controlled rotating polarizer stage for the petrographic microscope. Computers & Geoscience 23:203-208

Fueten F, Goodchild JS (2001) Quartz c-axes orientation determination using the rotating polarizer microscope. J Struct Geol 23:895-902

Fyson WK (1980) Fold fabrics and emplacement of an Archean granitoid pluton, Cleft Lake, Northwest Territories. Canad J Earth Sci 17:325–332

Gay NC (1968) The motion of rigid particles embedded in a viscous fluid during pure shear deformation of the fluid. Tectonophysics 5:81-88

Gershenfeld N (1999) The nature of mathematical modeling. Cambridge Univ Press

Ghoniem NM, Tong S-H, Sun LZ (2000) Parametric dislocation dynamics: A thermodynamics-based approach to investigations of mesoscopic plastic deformation. Phys Rev B 61:913-927

Ghosh SK, Ramberg H (1976) Reorientation of inclusions by combination of pure shear and simple shear. Tectonophysics 34:1-70

Ghosh SK, Sen G, Sengupta S (2003) Rotation of long tectonic clasts in transpressional shear zones. J Struct Geol 25:1083-1096

Ghoussoub J, Leroy YM (2001) Solid-fluid phase transformation within grain boundaries during compaction by pressure solution. J Mech Phys Solids 49: 2385-2430

Gibbs JW (1906) On the equilibrium of heterogeneous substances. In: The Scientific Papers of J. Willard Gibbs 1. Toronto, Ontario, Longmans, Green and Company, 55-349

Glazier JA, Gross SP, Stavans J (1987) Dynamics of two dimensional soap froths. Phys Rev A 36:306-312

Goodwin LB, Tikoff B (2001) Competency contrast, kinematics and the development of foliations and lineations in the crust. J Struct Geol 24:1065-1085

Gotoh M, Ishise F (1978) finite element analysis of rigid-plastic deformation of the flange in a deep-drawing process based on a fourth-degree yield function. Int J Mech Sci 20:423-435

Gottstein G, Mecking H (1985). Recrystallization. In: Wenk H-R (ed) Preferred Orientation in Deformed Metals and Rocks: An Introduction to Modern Texture Analysis. Academic Press, London, pp 183-214

Gottstein G, Shvindlerman LS (1999) Grain Boundary Migration in Metals: Thermodynamics, Kinetics, Applications. CRC Press, LLC, Boca Roca

Grasemann B, Stüwe K (2001) The development of fringe folds during simple shear deformation and their use as kinematic indicators. J Struct Geol 23:715-724

Griffith AA (1920) The phenomena of rupture and flow in solids. Trans R Soc Phil Ser A 221:163-198

Groome WG, Johnson SE (2006) Constraining the relative strengths of high-grade metamorphic rocks using foliation refraction angles: an example from the Northern New England Appalachians. J Struct Geol 28:1261-1276

Groome WG, Johnson SE, Koons PO (2006) The effects of porphyroblast growth on the effective viscosity of metapelitic rocks: implications for the strength of the middle crust. J Metam Geol 24:389-40

Guillopé M, Poirier J (1979) Dynamic recrystallization during creep of single crystalline halite: an experimental study. J Geophys Res 4:5557-5567

Guo YF, Wang CY, Zhao DL (2003) Atomistic simulation of crack cleavage and blunting in bcc-Fe. Mat Sci Eng A349:29-35

Handy MR (1990) The solid state flow of polymineralic rocks. J Geophys Res 95:8647-8661

Hanmer S, Passchier CW (1991) Shear sense indicators: a review. Geol Surv Canada 90:1-71

Haslam AJ, Yamakov V, Moldovan D, Wolf D, Phillpot SR, Gleiter H (2004) Effects of grain growth on grain-boundary diffusion creep by molecular-dynamics simulation. Acta Materialia 52:1971-1987

Hazzledine PM, Schneibel JH (1993) Theory of Coble creep for irregular grain structures: Acta Metallurgica et Materialia 41:1253-1262

Heilbronner RP, Pauli C (1993) Integrated spatial and orientation analysis of quartz c-axes by computer-aided microscopy. J Struct Geol 15:369-382

Hering E, Martin R, Stohrer M (1992) Physik für Ingenieure. VDI-Verlag, Düsseldorf

Hilgers C, Koehn D, Bons PD, Urai JL (2001) Development of crystal morphology during unitaxial growth in a progressively widening vein: II. Numerical simulations of the evolution of antitaxial fibrous veins. J Struct Geol 23: 873–885

Hippertt JF (1994) Microstructures and c-axis fabrics indicative of quartz dissolution in sheared quartzites and phyllonites. Tectonophysics 229:141-163

Hirth G, Kohlstedt DL (1995) Experimental constraints on the dynamics of the partially molten upper mantle: Deformation in the diffusion creep regime. J Geophys Res 100:1981–2001

Hirth JP, Lothe J (1982) Theory of Dislocations. Wiley, New York

Hirth G, Tullis J (1992) Dislocation creep regimes in quartz aggregates. J Struct Geol 14:145-159

Hobbs BE, Ord A, Teyssier C (1986) Earthquakes in the ductile regime? Pure Appl Geophys 124:309-336

Hoffman AW, Giletti BJ, Hinthorne JR and Andersen CA, Comaford D (1974) Ion microprobe analysis of a potassium self-diffusion experiment in biotite. Earth Planet Sci Lett 24:48-52

Holm EA, Miodownik MA, Healey KJ (2004) A subgrain growth model for strain-free nucleation during recrystallization. Material Science Forums 467-470:611-616

Houseman GA, England PC (1996) A lithospheric-thickening model for the Indo-Asian collision. In: Yin A, Harrison M (eds) Tectonic Evolution of Asia, Cambridge Univ Press, New York, pp 3-17

Houseman GA, Gubbins D (1997) Deformation of subducted oceanic lithosphere, Geophys J Int 131:535-551

Houseman G, Molnar P (1997) Gravitational (Rayleigh-Taylor) instability of a layer with non-linear viscosity and convective thinning of continental lithosphere. Geophys J Int 128:125-150

Houseman G, Neil EA, Kohler MD (2000) Lithospheric instability beneath the Transverse Ranges of California. J Geophys Res 105:16237-16250

Huebner KH (1975) The Finite Element Method for Engineers, John Wiley, New York

Humphreys FJ, Hatherly M (1995) Recrystallization and related annealing phenomena, Oxford, Pergamon

Hutchinson JW (1970) Elastic-Plastic Behavior of Polycrystalline Metals and Composites. Proc R Soc London A 319:247-272

Ildefonse B, Sokoutis D, Mancktelow NS (1992) Mechanical interactions between rigid particles in a deforming ductile matrix— analog experiments in simple shear-flow. J Struct Geol 14:1253–1266

Ilg BR, Karlstrom KE (2000) Porphyroblast inclusion trail geometries in the Grand Canyon: evidence for non-rotation and rotation? J Struct Geol 22:231-243

Jaeger JC, Cook NGW (1976) Fundamentals of rock mechanics. Chapman and Hall London

Jagota A, Bennison J (1995) Element breaking rules in computational models for brittle fracture. Modelling Simul Matter Sci Eng 3:485-501

Jamieson RA, Beaumont C, Nguyen MH, Lee B (2002) Interaction of metamorphism, deformation and exhumation in large convergent orogens. J Metam Geol 20:9-24

Jamtveit B, Austrheim H, Malthe-Sørenssen A (2000) Accelerated hydration of the Earth's deep crust induced by stress perturbations. Nature 408:75-78

Jeffery GB (1922) The motion of ellipsoidal particles immersed in a viscous fluid. Proc R Soc London A 102:161-179

Jessell MW (1988a) A simulation of fabric development in recrystallizing aggregates- I: Description of the model. J Struct Geol 10:771-778

Jessell MW (1988b) A simulation of fabric development in recrystallizing aggregates- II: Example model runs. J Struct Geol 10:779-793

Jessell MW, Lister GS (1990) A simulation of temperature dependence of quartz fabrics. In: Knipe RJ, Rutter EH (eds) Deformation mechanisms, rheology and tectonics. Geol Soc Lond Spec Publ 54:353-362

Jessell MW, Bons PD, Evans L, Barr T, Stüwe K (2001) Elle: a micro-process approach to the simulation of microstructures. Computers & Geosciences 27:17-30

Jessell MW, Bons PD (2002) The numerical simulation of Microstructure. Geol Soc Lond Spec Publ 200:137-147

Jessell MW, Kostenko O, Jamtveit B (2003) The preservation potential of microstructures during static grain growth. J Metam Geol 21 :481–491

Jessell MW, Siebert E, Bons PD, Evans L, Piazolo S (2005) A new type of numerical experiment on the spatial and temporal patterns of localization of deformation in a material with a coupling of grain size and rheology. Earth Planet Sci Letts 239:309-326

Jezek J (1994) Software for modelling the motion of rigid triaxial ellipsoidal particles in viscous flow. Computers & Geosciences 20:409-424

Ji S, Zhao P (1993) Flow laws of multiphase rocks calculated from experimental data on the constituent phases. Earth Planet Sci Letts 117:181-187

Ji S, Xia B (2002) Rheology of Polyphase Earth Materials. Polytechnic International Press pp 259

Johnson SE (1990) Lack of porphyroblast rotation in the Otago schists, New Zealand: implications for crenulation cleavage development, folding and deformation partitioning. J Metam Geol 8:13–30

Johnson SE (1992) Sequential porphyroblast growth during progressive deformation and low-P high-T (LPHT) metamorphism, Cooma Complex, Australia:

The use of microstructural analysis in better understanding deformation and metamorphic histories. Tectonophysics 214:311–339

Johnson WA Mehl R (1939) Reaction kinetics in processes of nucleation and growth. Trans AIME 135:42–58

Johnson SE, Dupee ME, Guidotti CV (2006) Porphyroblast rotation during crenulation cleavage development: an example from the aureole of the Mooselookmeguntic pluton, Maine, USA. J Metam Geol 24:55-73

Johnson SE, Vernon RH, Upton P (2004) Foliation development and progressive strain-rate partitioning in the crystallizing carapace of a tonalite pluton: microstructural evidence and numerical modeling. J Struct Geol 26:1845-1865

Jordan P (1988) The rheology of polymineralic rocks – an approach. Int J Earth Sc 77:285-194

Jull M, Kelemen PB (2001) On the conditions for lower crustal convective instability. J Geophys Res 106:6423-6446

Jung H, Waff HS (1998) Olivine crystallographic control and anisotropic melt distribution in ultramafic partial melts. Geophys Res Lett 25:2901-2904

Jung WS, Ree JH, Park Y (1999) Non-rotation of garnet porphyroblasts and 3-D inclusion trail data: an example from the Imjingang belt, South Korea. Tectonophysics 307:381-395

Jurewicz SR, Watson EB (1984) Distribution of partial melt in a felsic system: the importance of surface energy. Contrib Mineral Petrol 85:25-29

Kallend JS, Huang YC (1984) Orientation dependence of stored energy of cold work in 50% cold rolled copper. Metal Science 18:381-385

Kassner K, Misbah C, Müller J, Kappey J, Kohlert P (2001) Phase-field modeling of stress-induced instabilities. Phys Rev E 63:036117

Keating PN (1966) Effect of invariance requirements on the elastic strain energy of crystals with application to the diamond structure, Phys Rev 145:637-645

Kenis I, Urai JL, van der Zee W, Hilgers C, Sintubin M (2005) Rheology of fine-grained siliciclastic rocks in the middle crust - evidence from structural and numerical analysis. Earth Planet Sci Letts 233:351-360

Kenkmann T (2000) Processes controlling the shrinkage of porphyroclasts in gabbroic shear zones. J Struct Geol 22:471–487

Kenkmann T, Dresen G (1998) Stress gradients around porphyroclasts: palaeopiezometric estimates and numerical modelling. J Struct Geol 20:163-173

Khachaturyan AG (2000) In: Turchi PEA, Shull RD, Gonis A (eds) The science of alloys for the 21st century: a Hume-Rothery symposium celebration. Philadelphia, TMS, pp 293-308

Kirkwood JG (1939) The skeletal modes of vibration of long chain molecules, J Chem Phys 7:506-509

Kocks UF (1976) Laws for work-hardening and low-temperature creep. J Eng, Materials and Technology 98:76-85

Kocks UF (1985) Dislocation interactions: flow stress and strain hardening. In: Proceedings of the conference to celebrate the fiftieth anniversary of the concept of dislocation in crystals: Dislocations and Properties of real Materials. The Institute of Metals, London, pp 125-143

Koehn D, Arnold J (2003) Fracturing in Polycrystalline Materials. J Virtual Explorer 15

Koehn D, Hilgers C, Bons PD, Passchier CW (2000) Numerical simulation of fibre growth in antitaxial strain fringes. J Struct Geol 22:1311-1324

Koehn D, Aerden DGAM, Bons PD, Passchier CW (2001) Computer experiments to investigate complex fibre patterns in natural antitaxial strain fringes. J Metam Geol 19:217-232

Koehn D, Arnold J, Jamtveit B, Malthe-Sørenssen A (2003) Instabilities in stress corrosion and the transition to brittle failure. Am J Sci 303:956-971

Koehn D, Arnold J, Jamtveit B, Malthe-Sørenssen A (2003) Free Surface Dissolution of Stressed Solids, J Virtual Explorer 15

Koehn D, Dysthe DK, Jamtveit B (2004) Transient dissolution patterns on stressed crystal surfaces. Geochim Cosmochim Acta 68:3317-3325

Koehn D, Arnold J, Passchier CW (2005) Fracture and vein patterns as indicators of deformation history, a numerical study. In Gapais, Brun and Cobbold (eds), Deformation Mechanisms, Rheology and Tectonics, from Minerals to the Lithosphere. Geol Soc London Spec Publ 243:11-24

Koehn D, Malthe-Sørenssen A, Passchier CW (2006) The structure of reactive grain boundaries under stress containing confined fluids. J Chem Geol 230:207-219

Kohlstedt DL, Weathers MS (1980) Deformation induced microstructures, paleopiezometers and differential stresses in deeply eroded fault zones. J Geophys Res 85:6269-6285

Koons PO, Rubie DC, Frueh-Green G (1987) The effects of disequilibrium and deformation on the mineralogical evolution of quartz diorite during metamorphism in the eclogite facies. J Petrol 28:679-700

Kubin LP, Canova G (1990) The formation of ordered dislocation microstructes. In Messerschmidt U, Appel F (eds) Electron microscopy in plasticity and fracture research of materials. Proc Int Symposium, Dresden, Akademie-Verlag, Berlin, pp 23-32

Kubin LP, Canova G, Condat M, Devincre B, Pontikis V, Brechet Y (1992) Dislocation Microstructures and Plastic Flow: A 3D Simulation. Nonlinear Phenomena in Materials Science. Solid State Phenomena 23&24:455-472

Kuhl E, D'Addetta GA, Leukart M, Ramm E (2001) Microplane modelling and particle modelling of cohesive-frictional materials. In Vermeer PA, Diebels S, Ehlers W, Herrmann HJ, Luding S, Ramm E (eds) Continuous and discontinuous modelling of cohesive frictional materials, Lecture Notes in Physics, 568, Springer

Kunaver U, Kolar D (1988) Three-dimensional computer simulation of anisotropic grain growth in ceramics. Acta Materialia 46:4629-4640

Kuznetsov AR, Gornostyrev YN, Katsnelson MI, Trefilov AV (2001) Effect of the dislocations on the kinetics of a martensitic transition MD simulation of bcc-hcp transformation in Zr. Mat Sci Eng A309-310:168-172

Lallemant HGA (1985) Subgrain rotatin and dynamic recrystallization of olivine, upper mantle diapirism, and extension of the basin-and-range province. Tectonophysics 119:89-117

Landau LD, Lifshitz EM (1959) Quantum Mechanics. Pergamon Press, Oxford

Laporte D, Watson EB (1995) Experimental and theoretical constraintson melt distribution in crustal sources: the effect of crystalline anisotropy on melt interconnectivity. Chem Geol 124:161–184

Laporte D, Rapaille C, Provost A (1997) Wetting angles, equilibrium melt geometry, and the permeability threshold of partially molten crustal protoliths. In: Bouchez JL (ed) Granite: From segregation of melt to emplacement fabrics, Kluwer Academic Publishers

Lasaga AC (1998) Kinetic theory in the earth sciences. Princeton Univ Press, Princeton

Lebensohn RA (2001) N-site modelling of a 3D viscoplastic polycrystal using Fast Fourier Transform. Acta Materialia 49:2723-2737

Le Bouar Y, Loiseau A, Khachaturyan AG (1998) Origin of chessboard-like structures in decomposing alloys. Theoretical model and computer simulation. Acta Materiala 46: 2777-2788

Leiss B, Ullemeyer K, Weber K, Brokmeier HG, Bunge H-J, Drury M, Faul U, Fueten F, Frischbutter A, Klein H, Kuhs W, Launeau P, LLoyd GE, Prior DJ, Scheffzük Ch, Weiss T, Walther K, Wenk H-R (2000) Recent developments and goals in texture research of geological materials. J Struct Geol 22:1531-1540

Li QK, Zhang Y, Chu WY (2002) Molecular dynamics simulation of stress corrosion cracking in Cu_3 Au. Computational Materials Science 25:510-518

Li M, Chu WY, Gao KW, Qiao LJ (2004) Molecular dynamics simulation of healing of an ellipsoid crack in copper under compressive stress. Materials Letters 58:543-546

Lister GS (1981) The effect of the basal-prism mechanism switch on fabric development during plastic deformation of quartzite. J Struct Geol 3:67-75

Lister GS (1982) A vorticity equation for lattice reorientation during plastic deformation. Tectonophysics 82:351-366

Lister GS, Paterson MS (1979) The simulation of fabric development during plastic deformation and its application to quartzite: fabric transitions. J Struct Geol 1:99-115

Lister GS, Hobbs BE (1980) The simulation of fabric development during plastic deformation and its application to quartzite; the influence of deformation history. J Struct Geol 2:355-370

Lister GS, Paterson MS, Hobbs BE (1978) The simulation of fabric development in plastic deformation and its application to quartzite: the model. Tectonophysics 45:107-158

LLoyd GE, Freeman B (1991) SEM electron channelling analysis of dynamic recrystallization in quartz. J Struct Geol 13:945-953

Lloyd GE, Law RD, Mainprice D, Wheeler J (1992) Microstructural and crystal fabric evolution during shear zone formation. J Struct Geol 14:1079-1100

Lowenstern JB (1995) Applications of silicate melt inclusions to the study of magmatic volatiles. In: Thompson JFH (ed) Magmas, Fluid and Ore Deposits. Mineralogical Association of Canada Short Course 23:71-99

Malamud BD, Turcotte DL (1999) Self-organized criticality applied to natural hazards. Natural Hazards 20:93-116

Malthe-Sørenssen A, Jamtveit B, Meakin P (2006) Fracture Patterns Generated by Diffusion Controlled Volume Changing Reactions. Phys Rev Letts 96: 245501.1-4

Malthe-Sørenssen A, Walmann T, Jamtveit B, Feder J, Jøssang T (1998a) Modeling and characterization of fracture patterns in the Vatnajokull glacier. Geology 26:931-934

Malthe-Sørenssen A, Walmann T, Feder J, Jøssang T, Meakin P, Hardy HH (1998b) Simulation of extensional clay fractures. Phys Rev E 58:5548-5564

Malthe-Sørenssen A, Walmann T, Jamtveit B, Feder J, Jøssang T (1999) Simulation and Characterization of Fracture Patterns in Glaciers, J Geophys Res B104 23:157-23.174

Mancktelow NS, Arbaret L, Pennacchioni G (2002) Experimental observations on the effect of interface slip on rotation and stabilisation of rigid particles in simple shear and a comparison with natural mylonites. J Struct Geol 24:567–585

Martin JW, Doherty RD (1976) Stability of microstructure in metallic systems. Cambridge Univ Press

Marques FO, Coelho S (2001) Rotation of rigid elliptical cylinders in viscous simple shear flow: analogue experiments. J Struct Geol 23:609-617

Marques FO, Coelho S (2003) 2-D shape preferred orientations of rigid particles in transtensional viscous flow. J Struct Geol 25:841-854

Masuda T, Michibayashi K, Ohta H (1995) Shape preferred orientation of rigid particles in a viscous matrix: re-evaluation to determine kinematic parameters of ductile deformation. J Struct Geol 17:115-129

Maurice C (2001) Numerical modelling of grain growth: current status, In: Gottstein G, Molodov DA (eds) Recrystallization and Grain Growth, Proceedings of the First Joint International Conference, Springer-Verlag, Berlin, pp 123-134

McCrone WC, Chen PT (1949) Grain growth in octochloropropane. J Appl Phys 20:230-231

McKenzie D (1984) The generation and compaction of partially molten rock. J Petrol 25:713–765

McNamara AK, Karato SI, van Keken PE (2001) Localization of dislocation creep in the lower mantle: implications for the origin of seismic anisotropy. Earth Planet Sci Lett 191:85-99

Meakin P (1988) Simple models for colloidal aggregation, dielectric breakdown and mechanical breakdown patterns. In Stanley HE, Ostrowsky N (eds) Random Fluctuations and Pattern Growth, Kluwer, Dordrecht

Means WD (1983) Microstructure and micromotion in recrystallization flow of octachloropropane, a first look. Geologische Rundschau 72:511-528

Mecking H, Kocks U (1981) Kinetics of flow an strain-hardening. Acta Metall 29:1865-1875

Mehnert K, Klimanek P (1996) Monte Carlo Simulation of Grain Growth in Textured Metals Using Anisotropic Grain Boundary Mobilities. Computational Materials Science 7:103-108

Merimaa J, Perondi LF, Kaski K (2000) An interactive simulation program for visualizing complex phenomena in solids. Computer Phys Comm 124:60-75

Misch P (1970) Paracrystalline microboudinage in a metamorphic reaction sequence: Bull Geol Soc America 81:2483-2486

Mishin Y, Farkas D, Mehl MJ, Papaconstantopoulos DA (1999) Interatomic potentials for monoatomic metals from experimental data and ab initio calculations. Phys Rev B 59: 3393-3407

Mohles V (2004) The critical resolved shear stress of single crystals with long-range ordered precipitates calculated by dislocation dynamics simulations. Mat Sci Eng A365:144-150

Molnar P, Houseman GA (2004) The effects of buoyant crust on the gravitational instability of thickenend mantle lithosphere at zones of intracontinental convergence. Geophys J Int 158:1134-1150

Mueller W, Aerden D, Halliday A (2000) Isotope dating of strain fringe increments: Duration and rates of deformation in shear zones. Science 288:2195-2198

Mühlhaus HB, Sakaguchi H, Moresi L, Fahey M (2001) Discrete and continuum modelling of granular materials. In Vermeer PA, Diebels S, Ehlers W, Herrmann HJ, Luding S, Ramm E (eds) Continuous and discontinuous modelling of cohesive frictional materials, Lecture Notes in Physics 568, Springer

Nabarro FRN (1948) Deformation of crystals by the motion of single ions. In: Mott NF (ed) Report of a Conference on Strength of Solids, The Physical Society, London, pp 75-90

Nakamura M, Yurimoto H, Watson BE (2005) Grain growth control of isotope exchange between rocks and fluids. Geology 33:829-832

Neil EA, Houseman GA (1997) Geodynamics of the Tarim Basin and the Tian Shan. Tectonics 16:571-584

Nollet S, Urai JL, Bons PD, Hilgers C (2005) Numerical simulations of polycrystal growth in veins. J Struct Geol 27:217–230

Nordmeier V (1999) Zugänge zur nichtlinearen Physik am Beispiel fraktaler Wachstumsprozesse: ein generisches Fraktalkonzept. LIT-Verlag

Norris AN, Vemula C (1998) Crevice formation in thin plates by stress driven mass rearrangement. In: Tong P, Zhang TY, Kim JK (eds) Fracture and strength of solids. Key Engineering Materials, pp 145-149

Offerman SE, van Dijk NH, Sietsma J, Grigull S, Lauridsen EM, Margulies L, Poulsen HF, Rekveldt MTh, van der Zwaag S (2002) Grain Nucleation and Growth During Phase Transformations. Science 298:1003-1005

Oliver NHS, Bons PD (2001) Mechanisms of fluid flow and fluid-rock interaction in fossil metamorphic-hydrothermal systems inferred from vein-wallrock patterns, geometry, and microstructure. Geofluids 1:137-163

Ord A (1988) Deformation texture development in geological materials. in: Kallend JS, Gottstein G (eds) Proceedings of the 8th International Conference on Textures of Materials. The Metallurgical Society, Warrendale, Penn., pp 765-776

Osetsky YN, Bacon DJ, Serra A, Singh BN, Golubov SI (2000) Stability and mobility of defect clusters and dislocation loops in metals. J Nuclear Materials 276:65-77

Ostoja-Starzewski M (2002) Lattice models in micromechanics. Appl Mech Rev 55:35-60

Ostoja-Starzewski M, Sheng PY, Alzebdeh K (1996) Spring network models in elasticity and fracture of composites and polycrystals. Comput Mater Sci 7:82-93

Owen DRJ, Hinton E (1980) A simple guide to finite elements. Pineridge Press, Swansea

Paulcke, 1912 Das Experiment in der Geologie. Festschrift zur Feier dess Geburtstages Seiner Königl Hoheit des Großherzogs

Pauli C, Schmid SM, Heilbronner RP (1996) Fabric domains in quartz mylonites: Localized three dimensional analysis of microstructure and texture. J Struct Geol 18:1183-1203

Peaceman D, Rachford H (1955) The numeric solution of parabolic and elliptic equations. J Soc Industrially Appl Math 3:28-41

Pan J, Cocks ACF (1993) Computer simulation of superplastic deformation: Computational Materials Science 1:95-109

Park Y, Means WD (1996) Direct observation of deformation processes in crystal mushes. J Struct Geol 18:847-858

Park Y, Park D, Evans L , Ree JH (2003) An Elle-based 2-D model for cation exchange reaction between garnet and biotite, J Virtual Explorer 14

Passchier CW (1987) Stable positions of rigid objects in non-coaxial flow–a study in vorticity analysis. J Struct Geol 9:679-690

Passchier CW, Simpson C (1986) Porphyroclast systems as kinematic indicators. J Struct Geol 8:831–843

Passchier CW, Trouw RAJ (2005) Microtectonics. Springer, Berlin

Passchier CW, Trouw RAJ, Zwart HJ, Vissers RLM (1992) Porphyroblast rotation: eppur si muove? J Metam Geol 10:283–294

Passchier CW, ten brink CE, Bons PD, Sokoutis D (1993) Delta objects as a gauge for stress sensitivity of strain rate in mylonites. Earth Planet Sci Letts 120:239-245

Paterson MS (1973) Non-hydrostatic thermodynamics and its geologic applications: Rev Geophys Space Phys 11:355-389

Pennacchioni G, Di Toro G, Mancktelow NS (2000) Strain-insensitive preferred orientation of porphyroclasts in Mont Mary mylonites. J Struct Geol 23: 1281-1298

Pennacchioni G, Fasolo L, Cecchi MM, Salasnich L (2000) Finite element modelling of simple shear flow in Newtonian and non-Newtonian fluids around a circular rigid particle. J Struct Geol 22:683–692

Pennock GM, Drury MR, Spiers CJ (2005) The development of subgrain misorientations with strain in dry synthetic NaCl measured using EBSD. J Struct Geol 27:2159-2170

Piazolo S (2001) Shape fabric development during progressive deformation. Berichte aus der Geowissenschaft (PhD thesis), Shaker Verlag, Aachen

Piazolo S, Bons PD, Jessell MW, Evans L, Passchier CW (2002) Dominance of microstructural processes and their effect on microstructural development; insights from numerical modelling of dynamic recrystallization. Geol Soc London Spec Publ 200:149-170

Piazolo S, Passchier CW (2002) Experimental modeling of viscous inclusions in a circular high-strain shear rig: Implications for the interpretation of shape fabrics and deformed enclaves, J Geophys Res 107 doi:10.1029/2000JB000030,

Piazolo S, Jessell MW, Prior DJ, Bons PD (2004) The integration of experimental in-situ EBSD observations and numerical simulations: a novel technique of microstructural process analysis. J Microscopy 213:273-284

Poirier JP (1980) Shear localization and shear instability in materials in the ductile Field. J Struct Geol 2:135-142

Poirier JP, Nicolas A (1975) Deformation-induced recrystallization due to progressive misorientation of subgrains with special reference to mantle peridotites. J Geol 83:707-720

Pollard DD, Segall P (1987) Theoretical displacements and stresses near fractures in rocks: with applications to faults, joints, dikes and solution surfaces. In: Atkinson BK (ed) Fracture mechanics of Rock, Academic Press, London, pp 277-348

Porter DA, Easterling KE (1992) Phase Transformations in Metals and Alloys: Second Edition. Stanley Thornes (Publishers) Ltd, Chetham

Powell R, Holland TJB (1985) An internally consistent thermodynamic dataset with uncertainties and correlations: 1: Methods and worked example. J Metam Geol 3:327-342

Powell R, White L (1995) Diffusive equilibration between minerals during cooling: an analytical extension to Dodson's equation for closure in one dimension. Geol J 30:297-305

Powell R, Holland TJB, Worley B (1998) Calculating phase diagrams involving solid solutions via non-linear equations, with examples using THERMOCALC. J Metam Geol 16:577-588

Press WH, Teukolsky SA, Vetterling WT, Flannery BP (1992) Numerical Recipes in C: The Art of Scientific Computing. Cambridge Univ Press

Price NJ, Cosgrove JW (1990) Analysis of Geological Structures. Cambridge Univ Press

Raabe D (1998) Computational Materials Science: The Simulation of Materials Microstructures and Properties. Wiley-VCH, Weinheim

Raabe D (2002) Cellular Automata In Materials Science With Particular Reference To Recrystallization Simulation. Ann Rev of Materials Res 32:53-76

Raabe R, Becker RC (2000) Coupling of a crystal plasticity finite-element model with a probabilistic cellular automaton for simulating primary static recrystallization in aluminium. Modelling and Simulation in Mat Sci Eng 8:445-462

Rabinowicz M, Genthon P, Ceuleneer G, Hillairet M (2001) Compaction in a mantle mush with high concentrations and the generation of magma chambers. Earth Planet Sci Let 188:313–328

Radhakrishnan B, Sarma GB, Zacharia T (1998) Modeling the kinetics and microstructural evolution during static recrystallization- Monte Carlo simulation of recrystallization. Acta Materialia 46:4415-4433

Radhakrishnan B, Sarma G, Zacharia T, Kazaryan A, Wang Y (2001) Mesoscale Modeling of Abnormal Subgrain Growth. In: Srinivasan R et al. (eds) Microstructure Modeling and Prediction during Thermal Mechanical Processing. TMS, Warrendale, PA, pp 37-47

Ramsay JG (1980) The crack-seal mechanism of rock deformation. Nature 284:135–139

Ranalli G (1995) Rheology of the Earth. Chapman & Hall

Reinecke T, Bernhardt H-J, Wirth T (2000) Compositional zoning of calcite in a high-pressure metamorphic calc-schist: clues to heterogeneous grain-scale fluid distribution during exhumation. Contrib Mineral Petrol 139: 584-606

Renard F, Ortoleva P, Gratier JP (1999) An integrated mode for transitional pressure solution in sandstones. Tectonophysics 312:97-115

Renard F, Schmittbuhl J, Gratier J-P, Meakin P, Merino E (2004) Three-dimensional roughness of stylolites in limestones. J Geophys Res 109 doi: 10.1029/2003JB002555

Revil A (2001) Pervasive pressure solution transfer in a quartz sand. J Geophys Res 106:8665-8686

Rhee M, Zbib HM, Hirth JP, Huang H, de la Rubia T (1998) Models for long-/short-range interactions and cross slip in 3D dislocation simulation of BCC single crystals. Model Simul Mater Sci Eng 6:467-492

Ribe NM (1989) A continuum theory for lattice preferred orientation. Geophys J 97:199-207

Riggs MR, Green HW (2001) Shear localization in transformation-induced faulting: first-order similarities to brittle shear failure, Tectonophysics 340:95-107

Rohrer GS (2005) Influence of interface anisotropy on grain growth and coarsening. Ann Rev Materials Res 35:99-126

Roscoe R (1952) The viscosity of suspensions of rigid spheres. British J Appl Phys 3:267–269

Rosenberg CL, Handy MR (2000) Syntectonic melt pathways during simple shearing of a partially molten rock analogue (norcamphor-benzamide). J Geophys Res 105:3135-31349

Rosenfeld JL (1970) Rotated garnets in metamorphic rocks. Geol Soc America Special Paper 129:105

Rothman DH, Zaleski S (1994) Lattice-gas model of phase separation, interfaces, phase transitions, and multiphase flow. Rev Mod Phys 66:1417-1479

Rubie DC (1986) The catalysis of mineral reactions by water and restrictions on the presence of an aqueous fluid during metamorphism. Mineral Mag 50:399-415

Rutter EH (1976) The kinetics of rock deformation by pressure solution. Phil Trans Royal Soc London 283:203-219

Saada A (1993) Elasticity: theory and applications. Krieger

Samanta SK, Mandal N, Chakraborty C (2002) Development of structures under the influence of heterogeneous flow field around rigid inclusions: insights from theoretical and numerical models. Earth Sci Rev 58:85-119

Saylor DM, Mason DE, Rohrer GS (2000) Experimental Method for Determining Surface Energy Anisotropy and Its Application to Magnesia. J American Ceramic Soc 83:1226-1232

Schenk O (2004) Grain boundary structure in minerals and analogues during recrystallization in the presence of a fluid phase. Unpubl PhD, Univ of Aachen, p 155

Schenk O, Urai JL (2005) The migration of fluid-filled grain boundaries in recrystallizing synthetic bischofite: first results of in-situ high-pressure, high-temperature deformation experiments in transmitted light. J Metam Geol 23:695-709

Schmid DW, Podladchikov YY (2005) Mantled porphyroclast gauges. J Struct Geol 27:571-585

Schmidt S, Nielsen SF, Gundlach C, Margulis L, Huang X, Juul Jensen D (2004) Watching the growth of bulk grains during recrystallization of deformed metals. Science 305:229-232

Schmittbuhl J, Renard F, Gratier JP, Toussaint R (2004) The roughness of stylolites: Implications of 3D high resolution topography measurements. Phys Rev Lett 93:238501

Scholz CH (2002) The mechanics of Earthquakes and faulting, 2nd edition. Cambridge Univ Press, Cambridge

Schoneveld C (1977) A study of some typical inclusion patterns in strongly paracrystalline rotated garnets. Tectonophysics 39:453-471

Schwarz KW (1999) Simulation of dislocations on the mesoscopic scale. I. Methods and examples. J Appl Phys 85:108-119

Scott DR, Stevenson DJ (1986) Magma ascent by porous flow. J Geophys Res 91:9283–9296

Seward GGE, Celotto S, Prior DJ, Wheeler J, Pond RC (2004) In situ SEM-EBSD observations of the hcp to bcc phase transformation in commercially pure titanium. Acta Materialia, 52:821-832

Shea WT Jr, Kronenberg AK (1993) Strength and anisotropy of foliated rocks with varied mica contents. J Struct Geol 15:1097-1121

Shekar NC, Rajan KG (2001) Kinetics of pressure induced structural phase transitions - a review. Bull Materials Sci 24:1–21

Shewchuk JR (1996) Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator, First Workshop on Applied Computational Geometry (Philadelphia, Pennsylvania), pp 124-133

Shewchuk JR (2002) Delaunay Refinement Algorithms for Triangular Mesh Generation, Computational Geometry: Theory and Applications 22:21-74

Shewchuk JR (2005) http://www-2.cs.cmu.edu/~quake/triangle.poly.html

Shimizu I (1992) Nonhydrostatic and Nonequilibrium Thermodynamics of Deformable Materials. J Geophys Res 97:4587-4597

Simmons JP, Shen C, Wang Y (2000) Phase field modeling of simultaneaous nucleation and growth by explicitly incorporating nucleation events. Scripta Materialia 43:935-942

Sleep NH (1988) Tapping of melt by veins and dykes. J Geophys Res 93: 10255–10272

Smith CS (1964) Some elementary principles of polycrystalline microstructure. Met Rev 9:1-48

Soares A, Ferro AC, Fortes MA (1985) Computer simulation of grain growth in a bimodal polycrystal. Script Metall 19:1491-1496

Spear FS (1991) On the interpretation of peak metamorphic temperatures in light of garnet diffusion during cooling. J Metamorph Geol 9:379-388

Spear FS (1993) Metamorphic phase equilibria and pressure-temperature-time paths, Mineralogical Society of America, Washington, D.C. p 799

Squires RL, Weiner RT, Phillips M (1963) Grain boundary denuded zones in a magnesium 1/2 wt % zirconium alloy. Journal of Nuclear Materials 8:77-80

Srolovitz DJ (1989) On the stability of surfaces of stressed solids. Acta Metall 37:621-625

Stallard A, Shelley D (1995) Quartz c-axes parallel to stretching directions in very lowgrade metamorphic rocks. Tectonophysics 249:31-40

Stallard AR, Ikei H, Masuda T (2002) Does 3D inclusion trail geometry distinguish competing models of spiral formation? J Metam Geol 20:801-812

Steinhardt CK (1989) Lack of porphyroblast rotation in non- coaxially deformed schists from Petrel Cove, South Australia, and its implications. Tectonophysics 158:127-140

Stöckhert B, Duyster J (2000) Discontinuous grain growth in recrystallised vein quartz - implications for grain boundary structure, grain boundary mobility, crystallographic preferred orientation, and stress history. J Struct Geol 21:1477-1490

Stöckhert B, Brix MR, Kleinschrodt R, Hurford AJ, Wirth R (1999) Thermochronometry and microstructures of quartz – a comparison with experimental flow laws and prediction on the temperature of the brittle–plastic transition. J Struct Geol 21:351-369

Stüwe K (1997) Effective bulk composition changes due to cooling: a model predicting complexities in retrograde reaction texture. Contrib Mineral Petrol 129:43-52

Taber S (1916) The origin of veins of the asbestiform minerals. Proc Nat Acad Sci 2:659–664

Taborda R, Antunes J, Marques FO (2004) 2-D rotation behavior of a rigid ellipse in confined viscous simple shear: numerical experiments using FEM. Tectonophysics 379:127-137

Takeshita T, Wenk H-R, Lebensohn R (1999) Development of preferred orientation and microstructure in sheared quartzite: comparison of natural data and simulated results. Tectonophysics 312:133-155

Taylor GI (1938) Plastic Strain in Metals. J Inst Metals 62:307-324

Ten AA, Yuen DA (1999) Time-dependent formation of mantled inclusion structures for different rheologies under a simple shear. Earth Planet Sci Lett 165:25–35

ten Brink CE (1996) Development of porphyroclast geometry during non-coaxial flow. PhD-thesis Utrecht University, Geologica Ultraiectina 142

ten Brink CE, Passchier CW (1995) Modelling of mantle porphyroclasts using non-Newtonian rock analogue materials. J Struct Geol 17:131-146

Tenczer V, Stüwe K, Barr TD (2001) Pressure anomalies around rigid objects in simple shear. J Struct Geol 23:777-788

ten Grotenhuis SM, Passchier CW, Bons PD (2002) The influence of strain localisation on the rotation behaviour of rigid objects in experimental shear zones. J Struct Geol 24:485-499

Tenczer V, Stüwe K, Barr TD (2001) Pressure anomalies around cylindrical objects in simple shear, J Struct Geol 23:777-788

ter Heege JH, de Bresser JHP, Spiers CJ (2002) The influence of dynamic recrystallization oon the grain size distribution and rheological beaviour of Carrara marble deformed in axial compression. Geol Soc London Spec Publ 200:331-353

Tikoff B, Teyssier C (1994) Strain and fabric analyses based on porphyroclast interaction. J Struct Geol 16:477-491

Tóth LS, Molinari A, Raabe D (1997) Modelling of rolling texture development in a ferritic chromium steel. Metalurgical Transactions A 28:2343-2351

Treagus SH (1999) Are viscosity ratios measurable from cleavage refraction? J Struct Geol 21:895-901

Treagus SH (2002) Modelling the bulk viscosity of two-phase mixtures in terms of clast shape. J Struct Geol 24:57-76

Trimby PW, Prior DJ, Wheeler J (1998) Grain boundary hierarchy development in a quartz mylonite. J Struct Geol 20:913-935

Tullis J, Christie JM, Griggs DT (1973) Microstructures and preferred orientations of experimentally deformed quarzites. Geol Soc Am Bull 84:297-314

Tullis TE, Horowitz FG, Tullis J (1991) Flow laws of polyphase aggregates from end-member flow laws. J Geophys Res 96:8081-8096

Turnbull D (1956) Phase changes. Solid State Physics 3:225–306

Twiss RJ (1977) Theory and applicability of a recrystallized grain size paleopiezometer. Pure Appl Geophys 155:227-244

Urai JL, Means WD, Lister GS (1986) Dynamic recrystallization of minerals. In: Heard HC, Hobbs BE (eds) Mineral and rock deformation: Laboratory studies – The Paterson Volume. Am Geophys Union Geophys Monogr 36:161-199

Urai JL, Willimas PF, van Roermund HLM (1991) Kinematics of crystal growth in syntectonic fibrous veins. J Struct Geol 13:823-836

Van Houtte P (1988) A comprehensive mathematical formulation of an extended Taylor–Bishop–Hill model featuring relaxed constraints, the Renouard-Wintenberger theory and a strain rate sensitivity model. Textures and Microstructures 8-9:313-35

Vermeer PA, Diebels S, Ehlers W, Herrmann HJ, Luding S, Ramm E (2001) Continuous and discontinuous modelling of cohesive frictional materials, Lecture Notes in Physics 568, Springer

Vernon R (1981) Optical microstructure of partly recrystallised calcite in some naturally deformed marbles. Tectonophysics 78:601-612

Von Blanckenhagen B, Arzt E, Gumbsch P (2004) Discrete dislocation simulation of plastic deformation in metal thin films. Acta Materialia 52: 773-784

Wachspresse E, Habetler G (1960) An alternating-direction-implicit iteration technique. J Soc Industrially Appl Math 8:403-424

Wall ER (1988) The occurrence of staurolite and its implication for polymetamorphism in the Mt. Washington area, New Hampshire. Unpubl MSc Thesis, Univ of Maine, Orono

Walmann T, Malthe-Sørenssen A, Feder J, Jøssang T, Meakin P, Hardy HH (1996) Scaling relations for the lengths and widths of fractures, Phys Rev Letts 77:5292-5296

Walte NP, Bons PD, Passchier CW, Koehn D (2003) Disequilibrium melt distribution during static recrystallization. Geology 31:1009-1012

Waff HS, Bulau JR (1979) Equilibrium fluid distribution in an ultramafic partial melt under hydrostatic stress conditions. J Geophys Res 84:6109-6114

Waff HS, Faul UH (1992) Effects of crystalline anisotropy on fluid distribution in ultramafic partial melts. J Geophys Res 97:9003-9014

Wang YU, Jin YM, Cuitino AM, Khachaturyan AG (2001) Nanoscale phase field microelasticity theory of discolcations: Model and 3D simulations. Acta Materialia 49:1847-1857

Wang YU, Jin YM, Khachaturyan AG (2003) Phase field microelasticity modeling of dislocation dynamics near free surface and in heteroepitaxial thin films. Acta Materialia 51:4209-4223

Wark DA, Williams CA, Watson EB, Price JD (2003) Reassessment of pore shapes in microstructurally equilibrated rocks, with implications for permeability of the upper mantle. J Geophys Res 108, doi:10.1029/2001JB001575

Weaire D, Rivier N (1984) Soap, cells and statistics - Random patterns in two dimensions. Contemp Phys 25:59-99

Wenk H-R (1985) Preferred orientation in deformed metals and rocks: An introduction to modern texture analysis. Academic Press, Orlando

Wenk H-R, Canova G, Molinari A, Mecking H (1989a) Texture development in halite: comparison of Taylor model and self consistent theory. Acta Metall 37:2017-2029

Wenk H-R, Canova G, Molinari A, Kocks UF (1989b) Viscoplastic modelling of texture development in quartzite. J Geophy Res 94:17895-17906

Weyl PK (1959) Pressure solution and force of crystallisation - A phenomenological theory: J Geophys Res 64:2001-2025

White SH (1979) Grain and sub-grain size variations across a mylonite zone. Contrib Mineral Petrol 70:193-202

White SH, Burrows SE, Carreras J, Shaw ND, Humphreys FJ (1980) On mylonites in ductile shear zones. J Struct Geol 2:175-187

Willis DG (1977) A kinematic model of preferred orientation. Geol Soc Am Bull 88:883-894

Wilson CJL, Zhang Y (1996) Development of microstructure in the high temperature deformation of ice. Annals of Glaciology 23:293-302

Wilson CJL, Zhang Y, Stüwe K (1996) The effects of localized deformation on melting processes in ice. Cold Regions Science and Technology 24:177-189

Xiang Y, Cheng L-T, Srolovitz DJ, Weinan E (2003) A level set method for dislocation dynamics. Acta Materialia 51:5499-5518

Yamada Y, Ito K, Yokouchi Y, Tamano Y, Ohtsubo T (1975) Finite Element analysis of steady fluid and metal flow. In: Gallagher RH, Oden JT, Taylor C, Zienkiewicz OC (eds) Finite Elements in Fluids 1:73-94

Yang WH, Srolovitz DJ (1993) Cracklike surface instabilities in stressed solids. Phys Rev Lett 71:1593-1596

Zbib HM, Diaz de la Rubia T (2002) A multiscale model of plasticity. Int J Plasticity 18: 1133-1163

Zbib HM, Rhee M, Hirth JP (1998) On plastic deformation and the dynamics of 3D dislocations. Int J Mechanical Sci 40:113-127

Zhang J, Adams JB (2002) FACET: a novel model of simulation and visualisation of polycrystalline thin film growth. Modelling and Simulation in Mat Sci Eng 10:381–401

Zhang Z, Wilson CJL (1997) Lattice rotation in polycrystalline aggregates and single crystals with one slip system: a numerical and experimental approach. J Struct Geol 19:875-885

Zhang Y, Hobbs BE, Jessell MW (1993) Crystallographic preferred orientation development in a buckled single layer: a computer simulation. J Struct Geol 15:265-276

Zhang Y, Hobbs BE, Ord A (1994a) Numerical simulation of fabric development in polycrystalline aggregates with one slip system. J Struct Geol 16: 1297-1313

Zhang Y, Hobbs BE, Jessell MW (1994b) The effect of grain boundary sliding on fabric development in polycrystalline aggregates. J Struct Geol 16:1315-1325

Zhang Y, Jessell MW, Hobbs BE (1996) Experimental and numerical studies of the accommodation of strain incompatibility on the grain scale. J Struct Geol 18:451-460

Zhou SJ, Grønbech-Jensen N, Bishop AR, Lomdahl PS, Holian BL (1997) A nonlinear-discrete model of dynamic fracture instability. Physics Letters A 232:183-188

Zienkiewicz OC, Taylor RL (1977) The finite element method. McGraw-Hill, London.

Appendices

A	**Installing *Elle***
B	**Examples experiments**
C	**The *Elle* graphical user interface**
D	**Different ways to run an *Elle* experiment**
E	**How to create multiple-process experiments**
F	***Sybil*, the *Basil* post-processor**
G	**How to use *elle_latte***
H	**Miscellaneous processes and utilities**
I	**How to create an *Elle* file**
J	**The *Elle* file format**

These Appendices describe the steps necessary to run arbitrary single-processes or multi-process *Elle* experiments. Currently *Elle* runs under Windows and Linux, and as the code is continuously being updated, both to fix bugs and add features, **it is recommended that you visit the *Elle* web site to download the latest releases of the *Elle* platform (http://www.microstructure.info/elle)**; however the description here applies specifically to the code released on the CD which accompanies this book.

In the appendices we use the following formats to distinguish menu items, programs and files:

• **File/Open**	Menu item
• *Program*	Name of software or modules
• <u>File</u>	Name of a file or URL

READERS WHO ONLY WISH TO RUN THE EXAMPLE EXPERIMENTS DESCRIBED IN THIS BOOK SHOULD INITIALLY REFER TO APPENDICES A, B & C.

Appendix A Installing *Elle*

Mark W. Jessell

A.1 Windows

To install the *Elle* platform on your Windows computer, open the CD and double click on the **ElleWinSetup** icon. There are two installation levels for the platform: User and Developer.

1. Users are those people who wish to run or modify example experiments, and who will have access to the high-level executables, scripts and input files.
2. Developers are those people who wish to alter the codes to describe new processes, who will have also have a complete GCC compiler, a simple Linux-like environment (MSys) and the complete *Elle* source code installed on their system. The Linux-like environment runs under Windows and does not interfere with the normal operation of the operating system. If you choose this option, you can recompile the *Elle* platform at any time by opening an MSys shell, then change to the directory elle and type in `./install.win wx > debug.txt&`

For both Users and Developers, *Sybil*, the graphics postprocessor for the *Basil* Finite Element program, will also be installed at this stage, which uses an Open Source X Windows system, running under *Cygwin*.

After successful installation, all example experiments (Appendix B) are copied over to your harddisk, so you will not need the CD again after installation. On your harddisk a folder elle/ is created containing the following files and folders:

- elle/, a folder containing all code and other elle-related material
- experiments/, a folder containing all examples of Appendix B
- extras/, a folder containg some colour maps, elle input files, etc.
- COPYING.txt, a general public licence
- LICENCE.txt, the *Elle* licence
- Internet shortcut, a link to the *Elle* website
- unins000.exe & unins00.dat, files to un-install *Elle*

A.1.1. Installing *Basil* and *Sybil*

Basil **-** The Finite Element deformation program, *Basil,* will be installed with the *Elle* binaries when you run ElleWinSetup.exe.

Sybil - This is an X11 application -(Appendix F). For instructions on how to install this program, which requires the *Cygwin* system to also be installed on your PC, please go to the *Elle* Website (http://www.microstructure.info/elle) and select the menu "**Sybil**". If you already have *Cygwin* installed, or once you have it installed, you may install *Sybil* by running SybSetup.exe. The *Experiment Launcher* expects sybil.exe and sybilps.exe to be on the same drive as Cygwin and in the directory DRIVE:\cygwin\home\sybil\. If you install it in a different location, you will have to edit sybil.bat and sybilps.bat in the binwx directory of the Elle installation.

If you have a firewall or something similar installed under Windows, you have to allow *Cygwin* access to the internet. Internally it uses the network (a so-called loopback-device) to send messages. This is safe to do, as no data will be downloaded to your computer or sent to the internet.

A.2 Linux

To install the *Elle* platform on your Linux computer (for both **Users** and **Developers**):

1. open the CD;
2. copy the directory called elle to your hard disk;
3. open up a shell window, and then change to the directory elle
   type in ./install.sh wx > debug.txt&

This will compile the complete *Elle* platform on your computer (and the *Basil* Finite Element package), assuming that all the correct libraries have been installed on your system.

To install *Elle* you will require the following libraries to be installed

- WXWidgets installed using the --with-gtk option (http://www.wxwidgets.org and comes with all standard Linux distributions).
- GTK+ (http://www.gtk.org and comes with all standard Linux distributions)
- GCC with Fortran (comes with all standard Linux distributions)

If the compilation does not run to completion (you do not get a directory created called binwx full of files), you should check the debug.txt to see what the errors are, and refer to the Installation Forum at the *Elle* Website: http://www.microstructure.info/elle.

A.2.1. Installing *Basil* and *Sybil*

Pre-built binaries can be found at the *Elle* website, www.microstructure.info/Elle. Contact basil@earthsci.unimelb.edu.au or one of the authors if you are unable to find an appropriate package.

Sybil (Appendix F) uses X11 and Motif libraries and you may need to install the Motif libraries on your system. These are available on the web e.g www.lesstif.org or www.openmotif.org. These applications do not need to be installed in a particular location but your environment variable BASILPATH should be set such that $BASILPATH/bin is the location of the binaries (*Basil*, *Sybil* and *sybilps*) and $BASILPATH/bin should be added to your $PATH variable.

Appendix B Example experiments

Edited by Mark W. Jessell

After installation, all experiment files are found within the elle/experiments directory on your hard disk. This is necessary, because Elle cannot write output files onto the CD. The input files for the experiments described in this Appendix (see Table B.1), for example Experiment 3, are contained in a directory whose name starts with experiment_03… Where there are more than one set of input files for an experiment, there will be a series of subdirectories, e.g. **a, b, c** etc. To launch an example experiment without modification, use the *Experiment Launcher*, (elle/elle/binwx/experiment_launcher.exe under Windows).

The *Experiment Launcher*

The *Experiment Launcher* is a simple program that allows you to launch any of the example experiments in this Appendix. Once started up, the Launcher offers you a single menu: the **File menu**, and two panels the **Experiments panel**, the **Utilities panel.** The **Experiment panel** and its submenus list all the experiments in this Appendix. To run a particular example, select it from the **Experiment panel** and you will be provided with a brief overview of the experiment. If you wish to start the experiment, simply click on the **Go!** button.

Single-Process Experiments: If a single-process was started, an *Elle* window will open up showing the starting microstructure. To enact the experiment select **Run→Run** from within the *Elle* Window. See Appendix C for details on the graphical user interface for single processes.

Important: to kill a single process experiment under windows, use the task manager (in Windows press CTRL-ALT-DEL), rather than clicking on the $\boxed{\text{x}}$-icon, as otherwise the calculation will continue in the background!

Multi-Process Experiments: If a multi-process was started, the experiment will be started automatically in the background, and to view any of the output *Elle* files, select *showelle* from the **Utilities panel** of the Launcher, and load in the desired file. See Appendix C for details on the graphical user interface for *showelle*. All output files will be created in the same directory as the experiment, and have a naming convention described in Appendix E.

Important: to kill a multi-process experiment under windows, select the command-line window and type CTRL-C.

Starting Single Processes or Utilities: The *Experiment Launcher* may also be used to launch utilities such as *showelle*, without any input file pre-loaded.

The duration of individual experiments is highly variable and denoted with icons for "fast" (➤), "medium" (☕), "slow" (🚶), to "very slow" (🛏).

Table B.1. List of example experiments, and the sections in the book that describe them in more detail.

Experiment number	Section in book	Title
1	2.2	Diffusion*
2	2.2	Fluid flow in a porous medium*
3	2.6	Crystal growth from melt
4	3.12	Fracturing in granular aggregates*
5	3.2	Cation exchange reactions
6	3.3	Subgrain growth
7	3.5	Grain growth
8	3.7	Evolution of a partial melt
9	3.8	Rigid porphyroblast growing in a deforming matrix
10	3.9	Lattice rotations
11	3.11	Boudinage*
12	3.12	Dissolution grooves*
13	3.12	Stylolites*
14	4.8	Strain-rate partitioning during porphyroblast growth
15	4.1	Anisotropic grain growth
16	4.2	Dynamic recrystallisation
17	4.3	Deformation localisation
18	4.4	Expanding inclusions*
19	4.5	Mud cracks*
20	4.6	Visco-elastic deformation and fractures*
21	4.7	Strain localization and rigid object kinematics

* Note: Experiments that use *elle_latte* will not show the starting microstructure until the experiment is actually started (with **run→run**). The stage number shown on the display does not necessarily equal the number of time steps. Also note that *elle_latte* experiments cannot be re-run.

Although we have done our best to test and debug these examples, there may still be some bugs. If you encounter problems, please check the *Elle* website (http://microstructure.info/elle).

Experiment 1 - Diffusion

In this example we demonstrate diffusion through a fluid trapped in a granular medium using a simple Lattice-Gas automaton that uses the FHP rules as its basis and is mapped on a triangular grid (see Chap. 2.2). The algorithm is included in *elle_latte*. These experiments look "noisy" because we are looking at very small areas of material. Hence the random fluctuations are much more evident than in larger-scale types of models, such as those that use Finite Difference calculations.

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. On most computers you can use the higher-resolution files since the algorithm is relatively fast. Possible sub-experiments, which simply vary the density of unodes are:

- **a** diffusion in a system with 2850 particles
- **b** diffusion in a system with 11500 particles
- **c** diffusion in a system with 46000 particles

Once the file is loaded, select **Run** from the **Run menu** of the *Elle* window to watch the system evolve.

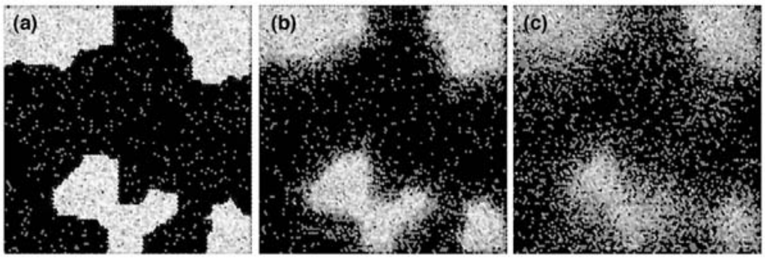


Fig. B1. Modelling diffusion with a Lattice-Gas automaton (experiment **b**). Time steps are **(a)** 1, **(b)** 10, **(c)** 40. High concentration is bright and low concentration black

Interface: These files will each load their appropriate preferences file automatically. The resulting plots show you the concentration of fluid particles per unode. Unodes that are part of a grain, and which are therefore not permeable, have a concentration of 6 to visualize the pore space.

Examples: Figure B1 shows plots of an example run that illustrates how diffusion can be modelled with a Lattice-Gas automaton. The input file is res100.elle. Plotted are the concentrations of fluid particles where the background is dark, and the brightness increses with concentration of fluid particles (U_DENSITY). The initial configuration is of four regions with a high concentration embedded in a background with a zero concentration.

Functions and parameters for users wishing to use the Experiment Interface to modify parameters: In the initialization function (Experiment::Init(), case 12) the following functions contain parameters that can be changed:

- SetFluidLatticeGasRandom(0.005): This function is used to set randomly distributed fluid particles in the background. The input variable specifies the concentration of fluid particles in the whole *Elle* box where 0.005 is 0.5%.
- SetFluidLatticeGasRandomGrain(0.7,j*10): This function specifies the fluid particle concentration in a specific grain. The first input value determines the fluid concentration (0.7 is 70%) and the second variable the specific grain (in this case a loop where every tenth grain is chosen).
- SetWallsLatticeGas(j*5): This function is used for the creation of porosity in the 2d fluid flow. All grains (j*5) are set to be impermeable so that a complex permeability structure evolves.

In the run function (Experiment::Run(), case 12) the following functions contain parameters that can be changed:

- UpdateFluidLatticeGas(): simply does a Lattice-Gas step where particles are transported and collisions are handled.
- InsertFluidLatticeGas(0.03, 0.9): inserts fluid particles at the left hand side of the box, every lattice particle with x position smaller than 0.03 becomes each time step a fluid concentration of 90 percent (second input value, 0.9).
- RemoveFluidLatticeGas(0.97): removes all fluid particles from lattice particles with an x position that is larger than 0.97.

Experiment 2 - Fluid flow in a porous medium

In this example we demonstrate fluid flow in porous medium using a simple Lattice-Gas automata that uses the FHP rules as basis and is mapped on a triangular grid (see Chap. 2.2). The algorithm is included in ***elle_latte***.
Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. On most computers you can use the higher-resolution files since the algorithm is relatively fast. Possible sub-experiments, which simply vary the density of unodes are:

a fluid flow in a system with 2850 particles
b fluid flow in a system with 11500 particles
c fluid flow in a system with 46000 particles

Once the file is loaded, select **Run** from the **Run menu** of the *Elle* window to watch the system evolve. It is a good idea to start with experiment **a**, as this gives a detailed expression of how the system behaves.

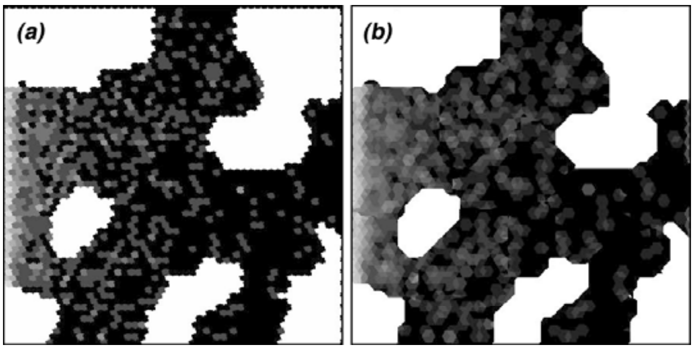


Fig. B2. Simulation of fluid flow through a porous medium. Fluid comes in at the left hand side and exits at the right hand side. Background is black, wall rock is white and fluid particles are bright. **(a)** Unodes are shown assmall hexagons whose brightness is a function of concentration. **(b)** Concentrations are interpolated between unodes when the option **triangulate unodes** is turned on in the **Graphics→Preferences→Unode** menu

Interface: These files will each load their appropriate preferences file automatically. The resulting plots show you the concentration of fluid particles per unode. Unodes that are part of a grain, and which are therefore not permeable, have a concentration of 6 to visualize the pore space.

Examples: This example is shown in Fig. B2. Here 2D porous flow is modelled with a Lattice-Gas automaton. The input file is res50.elle (experiment **a**) with a low resolution. Particle concentration on the left boundary is kept constant, while all particles reaching the right boundary of the system are removed. This effectively models fluid that is injected from the left hand side and that can escape on the right hand side of the model.

Functions and parameters for users wishing to use the Experiment Interface to modify parameters: In the initialization function (Experiment::Init(), case 13) the following functions contain parameters that can be changed:

- SetFluidLatticeGasRandom(0.005): This function is used to set randomly distributed fluid particles in the background. The input variable specifies the concentration of fluid particles in the whole *Elle* box where 0.005 is 0.5%.
- SetFluidLatticeGasRandomGrain(0.7,j*10): This function specifies the fluid particle concentration in a specific grain. The first input value determines the fluid concentration (0.7 is 70%) and the second variable the specific grain (in this case a loop where every tenth grain is chosen).
- SetWallsLatticeGas(j*5): This function is used for the creation of porosity in the 2d fluid flow. All grains (j*5) are set to be impermeable so that a complex permeability structure evolves.

In the run function (Experiment::Run(), case 13): the following functions contain parameters that can be changed:

- UpdateFluidLatticeGas(): simply does a Lattice-Gas step where particles are transported and collisions are handled.
- InsertFluidLatticeGas(0.03, 0.9): inserts fluid particles at the left hand side of the box. Each time step every lattice particle with an *x* coordinate smaller than 0.03 gets a fluid concentration of 90% (second input value, 0.9).
- RemoveFluidLatticeGas(0.97): removes all fluid particles from lattice particles with an *x* position that is larger than 0.97.

Experiment 3 - Crystal growth from melt

This example illustrates the growth of a single crystal from a melt, using a Phase Field approach. The background of the software (from Biben 2005) is explained in Chap. 2. In the following examples we can vary the crystal symmetry and the latent heat of crystallisation.

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. Possible sub-experiments, which vary the crystal symmetry 6-fold or (the physically extremely rare!) 5-fold symmetry; or the latent heat, are:

- **a** 6-fold symmetry, relatively high latent heat
- **b** 5-fold symmetry, relatively high latent heat
- **c** 6-fold symmetry, relatively low latent heat
- **d** 5-fold symmetry, relatively low latent heat

Interface: These files will each load their appropriate preferences file automatically. Once the file is loaded, select **Run** from the **Run menu** of the *Elle* window to watch the system evolve. You can either display the concentration (CONC_A) of the unodes (the default) or the temperature (U-ATTRIB-A). In order to speed up the experiment, only every 10-th time step is displayed.

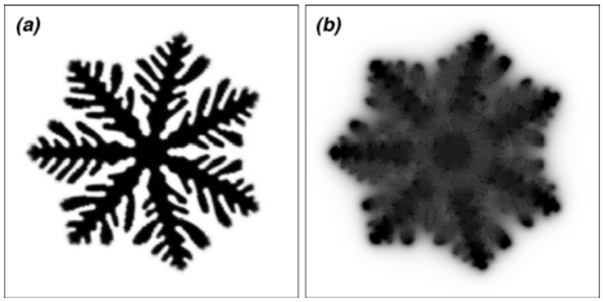


Fig. B3. Phase Field simulation of the growth of a crystal from a melt after 2000 time steps. **(a)** Shape of crystal (CONC_A) and **(b)** temperature (U_ATTRIB_A) field. The simulation shown is based on experiment **a**, but the Anisotropy was changed to achieve an unusual 7-fold symmetry (Select **Run->Run Options** then edit the **Anisotropy** field in the **UserData** area)

Functions and parameters for users wishing to modify runtime parameters: The default userdata settings for *elle_phasefield* are shown below:

- userdata[0] default= 0 (reset unode values if we want to create a seed crystal from a file that has no seed already defined)
- userdata[1] default = 1.8 (high latent heat) or 0.18 (low latent heat)
- userdata[2] default = 0.01 (interfacial width)

- userdata[3] default = 0.02 (modulation of the interfacial width)
- userdata[4] default = 1.57 (orientation of the anisotropy axis)
- userdata[5] default = 6.0 (6-fold symmetry) or 5.0 (5-fold symmetry)
- userdata[6] default= 0.9
- userdata[7] default = 10
  (where m(T)=ALPHA/PI*atan(GAMMA*(TEQ-T)))

Experiment 4 - Fracturing in granular aggregates

This example illustrates the development of fractures in a granular aggregate during compaction and pure shear deformation. The background of the software is explained in Chap. 3.12 and is based on a Lattice-Spring model (Chap. 2.8).

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. On most computers you should use the lower-resolution files since the algorithm is relatively slow. Possible sub-experiments, which simply vary the density of unodes are:

a 2850 particles ☕
b 11500 particles ☕
c 46000 particles ☕

Once the file is loaded, select **Run** from the **Run menu** of the *Elle* window to watch the system evolve.

Interface: These files will each load their appropriate preferences file automatically, and only the unodes will be visible. You can now choose to view the Young's moduli (U_YOUNGSMODULUS) of different grains, the fractures (U_FRACTURES), the mean stress (U_MEAN_STRESS), or the differential stress (U_DIFFSTRESS, default). In order to view fractures select **Graphics->Preferences** then click on the **Unodes** tab and select and set **Clamp color between** -1.0 and 0.0 to limit the display range. Unodes with fractured bonds will be blue and unfractured unodes red. Differential and mean stress should be scaled during a run, normal values are: differential stress 0.0 to 0.01 and mean stress - 0.01 to 0.01 (negative is compression).

Examples: Figure B3 (a) to (d) shows the four different plotting options. Figure B3a shows the distribution of Young's moduli of different grains where dark grey is high Young's modulus (range 0.0 to 1.7). Figure B3b shows the first fractures after 7 *Elle* steps where dark are fractured particles (range is –1.0 to 0.0). Figure B3c shows the differential stress where dark is high and light colour low stress (range 0 to 0.02) of stage 7 and Fig. B3d shows the mean stress of the same stage where dark is high stress (range –0.01 to 0.005, negative is high compressive stress). Figure B3e shows an example with 5 steps of uniaxial deformation followed by 15 steps of pure shear deformation whereas Fig. B3f shows an example with 15 steps of uniaxial deformation followed by 5 steps of pure shear deformation. Plots show again the differential stress (range 0-0.035). The second example that has experienced a longer uniaxial compression builds up more compressive stress and therefore develops more pronounced shear fractures whereas the first example shows a combination of extension and shear fractures. Comparing Fig. B3a and B3b one can also see that regions with high Young's moduli are fracturing first.

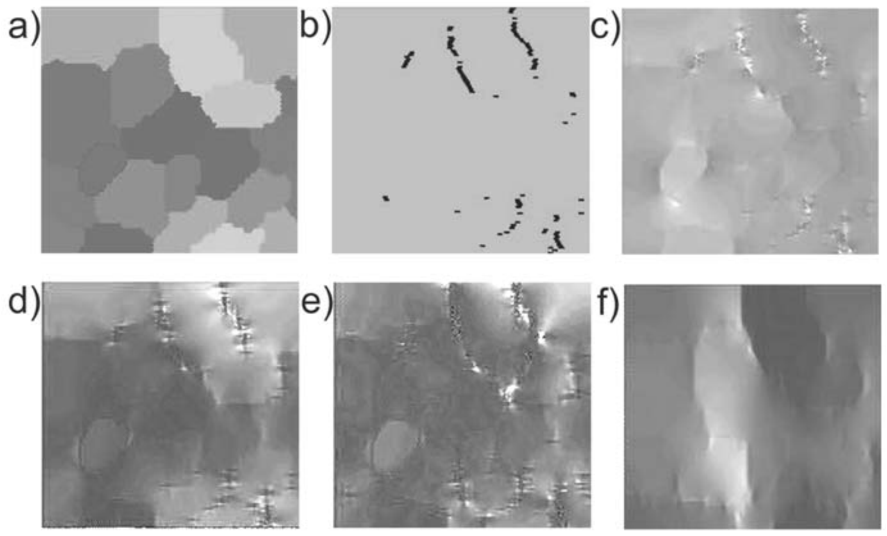


Fig. B3. Examples of fracture development in granular aggregates. **(a)** to **(d)** plotting options showing different unode attributes **(a)** U_YOUNGSMODULUS: Young's moduli of grains, **(b)** U_FRACTURES: fractured particles, **(c)** U_DIFFSTRESS differential stress and **(d)** UMEAN_STRESS: mean stress. **(e)** Differential stress after 5 stages of uniaxial compaction followed by 15 steps of pure shear deformation. **(f)** Differential stress after 15 stages of uniaxial compaction followed by 5 steps of pure shear deformation

Functions and parameters for users wishing to use the Experiment Interface to modify parameters: Parameters used in the example are:

- SetPhase(0.0,0.0,2.0,1.2),
- SetGaussianSpringDistribution(1.0,0.5),
- MakeGrainBoundaries(1.0,0.5),
- DeformLattice(0.001,1), and
- DeformLatticePureShear(0.001,1).

In the initialization function (`Experiment::Init()`, case 1) the following functions contain parameters that can be changed:

- SetPhase(0.0,0.0,2.0,1.2) is used to set a distribution on breaking strengths of springs. The breaking strengths will be distributed randomly between original strength - 0.6 * original strength and original strength +0.6 * original strength (0.6 is value 1.2/2 in function). The whole distribution will be multiplied by 2.0 (first value in function). Note that both values in the function change the behaviour of the model since the breaking strength is a function of the weakest bonds.
- SetGaussianSpringDistribution(1.0,0.5): This function is used to set a distribution on the Young's moduli of grains. In this case the distribution is Gaussian with a mean value 1.0 and a standard deviation of 0.5 (smaller values make the distribution narrower and larger values make the distribution wider). Grains are picked randomly.
- MakeGrainBoundaries(1.0,0.5): This function gives the grain boundaries different properties, in this case 1.0 means that the elastic properties are not changed and 0.5 that the breaking strength of grain boundary springs is on average half that of intragrain springs (This does not affect the distributions that were set before).
- SetFracturePlot(50,1) simply means that we make an extra plot after 50 fractures formed. This is useful in order to view the dynamics of the very non-linear fracture process. Note however that if you call this function your *Elle* stages will be non-linear with time and do not necessarily show the real time steps anymore.

In the run function (Experiment::Run(), case 1) the following functions contain parameters that can be changed:

- if (experiment_time <5) means that the program is applying 5 steps of uniaxial compression and starts with pure shear deformation afterwards.
- DeformLattice(0.001,1): This function is used to apply uniaxial compression with a vertical strain of 0.1 percent per step. The value 1 means

that a picture is taken after the deformation was applied and the lattice relaxed.

- DeformLatticePureShear(0.001,1): This function is used to apply pure shear deformation that is conserving the area of the box. The value 0.001 means vertical strain is 0.1 percent and 1 means a picture is taken after a deformation step.

Experiment 5 - Cation exchange reactions

This section gives details for running the cation exchange reactions between garnet and biotite that are described in Chap. 3.2 and illustrated in Fig. 3.2.6-8.

Execution: In order to run these experiments, start up the *Experiment Launcher* and select the experiment from the **Experiments Menu**. The subexperiments are:

a isothermal experiment with large matrix grain size ☕
b isothermal experiment with small matrix grain size ☕
c cooling experiment ☕

In practice, a shell script (or batch file in Windows) calls the appropriate *Elle* routines and requires an *Elle* input file containing a description of the initial microstructure. To start a simulation, select the desired experiment and select **Go**. Contrary to the previous experiment, you will now not get a new window with the model, but a console window with scrolling lines.

Interface: As these are multiprocess experiments, the easiest way to monitor the progress is to view the generated *Elle* files with *showelle.* (select *showelle* from the **utilities box** on the *Experiment Laucher window*) A preferences file, defaults.zip, will be read by *showelle* so that the look up table is optimised to show Fe concentration variation in the garnet as for Fig. 3.2.8.

Processes: These experiments combine two *Elle* processes, *elle_exchange* (processes/exchange) and *elle_gbdiff* (processes/gbdiff). The exchange process simulates lattice diffusion within the grains and cation exchange between the boundary and lattice, and the second process calculates diffusion along the boundaries.

Examples: The initial file for experiment 5a has one garnet grain and one biotite grain in a matrix of large grains providing few pathways for grain boundary diffusion (see Fig. B4a). In experiment 5b, the matrix consists of smaller grains with more pathways for diffusion between the biotite and garnet (Fig. B4b). In experiments 5a and 5b, the initial Fe mole fraction is 0.8 for the garnet and 0.6 for the biotite and the bnodes are set to 0.05. The temperature is a constant 600 °C. Experiment 5c is a cooling experiment for a microstructure that includes a large garnet and several biotite grains. The initial compositions for the garnet and biotite are assumed to be 0.8 and 0.54 (in Fe mole fraction), respectively. Since these compositions are disequilibrium compositions in the experimental temperature range, garnet and biotite start to exchange cations to approach the equilibrium KD value. There is little change in zonation in the latter part of the experiment as diffusion slows with decreasing temperature (Fig. B4c).

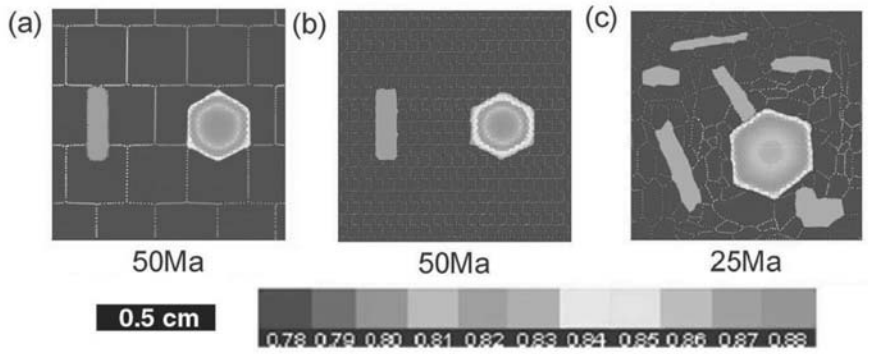


Fig. B4. **(a)** Results from an isothermal experiment with matrix grain size of 0.25cm. The biotite grain is on the left, the garnet grain on the right. **(b)** Same experiment, but with a small grain size. **(c)** Results from a cooling experiment. The garnet grain is the large hexagonal grain near the middle of the model. The CLUT (colour lookup table) of Fe molar fraction is optimized to highlight the garnet composition

Functions and Parameters for users wishing to modify parameters in the Shelle script: Details are given for experiment 5c but the parameters used and naming conventions apply to all the subexperiments. exp5c.elle is the initial elle file in which one grain (grain 84) is garnet, six grains are mica (25, 49, 54, 56, 90, 129) and the matrix is quartz. The unodes in the garnet have an Fe mole fraction of 0.8 and the unodes in the mica have an Fe mole fraction of 0.54. The bnodes have a constant initial Fe concentration of 0.05. The elle file also sets the initial temperature to 750°C.

exp5c.shelle/bat is a batch file generated using the Shelle Script Generator, shelle24.html, to set the following parameters which control the elle_experiment:

1. Experiment Name exp_5_out, the base for naming saved output files
2. *Elle* Input File exp5c.elle, the initial microstructure
3. Last Time **600**, the number of iterations for the experiment loop
4. Save Interval **100**, output files from all processes will be saved every 100 iterations

with the following processes turned on:

1. Grain Boundary Diffusion [20] Stages **500** Kappa **2.0e-9**
2. Exchange Reaction [23] Stages **500**. Kappa **30.90528**. Temperature increment **-0.5**
3. The saved output file has the process list LIST= "20 23"

The parameter, Kappa, is the pre-exponential factor used to calculate the diffusion coefficient D(T) using the Arrhenius equation D(T) = Kappa*exp(-Ea/(RT)). In this code the activation energy for diffusion (Ea) has been set to 239 kJ/mole.

All processes in the list are called once per TIMESTEP, i.e. at the end of this experiment, the exchange process will have performed a total of 30,000 stages and the temperature will have dropped from 750 to 450 °C.

Experiment 6 - Subgrain growth

This example illustrates the process of subgrain growth resulting from two different conceptual models, using a Potts Model approach. The background of the software is explained in Chap. 3.3.

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. There are two possible sub-experiments: isotropic and anisotropic:

a isotropic

b anisotropic

Interface: These files will each load their appropriate preferences file automatically. Once the file is loaded, select **Run** from the **Run menu** of the *Elle* window to watch the system evolve. You can either display any of the three EULER orientations of the unodes.

Examples: In the first example (Fig. B5) a simple subgrain growth experiment has been performed by taking an input file with a highly strained grain of NaCl derived from EBSD measurements, which exhibit high grain lattice distortions. We let the substructure of the grains evolve while the grain boundaries remain stable. The driving force is the reduction of energy where the energy below the critical misorientation (between adjacent subgrains) of 15° is isotropic.

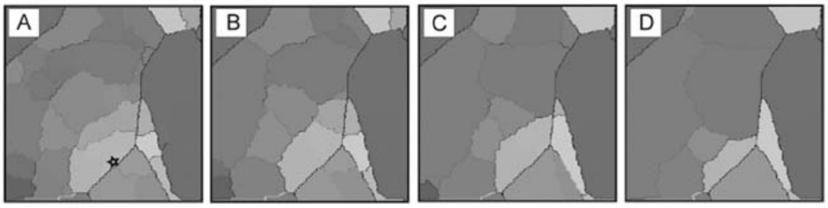


Fig. B5. Example run showing isotropic surface energy driven subgrain growth. Greyscale scheme shows relative misorientation from one crystallographic orientation (marked as black star) in greyscale; black lines signify grain boundaries with >15° misorientation; **(a-d)** results at model time steps 1500, 2500, 4000, 6000 (modified after Piazolo et al., 2004)

Example Elle Run: Subgrain growth – anisotropic In this simulation (Fig. B6) we take into account the anisotropy of surface energy and mobility of subgrain boundaries. The input microstructure is the same as for section 3.3.4, however the calculation of the energy state differs as now the energy between data points below the critical misorientation of 10° is taken to be anisotropic (see above for details). It can be seen that in this case more subgrains remain at the end of the simulation, as the anisotropy has the effect of slowing down the microstructure evolution.

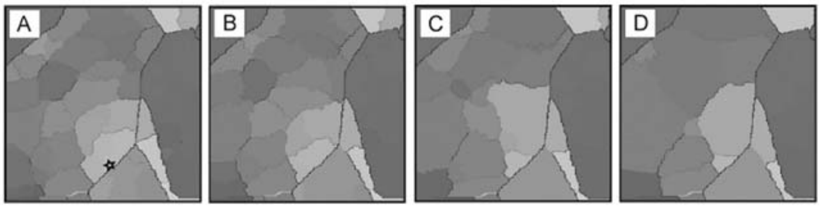


Fig. B6. Example run showing anisotropic surface energy driven subgrain growth. Greyscale scheme shows relative misorientation from one crystallographic orientation (marked as black star) in Greyscale; black lines signify grain boundaries with >10° misorientation; **(a-d)** results at model time steps 0, 20, 40, 60

Functions and parameters for users wishing to modify runtime parameters: This is a process simulates subgrain growth using a Potts model. It only used the unodes in the grains themselves, no migration of flynn_boundaries is taken into account. The user can vary 2 parameters:

1. userdata[0] = max_angle gives the angle at which the maximum energy is reached (e.g. 15° for salt)
2. userdata[1] = slope. A value of slope=1 signifies that all energies are the same, while another value allows differences in energy according to misorientation. Slope=2 signifies that the energy changes linearly, as a fraction of the max_angle. This means that at a misorientation of 10 and max_angle is 15, the energy is 10/15=0.66667. The energy reaches unity when misorient is equal or larger than max-angle. Slope=3 signifies a Shockley equation change between 0 and max_angle misorientation, with:

   $$\text{energy} = \left(\frac{\text{orient}}{\text{max\_angle}}\right)\left(1 - \ln\left(\frac{\text{orient}}{\text{max\_angle}}\right)\right)$$

   If the value for slope is not 1, 2, 3 it is set by default to 1. For example: elle_sub_gg -u 15 3 means that the max_angle is 15 (here the energy is 1), and the Shockley equation is used to calculate the energy.

Experiment 7 - Grain growth

The grain growth example shows how the grain size in a grain aggregate increases by grain boundary migration that is solely driven by grain boundary surface energy (see Section 3.5). It also illustrates how a non-equilibrium microstructure evolves towards a foam texture.

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. There are two possible sub-experiments, the first simply shows the current position of the grain boundaries, while the second tracks past positions so that the parts of grains previously swept by grain boundaries can be seen:

a without boundary tracking (Fig. B7)

b with boundary tracking (Fig. B8)

Interface: These files will each load their appropriate preferences file automatically. Once the file is loaded, select **Run** from the **Run menu** of the *Elle* window to watch the system evolve.

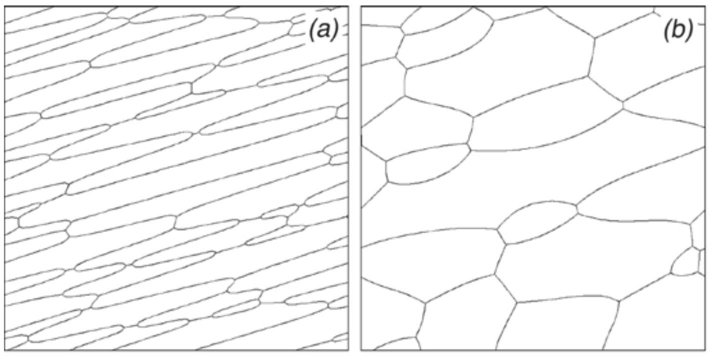


Fig. B7. Example of a grain growth experiment. **(a)** Originally foliated aggregate at t=0, and **(b)** at t=40000 time steps

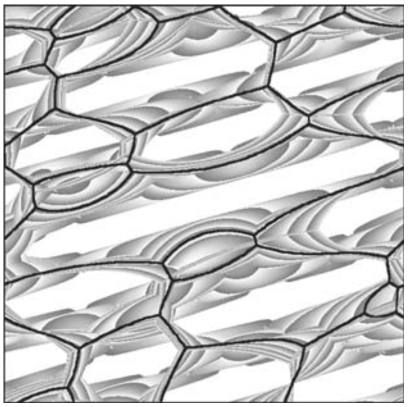


Fig. B8. The successive positions of all grain boundaries at each time step during this time interval are shown in shades of grey that vary cyclically with time. White areas have never been swept by grain boundaries

Examples: In this example (Fig. B7) we will perform a simple grain growth experiment by taking an input file with a microstructure far from equilibrium in terms of grain shapes and evolve the grain boundaries by defining a boundary energy term, which provides the driving force for grain boundary migration. The example can also be seen in Chap. 3.5 (Fig. 3.5.4). Figure B8 is the same experiment, but the past positions of grain boundaries are displayed. This way one can see which parts of the grains have been swept by grain boundaries at least once (Jessell et al., 2003).

Experiment 8 - Evolution of a partial melt

These experiments show how melt pockets evolve for different melt-crystal boundary energies. The example experiment has a solid-solid to solid-liquid surface energy ratio such that the wetting angle is 10°, 60° or 120° at equilibrium. The melt fraction is fixed at 2%. The starting microstructure already shows some disequilibrium features, but most melt pockets have wetting angles of around 10°. During the start of the simulation these melt pockets quickly adjust to an equilibrium shape depending on the wetting angle. This behaviour was predicted by Laporte et al. (1997) for <60° wetting angles in a static system when grains have similar sizes.

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. There are three possible sub-experiments, with varying wetting angles for the same starting microstructure:

a 10° wetting angle 🚶
b 60° wetting angle 🚶
c 120° wetting angle 🚶

Interface: These files will each load their appropriate preferences file automatically. Once the file is loaded, select **Run** from the **Run menu** of the *Elle* window to watch the system evolve. Red areas are melt, blue are crystal. Sometimes two melt pockets may merge, and a boundary will be seen between these pockets, however this does not have any physical significance in our calculations. The image is updated only every 10 timesteps. Note that the simulations take rather a long time.

Example: The example provided here (**c**) has a solid-solid to solid-liquid surface energy ratio such that the wetting angle is 120° at equilibrium (Fig. B9). The melt fraction is fixed at 2%. The starting microstructure is far from equilibrium for a 10° wetting angle. However, during the start of the simulation these melt pockets quickly adjust to an equilibrium shape (convex triangles). This behaviour was predicted by Laporte et al. (1997) for <60° wetting angles in a static system when grains have similar sizes.

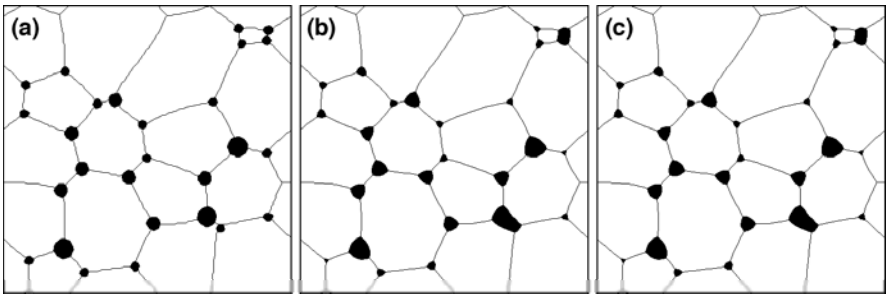


Fig. B9. Three stages from experiment **c**, where melt is black. **(a)** The starting melt pocket shape is strongly out of equilibrium for a 120° wetting angle. **(b)** after 1000 and **(c)** after 5000 steps of boundary movement. Notice the rapid change in melt pocket shape in the first 1000 steps and no visible difference in the next 4000 steps

Experiment 9 - Rigid porphyroblast growing in a deforming matrix

This section describes how to run the *Basil* program in conjunction with the *Elle* software using the specific examples shown in Figs. 3.8.1, 3.8.2 and 3.8.4. We model the development of grain growth during simple shear deformation and a relatively hard porphyroblast rotating within a relatively weak mantle deformed to high strains under simple shear deformation.

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments Menu**. The three experiments in this section are:

- **a** Rigid grain in a deforming matrix
- **b** Grain growing in a deforming matrix with viscosity contrast
- **c** High strain experiment with soft mantle

Experiments **b** and **c** will take several hours to run and you can either use *showelle* (see Appendix C) or *Sybil* (see Appendix F) to follow the evolution of the sample. Only *Sybil* can be used to view experiment **a**. This experiment should only take a few minutes and *Sybil* will open automatically if found on your system.

In practice, a shell script (or batch file in Windows) is run, which calls *Basil* and requires an input file containing the parameters needed to run *Basil*, including boundary conditions. For examples **b** and **c**, the script also calls the appropriate *Elle* routines and requires an *Elle* input file containing a description of the initial microstructure (grains, subgrains, etc.).

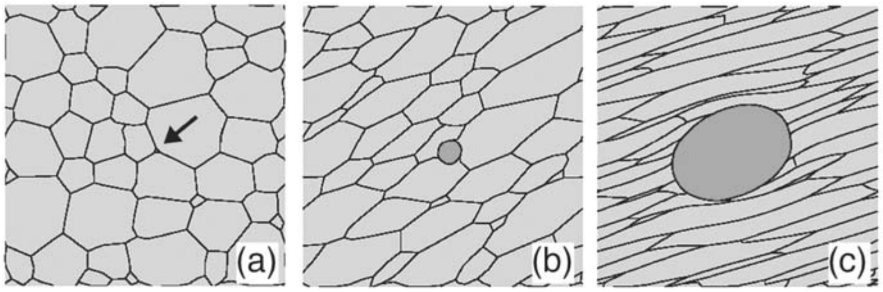


Fig. B10. Experiment 9b in which a central growing grain, with viscosity 5 times that of the matrix, resists deformation. **(a)** Initial microstructure. **(b)** After a strain of 1 and 750 growth steps. **(c)** After a strain of 3 and 2250 growth steps

Examples: In example **a**, we use *Basil* to calculate the velocity around a rigid, circular object in a soft matrix that is subject to simple shear.

In the example shown in Fig. B10, we use *Elle* and *Basil* to study the growth of a porphyroblast in a crystalline matrix that is subject to simple shear. The 2D square domain of unit dimensionless length shows a crystalline matrix with irregular grain size surrounding a small grain of harder material (viscosity coefficient 5, representing garnet). We assume Newtonian constant viscosity (stress exponent n=1) in both materials.

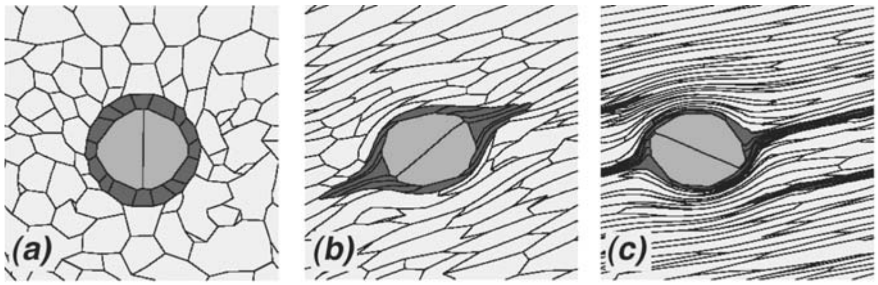


Fig. B11. Experiment 9c. Rotation of a porphyroblast (medium grey: viscosity $50\eta_o$) surrounded by a weak layer of crystals (dark grey: viscosity $0.5\eta_o$) embedded in a matrix (light grey: viscosity η_o) undergoing simple shear. The total finite strain is: **(a)** 0, **(b)** 2, and **(c)** 5. Boundary conditions are as described for Experiment 9b but bnode data is mapped onto a regular *Basil* mesh

In the final example, we show a dynamic shear experiment in which a constant volume porphyroblast that is embedded in a matrix that undergoes a large simple shear strain (Fig. B11). In this case the viscosity of the porphyroblast is 50 times that of the background matrix. A layer of grains whose viscosities are half that of the matrix surrounds the porphyroblast and the mechanical boundary conditions are the same as those for Fig. B11 but there is no grain growth.

Functions and Parameters for users wishing to modify parameters in the Shelle script: The *Elle* shell script file, exp9b.shelle/bat, is generated using the web page shelle24.htm*l,* to set the following parameters that control the *Elle* experiment:

1. Experiment Name exp9b_out, the base for naming saved output files
2. *Elle* Input File exp9b.elle, the *Elle* input file containing the initial microstructure
3. Last Time **150**, the number of time steps for the experiment loop
4. Save Interval **50**, output files from all processes will be saved every 50 time steps with the following processes turned on:
5. *Basil* [1 2 3]
6. *Basil* Input File exp9b.in
7. Reposition [13]
8. Expand [19], Expand Speed **1**, Expand Growth Stages **15**, Expand Max Area **0**
9. The saved output file has the process list: LIST= "**1 2 3 13 19**"

The first three processes in the list are *elle2poly* (1), *Basil* (2), *basil2elle* (3) and are an immutable group included when *Basil* is turned on. Reposition (13) puts the bnodes back into a unit cell and is normally called after any deformation process (e.g. *Basil*, *OOF*, *Manuel*). The last process in the list is *elle_expand* (19) (performed by the routine in the code processes/pblast/expand.elle.cc) and results in growth of any grains with a non-zero EXPAND attribute set in the *Elle* input file. Expand Area is turned off but allows the growth rate to decline exponentially as the grain expands. All processes in the list are called once per time step (the size of which is set by *Basil*), but the Expand process requires a finer discretisation in time than that required by *Basil*, in order to limit the distance a bnode can move in one stage, and thereby avoid instabilities and topological problems due to overstepping. The Expand process is therefore implemented using a number (15 in this example) of sequential Growth Stages in each *Basil* time step (0.02 in the example, as described below). The rate at which a grain boundary moves in the Expand process is the default value of 10^{-4} distance unit per growth stage (0.075 dimensionless velocity units in this example). This rate can be modified by a multiplicative factor set in the Expand Speed line (1 in this example).

Elle input file parameters: The *Elle* input file, exp9b.elle, contains the initial microstructure which includes a grain (grain number 25) with the EXPAND attribute set and a viscosity constant 5 times that of the matrix.

Basil input file parameters: In the *Basil* input file, exp9b.in, the following parameters can be changed to vary the time step and simple shear boundary conditions:

- STEPSIZE IDT0=50
- STOP KEXIT=500 TEXIT=0.02
- SAVE KSAVE=50 TSAVE=0.02

TEXIT is the dimensionless time level for the *Basil* run. It may be varied but keep TSAVE=TEXIT for the *Elle* experiment so that the *Basil* solutions will contain two records, t=0 and t=TEXIT. IDT0 controls the internal time step for the *Basil* calculations and should be set so that 1/IDT0 <= TSAVE, e.g. if TEXIT and TSAVE are set to 0.01 then set IDT0 to 100.

To change the strain rate for this problem, keep the *y*-component of the velocity, UY, set to zero and vary the *x*-component of the velocity, UX, for the top (Y=YMAX) and bottom (Y=YMIN) boundaries:

```
ON Y = YMIN : UX = -0.5
ON Y = YMIN : UY = 0.0
ON Y = YMAX : UX = 0.5
ON Y = YMAX : UY = 0.0
```

Displaying output: The output of experiment **a** can be viewed using *Sybil*. To reproduce Fig. B10(a), open a terminal or command window, change to the experiment directory, run *Sybil* and open exp9a.log from the File menu.

The output from **b** and **c** can be viewed using the auxiliary program, *showelle.* The graphics preferences will be loaded from the defaults.zip in the corresponding directory when an *Elle* file is opened.

Experiment 10 - Lattice rotations

This section gives details for running the lattice rotation during deformation experiment that is described in section 4.3.

Execution: In order to run one of these experiments (🚶), start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**.

As this is a multi-process experiment, it will start automatically and you can either use *showelle* to view the microstructural evolution (see Appendix C) or *Sybil* (see appendix F) to follow the stress/strain evolution of the sample.

In practice, a shell script (or batch file in Windows) is run, which calls the appropriate *Elle* and *Basil* routines, and requires an *Elle* input file containing a description of the initial microstructure (grains, subgrains, etc.) and a *Basil* input file containing the parameters needed to run *Basil*, including boundary conditions.

Interface: As these are multi-process experiments, they will start automatically and you can either use *showelle* to view the microstructural evolution (see Appendix C) or *Sybil* (see appendix F) to follow the stress/strain evolution of the sample.

To reproduce Fig. B12a-b, use *showelle* to open any *Elle* file generated by this experiment. The Postscript format orientation plots for each *Elle* file may be saved by selecting **Save→Orientation Plots** from the **Graphics menu**.

Example: In this example (Fig. B12) we couple a TBH calculation with the *Basil* FEM code to simulate deformation of a quartz polycrystal, and base the updated viscosities on the work term calculated by the TBH code. In this way grains which are able to deform by glide on slip systems with low Critical Resolved Shear Stress will deform more rapidly than grains which only have 'hard' slip systems available.

Processes: This experiment combines three processes:

1. *Basil*- non-linear viscous flow
2. *ell_tbh*- Taylor-Bishop-Hill lattice reorientation calculations
3. *elle_viscosity*- resetting of each grains viscosity value

***Functions and parameters for users wishing to modify parameters in the Shelle Script**:* exp10.elle is the initial elle file that consists of a simple foam texture with 121 grains. The initial viscosities of all grains is set to 1. exp10.shelle/ba*t* is a batch file generated using Shelle24.html to set the following parameters which control the *Elle* experiment:

1. Experiment Name **tbh**, the base for naming saved output files

2. *Elle* Input File exp10.elle, the initial microstructure
3. Last Time **30**, the number of iterations for the experiment loop
4. Save Interval **1**, output files from all processes will be saved every iteration with the following processes [and their process id] turned on:
5. **Basil** [7 1 2 3 13] The file **exp10.in** controls the *Basil* code, in this series of experiments you can change the stress exponent by altering the value SE on line 10 (this is initially set to 1). The initial time step includes a randomisation [7] of all crystallographic orientations.
6. **elle_tbh** [4] Calculation of lattice rotations and work term
7. **elle_viscosity** [16] ViscosityMode 0 (viscosity is based on work term calculated by TBH code)
8. The saved output file has the process list LIST= "LIST="7 1 2 3 13 4 16""

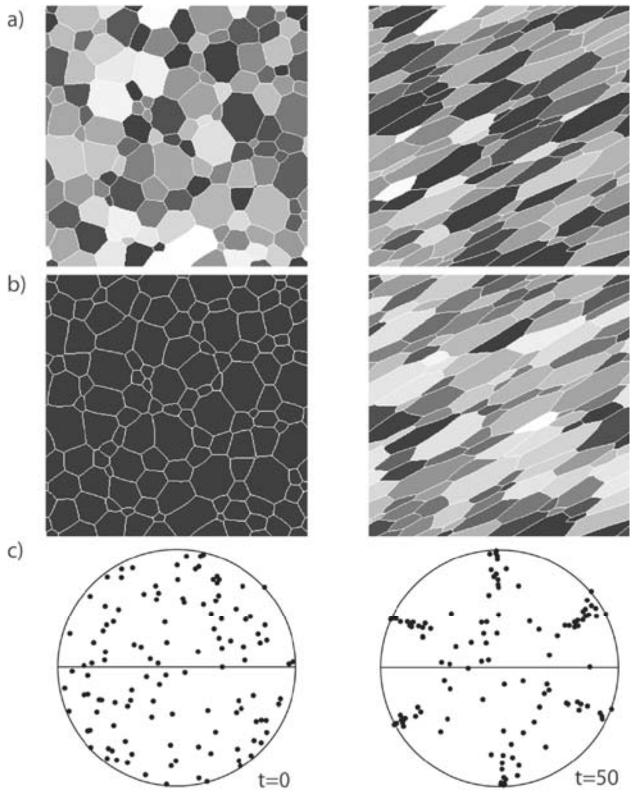


Fig. B12. Simple coupled TBH-FEM calculation. **(a)** Lattice orientations at *t*=0 and *t*=50, grey scale a function of only the Alpha Euler angle. Notice that by *t*=50 many of the grains share a similar orientation. **(b)** Instantaneous viscosities at *t*=0 and *t*=50 (bright=high viscosity) **(c)** c-axes stereograms at *t*=0 and *t*=50, the latter equivalent to a shear strain of 1.5

Functions and parameters for users wishing to modify parameters in the Shelle Script: exp10.elle is the initial elle file that consists of a simple foam texture with 121 grains. The initial viscosities of all grains is set to 1. exp10.shelle/ba*t* is a batch file generated using Shelle24.html to set the following parameters which control the *Elle* experiment:

9. Experiment Name **tbh**, the base for naming saved output files
10. *Elle* Input File exp10.elle, the initial microstructure
11. Last Time **30**, the number of iterations for the experiment loop
12. Save Interval **1**, output files from all processes will be saved every iteration with the following processes [and their process id] turned on:
13. **Basil** [7 1 2 3 13] The file **exp10.in** controls the *Basil* code, in this series of experiments you can change the stress exponent by altering the value SE on line 10 (this is initially set to 1). The initial time step includes a randomisation [7] of all crystallographic orientations.
14. **elle_tbh** [4] Calculation of lattice rotations and work term
15. **elle_viscosity** [16] ViscosityMode 0 (viscosity is based on work term calculated by TBH code)
16. The saved output file has the process list LIST= "LIST="7 1 2 3 13 4 16""

All processes in the list are called once per time step.

quartz.crss This file defines the relative Critical Resolved Shear Stress values for different slip systems. In the example used here they were defined as:

1	5.00	5	∞	9	9.50
2	11.00	6	9.50	10	15.00
3	6.00	7	9.50	11	15.00
4	∞	8	9.50		

where the slip system id numbers, which are defined in quartz.xl are:

1 BASAL A	6 (2-1-1 1) C+A2	10 RHOMB A
2 PRISM C	7 (2-1-1 1) C+A3	11 RHOMB -A
3 PRISM A	8 (2-1-1 1) -C-A2	12 RHOMB C+A
4 PRISM C+A	9 (2-1-1 1) -C-A3	13 RHOMB -C-A
5 PRISM -C-A		

Experiment 11 - Boudinage

This example illustrates the development of boudinage in a granular aggregate during compaction and pure shear deformation. The background of the software is explained in Chap. 3.11 and is based on a Lattice-Spring model (Chap. 2.8).

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. On most computers you should use the lower-resolution files since the algorithm is relatively slow. Possible sub-experiments, which simply vary the density of unodes are:

a 2850 particles
b 11500 particles
c 46000 particles

Once the file is loaded, select **Run** from the **Run menu** of the *Elle* window to watch the system evolve.

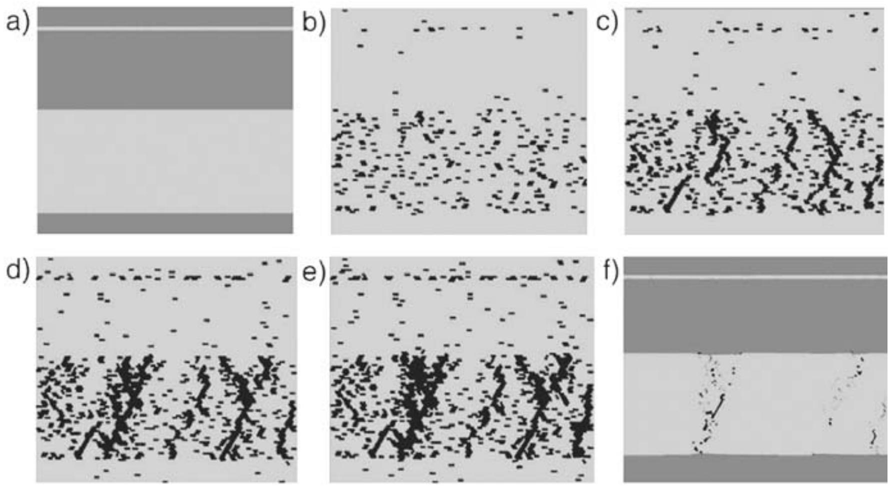


Fig. B13. Fracture development in two competent layers (Young's modulus is 10 times that of the matrix). **(a)** Initial setup plotting the Young's moduli. **(b-e)** fracture pattern after **(b)** 10 steps, **(c)** 20 steps, **(d)** 30 steps and **(e)** 40 steps. **(f)** Final fractures or larger damage zones in the thicker layer (plotting Young's moduli)

Interface: These files will each load their appropriate preferences file automatically, and only the unodes will be visible. You can now choose to view the Young's moduli of different grains, the fractures and the differential and mean stress. In order to view fractures, choose values between -1.0 and 0.0, unodes with fractured bonds will be blue and unfractured unodes red. Differential and mean stress should be scaled during a run, normal values are: differential stress 0.0 to 0.01 and mean stress - 0.01 to 0.01 (negative is compression).

Examples Figure B13a and B13f show the initial and final geometry with Young's modulus as parameter. Figure B13b-e show the fracture patterns during 40 stages (fractured particles are dark, set fractures to -1 and 0.0). It can be seen that lattice geometry does have an influence on the fracture patterns. The thick layer develops a fracture spacing where two fracture clusters dominate and start to open.
Functions and parameters for users wishing to use the Experiment Interface to modify parameters The parameters used in the example are:

- SetGaussianStrengthDistribution(2.0,0.8),
- WeakenAll(0.1,1.0,1.0,
- WeakenHorizontalParticleLayer(0.9,0.92,10.0,1.0,1.0),
- WeakenHorizontalParticleLayer(0.2,0.6,10.0,1.0,1.0),
- DeformLatticePureShear(0.001,1).

In the initialization function (Experiment::Init(), case 2) the following functions contain parameters that can be changed:

- SetGaussianStrengthDistribution(2.0,0.8) This function is used to set a Gaussian distribution on breaking strengths of all springs. Mean of the distribution is 2.0 and the standard deviation 0.8. The breaking strength of the material is dependent on the lowest breaking strengths of springs so that a variation of the standard deviation will affect the breaking strength of the whole material.
- WeakenAll(0.1,1.0,1.0) This function changes the Young's modulus, viscosity or breaking strength of all particles. In this case the material made softer.
- WeakenHorizontalParticleLayer(0.2,0.6,10.0,1.0, 1.0) This function inserts a horizontal layer of particles that may have different Young's modulus, viscosity and breaking strength than the surrounding matrix. All particles between $x = 0.2$ and $x = 0.6$ are part of the layer. In this example the layer has a Young's modulus that is 10 times that of the surrounding matrix.

- SetFracturePlot(50,1) This function just specifies that a plot is made after 50 bonds are broken. In that case *Elle* stages may not necessarily represent time steps.

In the run function (Experiment::Run(), case 2) the following function contain parameters that can be changed:

- DeformLatticePureShear(0.001,1) Deform the lattice by pure shear deformation with a vertical strain of 0.001.

Experiment 12 - Dissolution grooves

This example illustrates the development of dissolution grooves at a stressed crystal-solute interface. The background of the software is explained in Chap. 3.12 and is based on a Lattice-Spring model (Chap. 2.8).

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. On most computers you should use a medium-resolution file since the algorithm is relatively fast. Possible sub-experiments, which simply vary the density of unodes are:

a	2850 particles ☕
b	11500 particles 🚶
c	46000 particles 🚶

Once the file is loaded, select **Run** from the **Run menu** of the *Elle* window to watch the system evolve.

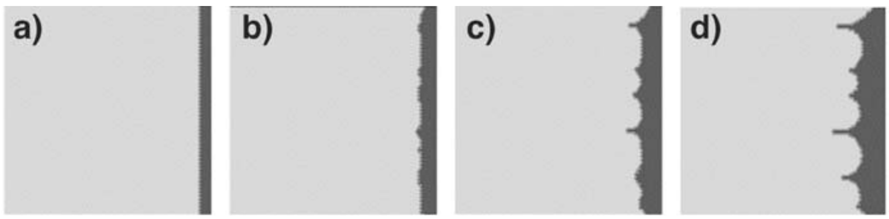


Fig. B14. Development of grooves on a crystal surface after **(a)** 2, **(b)** 15, **(c)** 30 and **(d)** 49 stages. Fluid is black and solid grey. The initial roughness that develops due to the heterogeneous dissolution develops into cusp instabilities with a well-defined wavelength

Interface: These files will each load their appropriate preferences file automatically, and only the unodes will be visible. You can now choose to view the different phases (from –1 to 1; where the fluid is –1, the interface 0 and the solid 1) or the stress. Differential and mean stress should be scaled during a run, normal values are: differential stress 0.0 to 0.01 and mean stress - 0.01 to 0.01 (negative is compression).

Examples Figure B14 shows the development of grooves on a crystal surface. Input file is res200.elle. Parameters used are the ones given above. The solid in Fig. B14 is grey and the fluid black. The figure shows the roughening of the surface (Fig. B14b) and development of cusps after 30 stages (Fig. B14c), which have developed a well-defined wavelength after 49 stages (Fig. B14d).

Functions and parameters for users wishing to use the Experiment Interface to modify parameters: In the initialization function (Experiment::Init(), case 6) the following functions contain parameters that can be changed:

- SetPhase(0.0,0.0,500.0,0.8*)* This function makes the lattice unbreakable since fractures are not wanted in the dissolution processes (at least in this example).
- SetGaussianRateDistribution(2.0,0.3) Set a Gaussian distribution on rate constants in order to induce the roughening. Mean is 2.0 and standard deviation 0.3.
- WeakenAll(8.0,1.0,1.0) Multiply spring constants by 8.
- Set_Mineral_Parameters(1) Mineral is quartz (set molecular volume and surface free energy)
- Set_Absolute_Box_Size(0.0001) x dimension of the simulation box is 0.0001 m.
- Set_Time(6000.0,4) time of one deformation step is 6000 years.
- DissolveXRow(0.95,1.1) Dissolve particles between $x = 0.95$ and $x = 1.1$.
- In the run function (Experiment::Run(), case 6) the following functions contain parameters that can be changed:
- DeformLattice(0.002,1) Deform the lattice by uniaxial vertical compression with confined walls on the sides and a vertical strain of 0.002.
- Dissolution_Strain(20) Dissolve as a function of elastic and surface energies and plot a picture after 20 particles have dissolved.

Experiment 13 - Stylolites

This example illustrates the development of stylolites at a stressed crystal-solute interface. The background of the software is explained in Chap. 3.12 and is based on a Lattice-Spring model (Chap. 2.8).

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. On most computers you should use the low-resolution files since the algorithm is relatively slow. Possible sub-experiments, which simply vary the density of unodes are:

a 2850 particles ☕
b 11500 particles 🚶
c 46000 particles 🚶

Once the file is loaded, select **Run** from the **Run menu** of the *Elle* window to watch the system evolve.

Interface: These files will each load their appropriate preferences file automatically, and only the unodes will be visible. You should now choose to view the phases, where –1 is the fluid, 0 the interface and 1 the solid.

Examples Figure B15 illustrates an example of stylolite development in a lattice that has a horizontal resolution of 100 particles. Parameters in the functions are the same as above. The initially flat surface roughens during progressive dissolution due to the heterogeneities in dissolution constants. The absolute size of the box is large (0.1 m horizontally) so that surface energy is rather low and elastic effects dominate. Therefore some relatively steep structures develop that are comparable with the characteristic stylolite "teeth".

Figure B16 illustrates roughening of a stylolite with the same properties as the stylolite shown in Fig. B15 except for the dimension of the box, which is only 1 mm horizontally. Therefore surface energy effects are more pronounced so that the structures appear more rounded.

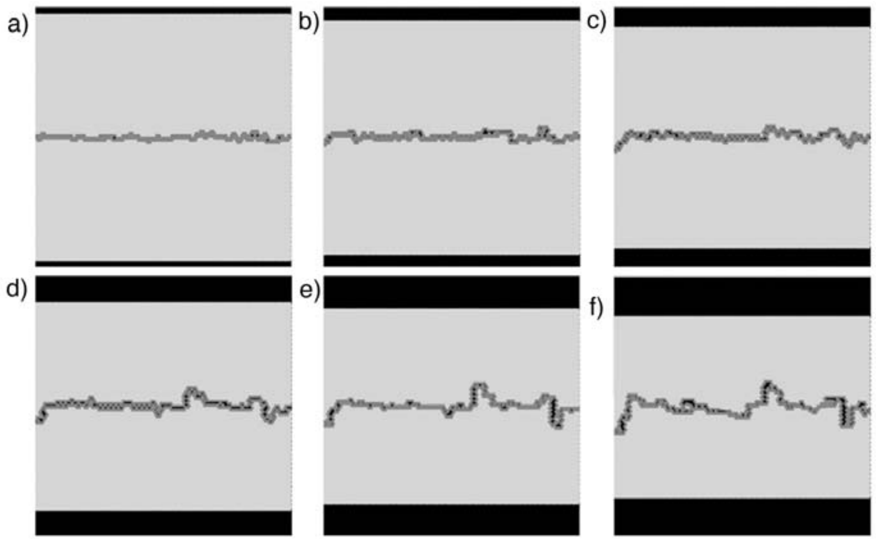


Fig. B15. Development of the roughness of a stylolite. Stages are **(a)** 500, **(b)** 1000, **(c)** 1500, **(d)** 2000, **(e)** 2500 and **(f)** 3000. The horizontal dimension of the box is 0.1 m

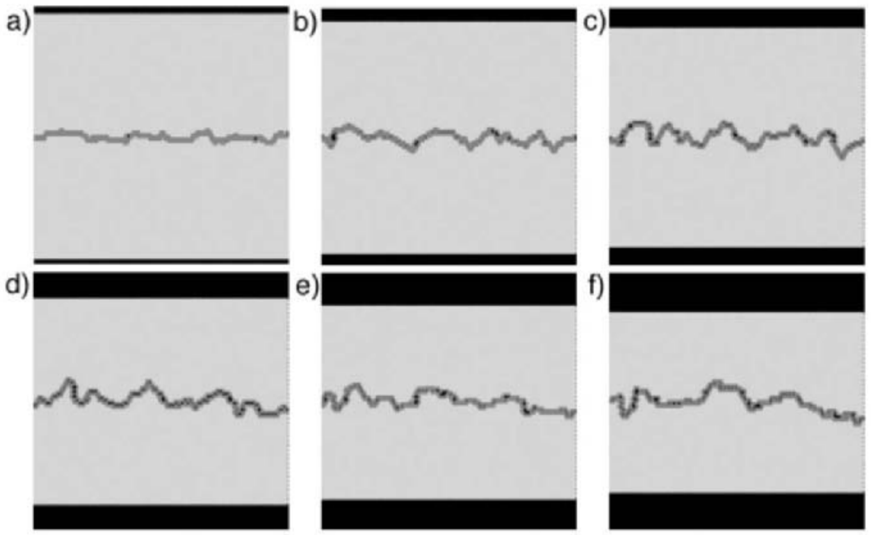


Fig. B16. Development of the roughness of a stylolite where the horizontal dimension (1 mm) of the box is two orders of magnitude smaller than that of Fig. 13.1. Stages are **(a)** 500, **(b)** 1000, **(c)** 1500, **(d)** 2000, **(e)** 2500 and **(f)** 3000

Functions and parameters for users wishing to use the Experiment Interface to modify parameters: In the initialization function (Experiment::Init(), case 7) the following functions contain parameters that can be changed:

- SetPhase(0.0,0.0,500.0,0.8) As in the groove case this turns fracturing off.
- SetGaussianRateDistribution(2.0,0.5) Set a Gaussian distribution on dissolution constants, 2.0 is mean of the distribution and 0.5 the standard deviation.
- Set_Rate_Two_Phase(0.05,0.6,1.0) Set a bimodal distribution on the dissolution constants of particles. 5% of all particles dissolve a factor 0.6 slower than the rest.
- WeakenAll(4.0,1.0,1.0) Multiply the spring constants by 4.
- Set_Mineral_Parameters(1) Mineral is quartz (set molecular volume and surface free energy)
- Set_Absolute_Box_Size(0.1) *x* dimension of box is 0.1 m.
- Set_Time(40.0,4*)* One time step represents 40 years.
- DissolveYRow(0.49,0.5, true) Dissolve a row of particles in the centre between *y* = 0.49 and *y* = 0.5.True means that the dissolved particles are removed from the picture.

In the run function (Experiment::Run(), case 7) the following functions contain parameters that can be changed:

- DeformLatticeNewAverage2side(0.00005,1) Deform the lattice by pushing upper and lower crystal together.
- Dissolution_StylosII(100000,0,0,0,1) Dissolve as a function of stress and elastic and surface energy.

Experiment 14 - Strain-rate partitioning during porphyroblasts growth

This section explores the evolving strain-rate partitioning behaviour within a layered succession during porphyroblast growth, and explore the feedback between strain-rate partitioning around effectively rigid porphyroblasts and metamorphic reactions, as described in section 4.8.

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**.

Possible sub-experiments are:

- **a** porphyroblast growth, no deformation
- **b** deformation, but no porphyroblast growth
- **c** porphyroblast growth, with deformation

As these are multi-process experiments, they will start automatically and you can either use *showelle* to view the microstructural evolution (see Appendix C) or *Sybil* (see appendix F) to follow the stress/strain evolution of the sample.

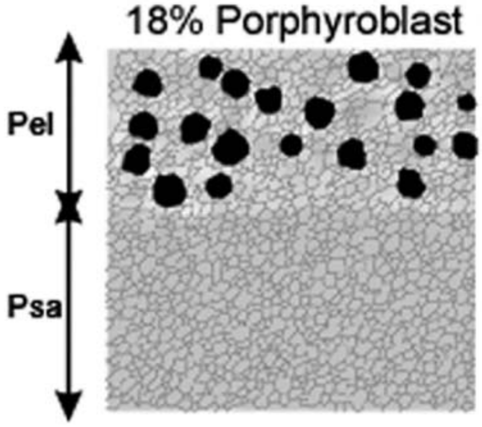


Fig. B17. Porphyroblast growth without deformation (Experiment a)

Examples: These examples compare porphyroblast with and without deformation, and deformation with and without porphyroblasts. The first experiment simply shows the growth from small nuclei of a set of porphyroblasts (Fig. B17). In the following two experiments (Fig. B18), two geometries were deformed in simple shear to bulk strains of approximately 1.5 to illustrate the difference between strain partitioning in a porphyroblast-free layered system and a porphyroblastic layered system. In the absence of porphyroblasts, the pelitic layer in our model has a viscosity 0.4 times that of the psammitic layer. Strain is partitioned in this model such that the pelitic layer records a shear strain of approximately 2.1 and the psammitic layer records a shear strain of approximately 0.8, consistent with the viscosity contrast between the two layers. A second geometry, with approximately 18% porphyroblast in the pelitic layer, was deformed to a similar bulk strain.

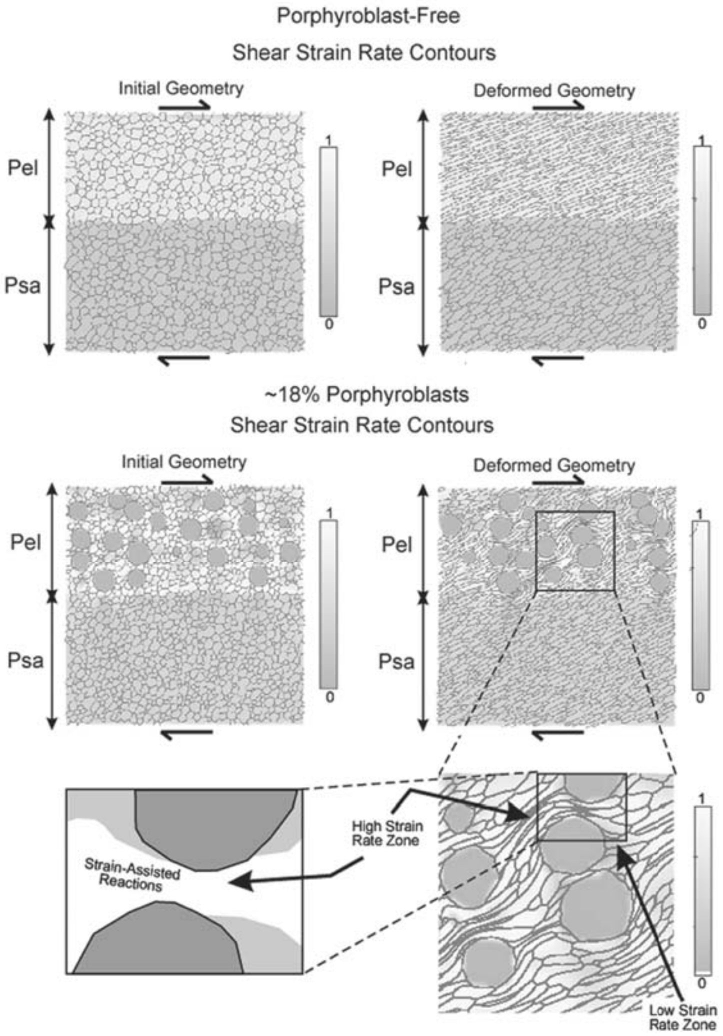


Fig. B18. Top: Shear strain rate contour maps for the porphyroblast-free simple shear deformation experiment b. Note the homogeneous shear strain rate in each layer and the bulk strain rate partitioning into the pelitic layer. Middle: Shear strain rate contour maps for the 18% porphyroblast simple shear deformation experiment c. Note strain rate partitioning within the pelitic layer with high strain rate zones around the porphyroblasts. Also, the strain rate is higher in the psammitic layer than in the porphyroblast-free experiment. Bottom: Close-up images of the shear strain rate distribution around porphyroblasts showing high and low shear strain rate zones. The image on the left has had the grain topology removed for simplification. The location of strain-assisted reaction between two porphyroblasts is indicated

Interface: Strain-rate maps and viscosity profiles are created using *Sybil*, and the evolving topology can be viewed using *showelle*.

To reproduce strain-rate maps, open any solution file generated by the shelle script in *Sybil*. In *Sybil*, select **contour → strain → msst**. To reproduce the figures used in Chap. 4.8, set the options to "min = 0, max = 2. " To produce viscosity profiles through the model, open any solution in *Sybil*. In *Sybil*, select **profile → 2-D → stress → visc**. Set the lower limit to 0 and the upper limit to 1 (in the Y-direction) to generate a profile through both layers in the model. To generate a profile through just the upper pelite layer, set the limits to min=0.5, max=1.0. Set the minimum scale value to 0 and the maximum value to 2 (note the scale is $\log_{10}$ of the viscosity).

Processes: This is a multi-stage experiment using the following processes:

1. ***Basil*:** The models in Chap. 4.8 use linear viscous flow
2. ***elle_gg***: curvature-driven grain boundary migration
3. ***elle_pblast***: selected grains grow as porphyroblasts. To reproduce the syn-deformation growth experiments in Chap. 4.8, the following parameters are used in the shelle script: Lasttime=030, Timestep=001, GGstages=003 (Process [18]), PblastStages=015 (Process [8]), SaveInterval=1, List= "1 2 3 8 18 13".

Modifying the Shelle Script: A single deformation step is modelled using a .shelle script with the following parameters:

- Startfile = the name of the porphyroblast growth stage you want to deform,
- Time = 0,
- LastTime = 1,
- GGStages = 0,
- Saveinterval = 1,
- SaveAll = 13,
- Basinfile = exp14b.in,
- Processes: "1 2 3 13"

Experiment 15 - Anisotropic grain growth

This section gives details for running the anisotropic grain growth experiments that are described in section 4.1. Be warned that these experiments each take several hours to run; overnight experiments often prove to be the least disruptive.

Processes: This experiment use only the *elle_gbm* process, in its earlier version, see Chap. 3.5 for details.

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. Possible sub-experiments, which simply vary the amount of anisotropy are:

a No boundary anisotropy
b 10% Anisotropy
c 20% Anisotropy
d 30% Anisotropy
e 40% Anisotropy
f 50% Anisotropy

Once the file is loaded, select **Run** from the **Run menu** of the *Elle* window to watch the system evolve.

Interface: As this is a single process experiment, the easiest way to monitor the progress is to view the generated *Elle* files directly in *elle_gbm*. In addition, every 1000 time steps an *Elle* file will be saved.

Example: An initially randomly oriented aggregate consisting of 315 grains (and hence an average grain area of $3.17 \cdot 10^{-3}$ (the entire model has unit area) with a foam texture, was allowed to grow for between 100,000 and 300,000 time steps (this may take several hours). Experiment **a**, which has an isotropic surface energy distribution, produces a foam texture with triple junction angles of about 120°. At a first glance, the microstructure in the other experiments looks like a foam texture, but most triple junction angles are in fact far from 120 degrees (Fig. B19).

The rate of grain growth in all experiments with a surface energy anisotropy is vastly decreased relative to an identical experiment where surface energies were assumed to be isotropic (A), and experiments with a high boundary energy anisotropy quickly reach an apparently stagnant grain size. After a considerable time in the experiments with higher surface energy anisotropies, a second phase of grain growth commences, dominated by the exaggerated growth of just one or two grains (Fig. B20).

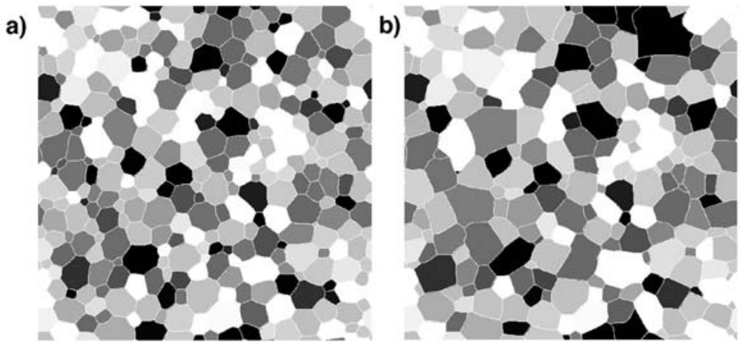


Fig. B19. Experiment F. **(a)** Time step 0, shading reflect lattice orientation. **(b)** Time step 100,000. Most triple junction angles reflect surface energy anisotropy by being far from 120 degrees

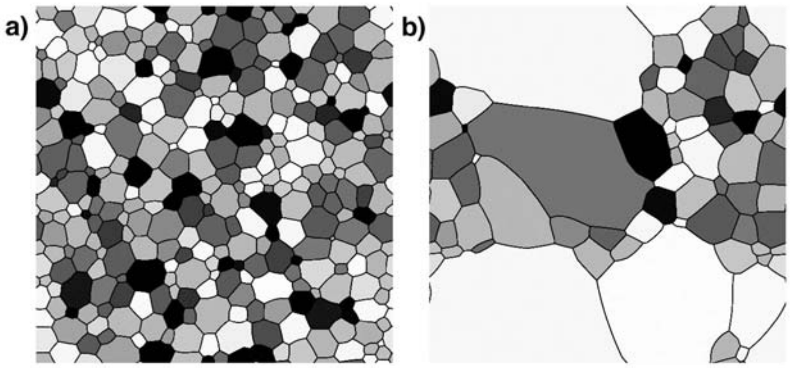


Fig. B20. Experiment C. **(a)** Time step 0, shading reflects lattice orientation **(b)** Time step 200,000

Experiment 16 - Dynamic recrystallisation

In the model presented here, we can simulate dynamic recrystallisation during plane strain simple shear deformation of a polycrystal, involving crystal lattice rotation, formation of subgrains, recrystallisation by nucleation, recrystallisation by subgrain rotation (also called rotational recrystallisation), grain boundary migration and recovery (see Section 4.2 for more details, but the complete list of parameters used and reasoning behind those the reader is referred to Piazolo (2001) and Piazolo et al. (2002). Three coarse-grained starting microstructure files and three fine-grained files are provided, each at three different grain boundary mobilities ($1{\cdot}10^{-12}$ $20{\cdot}10^{-12}$ and $40{\cdot}10^{-12}$).

***Execution*:** In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. Possible sub-experiments are:

a fine grained with low mobility
b fine grained with medium mobility
c fine grained with high mobility
d coarse grained with low mobility
e coarse grained with medium mobility
f coarse grained with high mobility

As these are multi-process experiments, they will start automatically and you can either use *showelle* to view the microstructural evolution (see Appendix C) or *Sybil* (see appendix F) to follow the stress/strain evolution of the sample.

Interface: In practice, a shell script (or batch file in Windows) is run, which calls the appropriate *Elle* and *Basil* routines, and requires an *Elle* input file containing a description of the initial microstructure (grains, subgrains, etc.) and a *Basil* input file containing the parameters needed to run *Basil*, including boundary conditions.

Examples: To investigate the effect of the relative rates of the different microscale processes active during dynamic recrystallisation experiments were performed in which one parameter was varied. The grain boundary mobility (*GBMob*) was chosen to be $1{\cdot}10^{-12}$, $20{\cdot}10^{-12}$ or $40{\cdot}10^{-12}$ $m^2s^{-1}J^{-1}$. The interested reader is referred to Piazolo (2001) and Piazolo et al. (2002) for further simulations the energy threshold value for recrystallisation by nucleation was also varied. Two different starting microstructures were used: one fine-grained and one-coarse grained aggregate Figure B21 shows the development of the microstructure in simple shear at a shear srain of 0.25 and 2, showing both the grains as well as their dislocations densities. A variety of image sequences of developing microstructures during progressive deformation are provided at

www.uni-mainz.de/FB/Geo/Geologie/tecto/elle_movies._

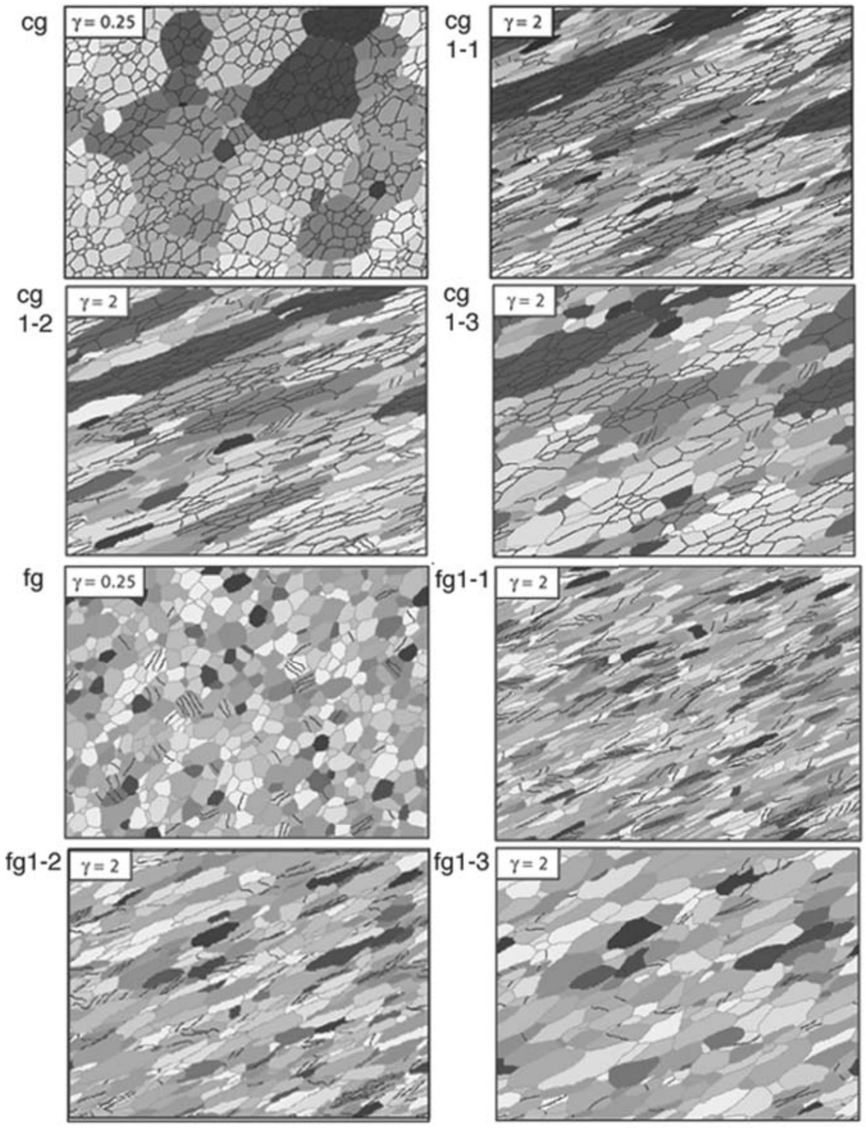


Fig. B21. Results of the 6 simulation runs. Different grey colours signify different grains, dark boundaries low angle boundaries and light grey coloured boundaries high angle boundaries. For the naming of simulation runs refer to Table 4.2.1 (modified from Piazolo 2001)

Processes: This experiment combines several processes:

1. ***Basil***- non-linear viscous flow
2. ***elle_split***- creation of new subgrain boundaries
3. ***elle_gbm***- curvature and defect energy driven grain boundary migration
4. ***elle_tbh***- calculates lattice rotations
5. ***elle_nucl_xx***- nucleation of new grains
6. ***elle_angle_rx***- nucleation of new grains by rotation recrystallisation
7. ***elle_recovery***- reduction in defect energies within grains
8. ***elle_viscosity***- resetting of each grains viscosity value

Experiment 17 - Deformation localisation

This section gives details for running the deformation localisation experiment which is described in section 3.9. As this experiment depends on probabilistic calculations, no two experiments will run the same, and its long-term evolution may be quite different from that shown in the figures.

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. The three experiments in this section are:

a Stress Exponent of 1, no Grain Size Modifying Processes ☕
b Stress Exponent of 3, no Grain Size Modifying Processes ☕
c Stress Exponent of 1, with Grain Size Modifying Processes ☕
d Stress Exponent of 3, with Grain Size Modifying Processes 🚶

As these are multi-process experiments, they will start automatically and you can either use *showelle* to view the microstructural evolution (see Appendix C) or *Sybil* (see appendix F) to follow the stress/strain evolution of the sample.

In practice, a shell script (or batch file in Windows) is run, which calls the appropriate *Elle* and *Basil* routines, and requires an *Elle* input file containing a description of the initial microstructure (grains, subgrains, etc.) and a *Basil* input file containing the parameters needed to run *Basil*, including boundary conditions.

Examples: We simulate a simple system in which three processes are active. We stress that these are not supposed to accurately represent the detailed behaviour of any single deformation process, but rather act as proxies for three classes of processes, namely a grain-size and strain dependant rheology, a process that increases the grain size, and a process that reduces the grain size (see Jessell et al. (2005) for a complete set of the experimental parameters).

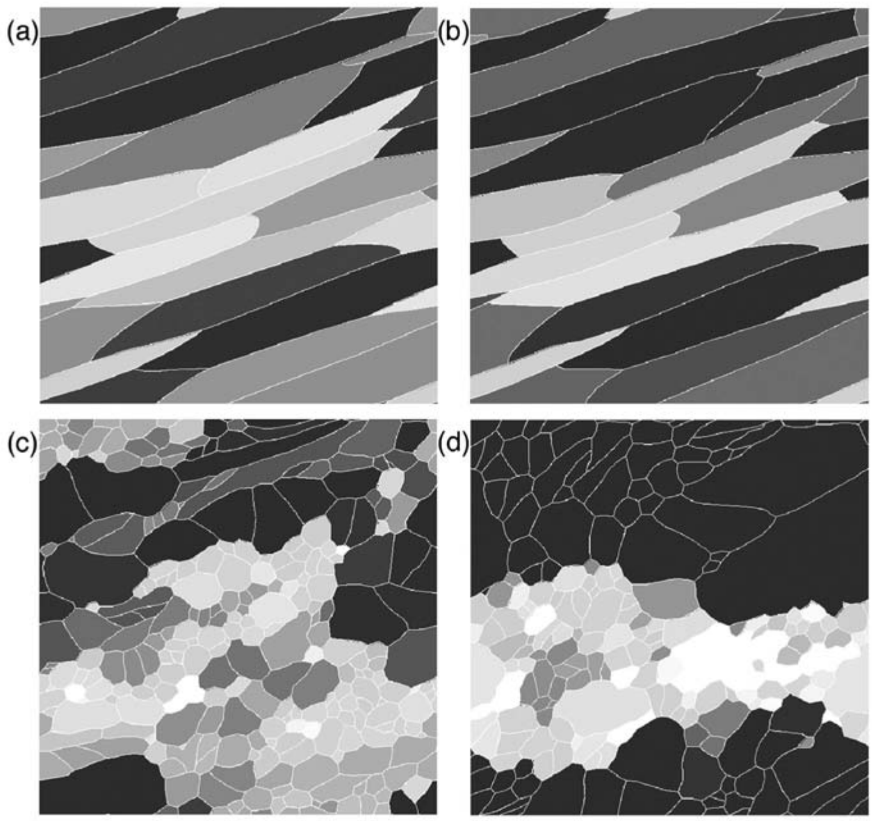


Fig. B22. Maps of the average finite first principal strain for each grain during time step 50 for each of the four experiments. The shading in each map is proportional to accumulated strain, overlaid by the grain boundary network as white lines. The same look up table is used for all four maps. **(a)** Experiment A, showing a relatively homogeneous strain distribution, with a concentration of deformation in the small grain in the top-right. **(b)** Experiment B, showing a relatively homogeneous strain distribution, but with a concentration of deformation in the small grain in the top-right. **(c)** Experiment C, showing a broad zone of higher strain. **(d)** Experiment D, showing a broad sub-horizontal zone of high strain. Notice that the high strain rate zones in Fig. 4.3.2d are not all high finite strain zones in this figure, and that the zone of bulk finite strain is much more clearly defined than the equivalent instantaneous high strain rate zone

A single experiment consists of defining a starting microstructure, then cycling this microstructure through the three processes, so that the microstructure is slightly modified by each process in turn. In this sequence of experiments we chose to change only two parameters. The first of the two parameters is the stress exponent of the grain size dependent flow law, which could have values of 1 or 3. The second variable is the activation or non-activation of the Grain Size Reduction and Grain Size Increase processes. We thus define four Experiments A, B, C and D (Table 4.3.1).

These four simple experiments demonstrate a range of behaviours (Fig. B22), a monotonic evolution for Experiments A and B versus a more complex path for Experiments C & D. As one might expect, in these experiments, the more non-linear the system, either in terms of the rheology or the process coupling, the more complex the behaviour.

Processes: This experiment combines four processes:

1. ***Basil***- non-linear viscous flow
2. ***elle_split***- creation of new high angle grain boundaries
3. ***elle_gg***- curvature driven grain boundary migration and
4. ***elle_viscosity***- resetting of each grains viscosity value

Functions and parameters for users wishing to modify parameters in the Shelle Script: exp17.elle, the initial elle file that consists of a simple foam texture with 16 grains. The initial viscosity of all grains is set to 1. exp17.shelle/bat is a batch file generated using Shelle24.html to set the following parameters which control the Elle_experiment:

1. Experiment Name exp_17a, the base for naming saved output files
2. *Elle* Input File loca.elle, the initial microstructure
3. Last Time **100**, the number of iterations for the experiment loop
4. Save Interval **5**, output files from all processes will be saved every fifth iteration with the following processes [and their process id] turned on:
5. ***Basil*** [1 2 3 13] The file exp17.in controls the Basil code, in this series of experiments you can change the stress exponent by altering the value SE on line 10 (this is initially set to 3 in experiments (b) and (d)).
6. ***elle_split*** [5] SplitMode 1 (split probability is based on bulk strain of each grain)
7. ***elle_gg*** [18] GGStages 010
8. ***elle_viscosity*** [16] ViscosityMode 1 (viscosity is based on grain area and bulk strain of each grain)

9. The saved output file has the process list LIST= "1 2 3 13 5 18 16" to modify this experiment to run without grain size modifying processes, change this to LIST= "1 2 3 13 16".

All processes in the list are called once per time step.

Experiment 18 - Expanding inclusions

This example illustrates single grains expanding in a brittle matrix. The background of the software is explained in Chap. 4.4.3 and is based on a Lattice-Spring model (Chap. 2.8).

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. On most computers you should use the medium-resolution files since the algorithm is relatively slow. Possible sub-experiments, which simply vary the density of unodes are:

a 2850 particles
b 11500 particles
c 46000 particles

Once the file is loaded, select **Run** from the **Run menu** of the *Elle* window to watch the system evolve.

Interface: These files will each load their appropriate preferences file automatically, and only the unodes will be visible. You can now choose to view the Young's moduli of different grains, the fractures and the differential and mean stress. In order to view fractures choose values between -1.0 and 0.0, unodes with fractured bonds will be blue and unfractured unodes red. Differential and mean stress should be scaled during a run, normal values are: differential stress 0.0 to 0.01 and mean stress - 0.01 to 0.01 (negative is compression).

Example Figure B23 shows the fracture pattern around the inclusion after 15 and 30 steps, as well as the mean and differential stress. Values for mean stress range from -0.03 to 0.005 and values for differential stress range from 0 to 0.03. Figure (d) shows the fracture pattern after 30 steps and figure (e) and (f) the corresponding mean and differential stress using the same values.

The example shows the fracture and stress pattern around an expanding grain or inclusion. Input file is res100.elle.

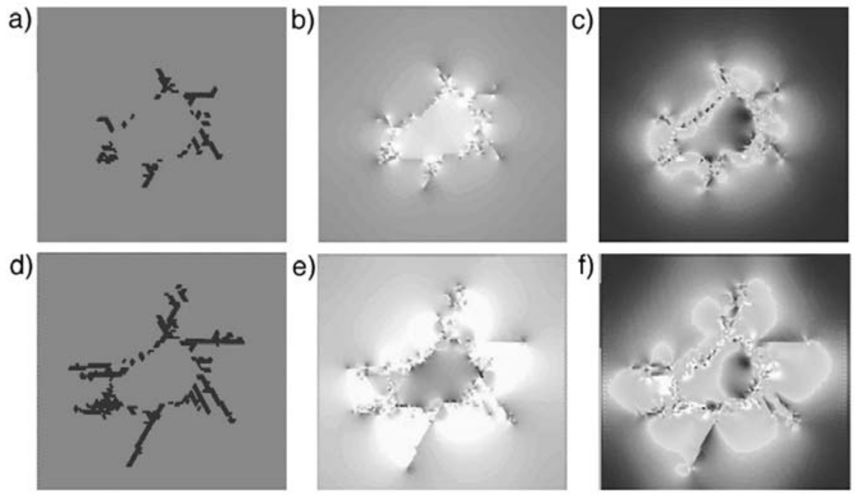


Fig. B23. Expanding inclusions, associated fracture and stress patterns. **(a)** Fracture pattern after 15 steps. **(b)** Mean stress after 15 steps, ranging from -0.03 (bright?) to 0.005 (dark). **(c)** Differential stress after 15 steps, ranging from 0 (dark) to 0.03 (bright). **(d)** Fracture pattern after 30 steps, fractured particles are dark. **(e)** Mean stress after 30 steps and **(f)** differential stress after 30 steps. Mean stress values are from –0.008 to 0.0001 and differential stress values from 0.0 to 0.02

Functions and parameters for users wishing to use the Experiment Interface to modify parameters In the initialization function (Experiment::Init(), case 3) the following functions contain parameters that can be changed:

- SetPhase(0.0,0.0,0.8,1.2) Set a linear distribution on breaking strength of single bonds. Mean is 0.8 times the default value. Distribution will be from default value ± 0.6 times default value.
- SetFracturePlot(20,1) Plot a picture after 20 particles are fractured.

In the run function (Experiment::Run(), case 3) the following functions contain parameters that can be changed:

- ShrinkGrain(5,-0.002,1) Shrink grain number 5 by the amount -0.002 times area of grain and plot a picture.

Experiment 19 - Mud cracks

This example illustrates the development of shrinkage cracks in a brittle medium. The background of the software is explained in Chap. 4.4.3 and is based on a Lattice-Spring model (Chap. 2.8).

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. As input file you should use a rather low-resolution file unless you have a very fast computer (**a** & **b** work fine, **c** is ok, **d** becomes problematic and **e** takes a long time even on a fast computer), **c** gives good results in reasonable time (several hours). Possible sub-experiments, which simply vary the density of unodes are :

- **a** 2850 particles ☕
- **b** 11500 particles 🚶
- **c** 46000 particles 🚶

Once the file is loaded, select **Run** from the **Run menu** of the *Elle* window to watch the system evolve.

Interface: These files will each load their appropriate preferences file automatically, and only the unodes will be visible. You can now choose to view the Young's moduli of different grains, the fractures and the differential and mean stress. In order to view fractures, choose values between -1.0 and 0.0, unodes with fractured bonds will be blue and unfractured unodes red. Differential and mean stress should be scaled during a run, normal values are: differential stress 0.0 to 0.01 and mean stress - 0.01 to 0.01 (negative is compression).

Examples: Several stages in the development of a mud crack are shown in Fig. B24. After nucleation, the crack propagates through the whole model (Fig. B24a-c), when a regular spacing develops. This spacing subsequently decreases (Fig. B24d-f). The typically hexagonal fracture pattern develops that is well known from mud-cracks or columnar basalts.

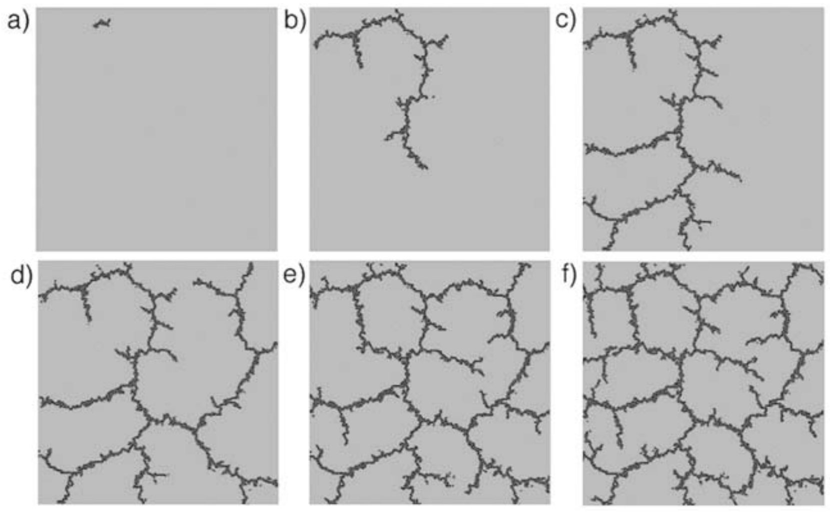


Fig. B24. Mud cracks after stages **(a)** 10, **(b)** 50, **(c)** 100, **(d)** 150, **(e)** 200 and **(f)** 240. Cracks develop a regular spacing with a hexagonal symmetry

The example in Fig. B 19.1 shows the progressive development of fractures in a shrinking layer (for example mud-cracks). The input file is res200.elle.

Functions and parameters for users wishing to use the Experiment Interface to modify parameters In the initialization function (Experiment::Init(), case 4) the following functions contain parameters that can be changed:

- WeakenAll(2.0,1,1.0) Change spring constants of all springs (multiply by 2)
- SetPhase(0.0,0.0,1.0,1.0) Set a linear distribution on breaking strength of single bonds.
- SetFracturePlot(20,1) Make a plot after 20 bonds are fractured

In the run function (Experiment::Run(), case 4) the following functions contain parameters that can be changed:

- ShrinkBox(0.001,1,1) Shrink all particles in the box by the amount 0.001 times area.

Experiment 20 - Visco-elastic deformation and fractures

This example illustrates the development of structures in visco-elastic materials. The background of the software is explained in Chap. 4.6 and is based on a Lattice-Spring model (Chap. 2.8).

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. As input file you should use a rather low resolution file unless you have a very fast computer (**a** & **b** work fine, **c** is ok, **d** becomes problematic and **e** takes a long time even on a fast computer), **c** gives good results in reasonable time (several hours). Two sets of possible sub-experiments, which simply vary the density of unodes are provided for the case of a single brittle grain in a viscous matrix, and for a brittle layer in a viscous matrix:

a brittle grain 2850 particles
b brittle grain 11500 particles
c brittle grain 46000 particles

d brittle layer 2850 particles
e brittle layer 11500 particles
f brittle layer 46000 particles

Once the file is loaded, select **Run** from the **Run menu** of the *Elle* window to watch the system evolve. Visco-elastic deformation takes longer to calculate than purely elastic deformation!

***Interface*:** These files will each load their appropriate preferences file automatically, and only the unodes will be visible. You can now choose to view the Young's moduli of different grains, the fractures and the differential and mean stress. In order to view fractures, choose values between -1.0 and 0.0, unodes with fractured bonds will be blue and unfractured unodes red. Differential and mean stress should be scaled during a run, normal values are: differential stress 0.0 to 0.01 and mean stress - 0.01 to 0.01 (negative is compression).

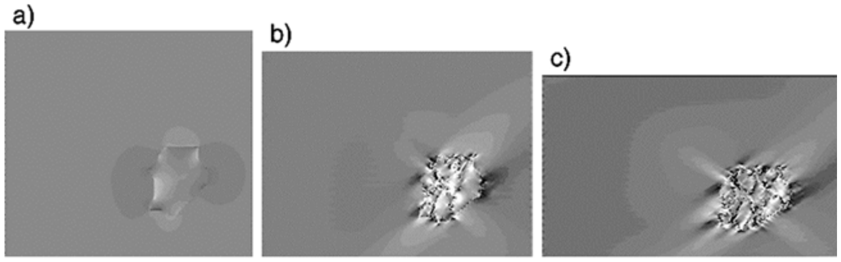


Fig. B25. Viscoelastic deformation of a hard grain in a more viscous matrix. Plot shows the mean stress (values -0.04 to 0.01) where light colour is high compressive stress (negative in this case) and dark colour low compressive stress. The hard grain is deforming by internal fractures that are opening whereas the viscous matrix flows around it without fracturing

Examples: The first example of visco-elastic deformation and fracturing is illustrated in Fig. B25 where a hard grain is embedded in a weaker matrix. The input file is res100.elle.

During the simulation the hard grain fractures whereas the matrix flows around the hard grain without fracturing. Plotted is the mean stress where light colour is large compressive stress and dark colour low compressive stress. Fractures in the hard grain open and may be filled in nature with vein material.

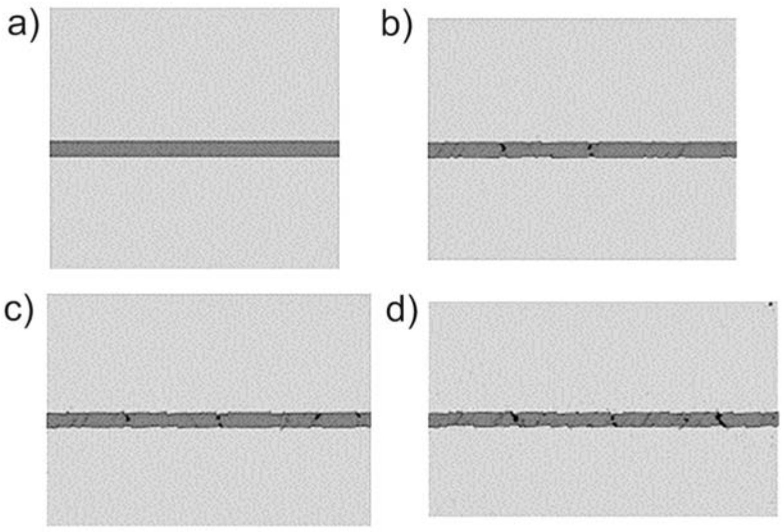


Fig. B26. Viscoelastic deformation leading to the formation of rotating and folding boudins. Steps are **(a)** 20, **(b)** 80, **(c)** 120 and **(d)** 180

Figure B26 shows the progressive deformation of the matrix and the layer. The more competent layer fractures and folds during extension. Small flanking folds develop and the boudins are slightly rotating and necking. In the end some of the larger fractures are opening between boudins.

Functions and parameters for users wishing to use the Experiment Interface to modify parameters The parameters for the brittle grain example are:

- SetWallBoundaries(1,1.0),
- SetPhase(0.0,0.0,0.2,1.2),
- WeakenAll(0.8,1.1,1.0),
- WeakenGrain(7,10,10,1),
- DeformLatticePureShear(0.001,1),
- ViscousRelax(1,1e11).

The parameters for the brittle layer example are:

- SetWallBoundaries(1,1.0),
- SetPhase(0.0,0.0,1.0,0.6),
- WeakenAll(0.1,1.1,1.0),
- WeakenHorizontalParticleLayer(0.47,0.52,5.0,5.0,1),
- DeformLatticePureShear(0.001,1),
- ViscousRelax(1,1e11).

In the initialization function (Experiment::Init(), case 5) the following functions contain parameters that can be changed:

- SetWallBoundaries(1,1.0) Set external boundary walls that repel particles. 1 means leave the particle walls and 1.0 is the repulsion constant of the walls.
- SetPhase(0.0,0.0,0.2,1.2) Set a linear distribution on breaking strength of bonds. Mean multiplied by 0.2 and 1.2 is the size of the distribution.
- WeakenAll(0.8,1.1,1.0) Change elastic constants of all particles (multiply by 0.8)
- WeakenGrain(7,10,10,1) Change elastic constant, viscosity and breaking strength of grain number 7. First number behind the grain number is the elastic constant (times 10), second number the viscosity (times 10) and last number the breaking strength (no change, times 1).
- WeakenHorizontalParticleLayer(0.47,0.52,20.0, 20.0,1.0) Define a horizontal layer between $y = 0.47$ and y is 0.52 with an elastic constant that is 20 times higher than the matrix, a viscosity that is 20 times higher and a remaining breaking strength (times 1.0).

- SetFracturePlot(50,1) Make a plot after 50 fractures developed.

In the run function (Experiment::Run(), case 5) the following functions contain parameters that can be changed:

- DeformLatticePureShear(0.001,1) Deform the lattice by pure shear deformation with a vertical strain component of 0.001 and plot a picture (1 is plot).
- ViscousRelax(1,1e11) Use the viscous relaxation routine, 1 is plot a picture and 1e11 the viscous time for one deformation step (in seconds). With the above strain this leads to a strain rate of 10^{-12}.

Experiment 21 - Strain localisation and rigid object kinematics

This section gives details for running one of the deformation experiments involving porphyroblasts with deformable mica caps described in section 4.7.2.

Execution: In order to run one of these experiments, start up the *Experiment Launcher* and select the appropriate experiment from the **Experiments menu**. Possible sub-experiments, which simply vary the relative viscosity of the mica caps are:

a	viscosity ratio 1 ➡
b	viscosity ratio 5 ➡
c	viscosity ratio 25 ☕

As these are multi-process experiments, they will start automatically and you can either use *showelle* to view the microstructural evolution (see Appendix C) or *Sybil* (see appendix F) to follow the stress/strain evolution of the sample.

Interface: The easiest way to monitor the progress is to view the newly generated *Elle* files with *showelle*. Alternatively, the velocity, mechanical quantities and file geometry can all be viewed in *Sybil*. To view only the grain geometry in Fig. 4.7.3, use showelle to open any *Elle* file generated by this experiment.

To reproduce the velocity, vorticity or deviatoric shear stress plots in Fig. 4.7.3, use *Sybil* to open any solution file generated by this experiment, then select **Contour** and the appropriate quantity from the **Graphics menu**, and click the box to use the maximum and minimum range of data values. To overlay the grain geometry on your plot, choose **Elle** from the **XYPlot menu**, and select the *Elle* file that corresponds to the solution to have plotted. To view the Lagrangian mesh for the experiments, go to **XYPlot**, the **LGMesh** and choose **elements**.

Examples: The physical model is shown in Fig. 4.7.2. A central, hard inclusion is intended to simulate a garnet porphyroblast. This grain is surrounded above and below by "caps" that are intended to simulate material that is relatively weak in shear, such as micas with or without thin fluid layers. The surrounding polygonal framework is intended to simulate a background matrix with a homogenous, isotropic viscosity structure. The matrix and central inclusion are assigned constant viscosities, whereas the relative viscosity of the caps was varied from one experiment to another (Fig. B27).

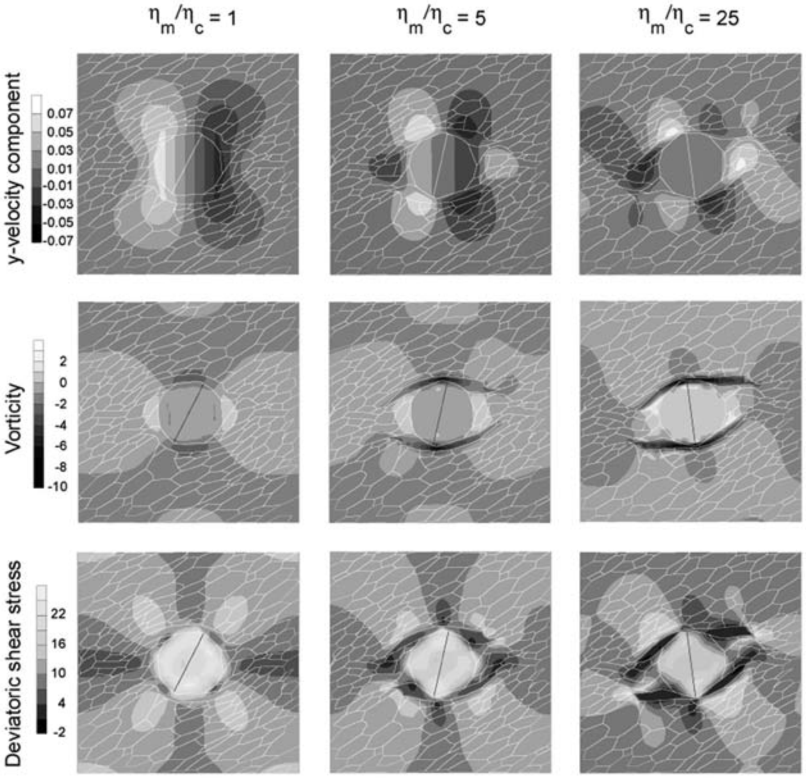


Fig. B27. Contour plots of the *y*-velocity component, vorticity and deviatoric shear stress for the three experiments with different viscosity caps. Total shear strain in each is 1.0. The deformed microstructure from Fig. 4.7.2 has been overlain on the contour plots

Processes: This experiment uses only *Basil*- a Finite Element formulation for linear or non-linear viscous flow.

Functions and parameters for users wishing to modify parameters in the Shelle Script: exp21.elle is the initial elle file that consists of a simple foam texture with a semi-circular grain in the centre. The default viscosities of all matrix grains is set to 10, the viscosity of the larger central grain is set to 1000, and the viscosity of the lubricating caps is set to 10, 2 and 0.4 in experiments (a), (b) and (c) respectively.
exp21.shelle/bat is a batch file generated using Shelle24.html to set the following parameters which control the Elle_experiment:

1. Experiment Name **exp21_out**, the base for naming saved output files
2. *Elle* Input File exp21.elle, the initial microstructure
3. Last Time **30**, the number of iterations for the experiment loop
4. Save Interval **1**, output files from all processes will be saved every iteration with the following processes [and their process id] turned on:
5. **Basil** [1 2 3 13] The file exp21.in controls the *Basil* code, and in this series of experiments you can change the stress exponent by altering the value SE on line 10. The experiments in section 4.7.2 were run with SE=1, or a linear viscous rheology.
6. The saved output file has the process list LIST= "1 2 3 13". All processes in the list are called once per time step.

Appendix C The *Elle* graphical user interface

Jens Becker

C.1 Introduction

A graphical user interface has been developed to simplify the usage of the *Elle* platform. In the following, a short introduction to the usage of this graphical user interface (GUI) is given. However, users vaguely familiar with computers in general will have no problems to use it since it follows standard, menu-driven applications. Upon program start-up (either a process or a utility), the main application window displays a menu bar at the top and a status bar at the bottom of the main application window. In the main bar, the name of the process currently used is displayed.

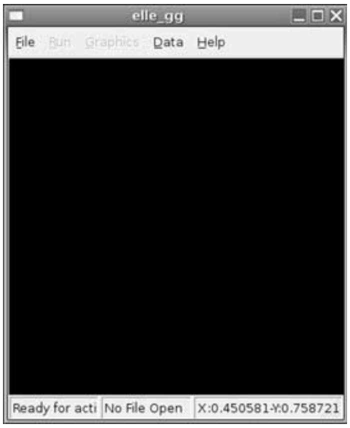


Fig. C1. Main window at program start-up (no file is loaded). The **Run** and **Graphics menus** are disabled as long as no input file is loaded

Start-up procedure: When an *Elle* utility or process starts up or a new *Elle* file is loaded from the GUI, you may pass additional information. If you are in a command line system such as Linux or Msys, you can use the userdata to transfer extra control parameters (see Appendix D). If however a zip file with the same name as the *Elle* file or defaults.zip is found in the same directory as the *Elle* file, this file is loaded, the userdata information is taken from userdata.txt within that zip file. This zip file also includes three other files that control: the graphical appearance of the microstructure (dsettings.txt); the colour look up table (cmap.txt) and the run options (runopts.txt), which are also loaded at the same time. Whenever an *Elle* file is saved using the **Save menu** item from the GUI, a zip file containing all four of these files is also created.

C.2 The status bar

The status bar is divided into three fields (Fig. C1). The first field has two functions: during an experiment it displays the stage numbers or indicates whether there was an error. While there is no experiment running, it displays various status messages. The second field displays the name of the file currently loaded (or the message "No File Open" if no file is loaded). The third field displays the coordinates of the mouse pointer (in *Elle* units, which range from 0 to 1, the origin being in the lower left corner).

C.3 The menu-bar

The menu bar has 5 entries, **File**, **Run**, **Graphics**, **Data** and **Help**. Upon program start-up and during an experiment, the **Run** and **Graphics** menus are disabled to prevent the user from making potentially harmful changes of the simulation while it is running. In the **File** menu, you can choose to load and save *Elle* files as well as quit the program. Of particular importance is the entry "Show Log". This opens another window where logging output of the *Elle* simulations package is displayed, where error messages will be stored.

C.3.1 The File Menu

1. **Open:** Open an *Elle* file
2. **Save:** Save an *Elle* file and a preferences zip file containing all the current graphical and runtime options
3. **Show Log:** Open a window showing he debugging information
4. **Exit:** Quit the *Elle* process

C.3.2 The Run Menu

1. **Run:** Run an *Elle* process or update the graphics if a new file has been loaded in *showelle.* The process runs the specified number of time steps each time **Run** is selected, using the just modified microstructure as the starting point.
2. **Rerun:** Restarts an *Elle* process using the initial microstructure defined in the *Elle* file.
3. **Run Options:** Opens the Run Options Dialogue

C.3.3 The Graphics Menu

1. **Preferences:** Opens the Graphics Preferences Dialogue
2. **Zoom:** Opens the Zooming Window
3. **Always Overlay:** Displays each new image on top of the previous ones, without clearing screen first
4. **Save:** Opens a submenu where you can choose between saving the currently displayed *Elle* file as a bitmap, saving all *Elle* files in a certain directory as bitmaps, or saving a stereoplot of the three principal axes of each grain (if defined)
5. **Redraw:** Simply redraws the current *Elle* file

C.3.4 The Data Menu

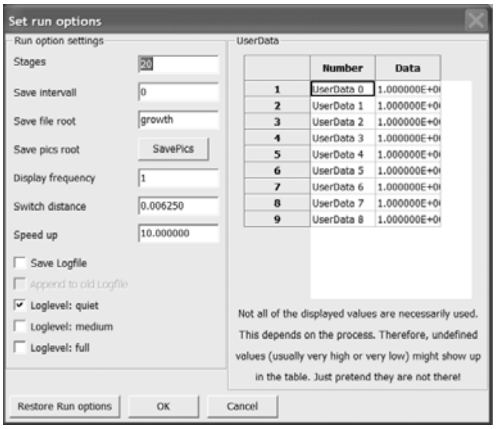
1. **Flynns:** Open the flynn Data Window
2. **Bnodes:** Open the bnodes Data Window
3. **Unodes:** Open the unodes Data Window

C.3.5 The Help Menu

1. **About:** Acknowledgements
2. **Help:** Not yet implemented

C.4 The Run Options dialog

The Run Options Dialog is used to control important settings of the simulations.



Stages 20
Save intervall 0
Save file root growth
Display frequency 1
Switch distance 0.006250
Speed up 10.000000

- [ ] Save Logfile
- [ ] Append to old Logfile
- [x] Loglevel: quiet
- [ ] Loglevel: medium
- [ ] Loglevel: full

	Number	Data
1	UserData 0	1.000000E+0
2	UserData 1	1.000000E+0
3	UserData 2	1.000000E+0
4	UserData 3	1.000000E+0
5	UserData 4	1.000000E+0
6	UserData 5	1.000000E+0
7	UserData 6	1.000000E+0
8	UserData 7	1.000000E+0
9	UserData 8	1.000000E+0

Not all of the displayed values are necessarily used. This depends on the process. Therefore, undefined values (usually very high or very low) might show up in the table. Just pretend they are not there!

Fig. C2. The Run Options dialog. This dialog is used to control important settings of the simulation

Stages: Enter the number of time steps (stages) the simulation should run.
Save interval: Enter the frequency of saving out *Elle* files. If set to 10, a file will be saved every 10th time step.
Save file root: The filename of the saved files. Each file will begin with the filename (e.g. growth, see Fig. C2) followed by the stage number and the ending elle. A complete filename of a single process run therefore looks like this: growth1200.elle.
Save pics root: Next to the *Elle* file (which is a regular ASCII-text file), pictures in various formats can also be saved during the simulation. Enter the root filename of the pictures here.
Display frequency: Only update the *Elle* window every x-th stage.
Switch distance: Adjust the spacing between nodes.
Speed up: Setting this option to anything else than 1 will speed up (or slow down in case of values <1) the simulation. If the speed up is set to 1 and the node would move a distance of 0.25, a speed up of 2 would cause the node to move a distance of 0.5, a speedup of 10 would cause a movement of 2.5 etc. Use with caution since large node movements frequently cause topology problems.
Save log file: During the simulation, the logging messages can be saved out to a new file or appended to an old one. Several log-levels are available: *quiet* will only show the most important messages *medium* will show some debugging information *full* will show a lot of information. This is only useful for debugging.
User Data: Any input data read from the command line or via an associated zip file.

C.5 The Preferences Window

The preferences window can be opened from the **Graphics menu**. It will open as a book with tabs for flynns, bnodes, unodes etc. It is possible to load and save the settings made in the preferences dialog by using the load and save buttons. Apply applies all the settings to the current microstructure but does not close the preferences window. Ok applies all the changes and closes the preferences window. Cancel does not apply the current changes (nor does it reset changes already made to default values) and closes the window.

C.5.1 Flynn Preferences

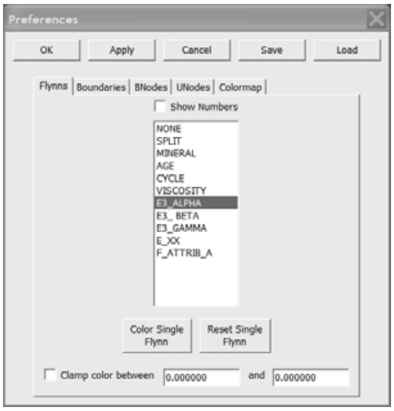
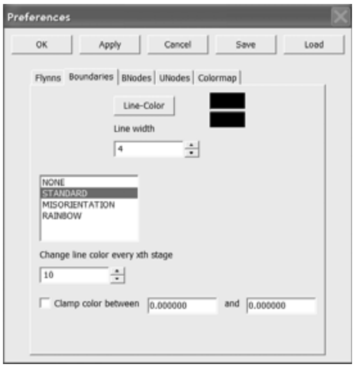


Fig. C3. Preferences page of flynn settings

Show Numbers: Shows the flynns numbers.
List of attributes: The flynn attributes are shown here. Chose one of them to colour the flynns accordingly.
Color single flynn: Give the number of a flynn and it's colour that should be marked. The colour will be displayed during runs so it is possible to (visually) mark a flynn during a simulation.
Reset single flynn: Reset the colour of a flynn again to its normal state (according to its attributes). Asks for the flynn number to be reset.
Clamp color between: Limits the range of the colour look up table to the lower and upper values defined to the right of this check box.

C.5.2 Boundaries Preferences



Line width
4

NONE
STANDARD
MISORIENTATION
RAINBOW

Change line color every xth stage
10

Clamp color between 0.000000 and 0.000000

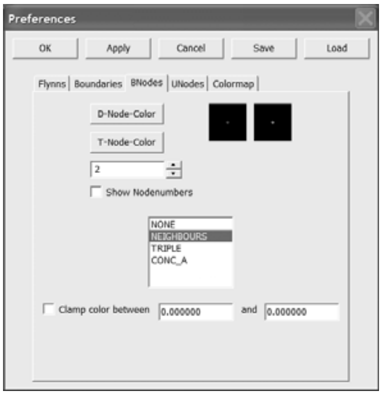
Fig. C4. Preferences page of boundary settings

Line-color: The line colour and its size can be changed. The current colour and size are shown in the box next to the button.
Line width: The line width can be adjusted using the text field.
List of attributes: It can be chosen in which way the boundaries are displayed. If the rainbow option is chosen, the colour of the boundaries will change according to the current colour map. The frequency of colour changes can be adjusted (see below).
Change line color every xth stage: This adjust how often the line colour changes if the rainbow option is chosen. A value of 10 for example indicates that the line colour will change every 10 time steps. This option only has an effect if the rainbow option is chosen, and only makes sense if "Always Overlay" is turned on in the graphics menu.
Clamp color between: Limits the range of the colour look up table to the lower and upper values defined to the right of this check box.

C.5.3 Bnode Preferences

D-Node and T-Node color: The colour of double and triple nodes can be changed independently, the size of nodes is the same for both. The respective colours of double and triple nodes and their sizes are shown in the boxes.
Show Node numbers: If checked this will display all the node numbers in the main window. This is only useful during zooming as there are (usually) far too many nodes so the node numbers cannot be read.

List of attributes: As with flynns and boundaries, the attributes chosen here will be used to colour the nodes.
Clamp color between: Limits the range of the colour look up table to the lower and upper values defined to the right of this check box.



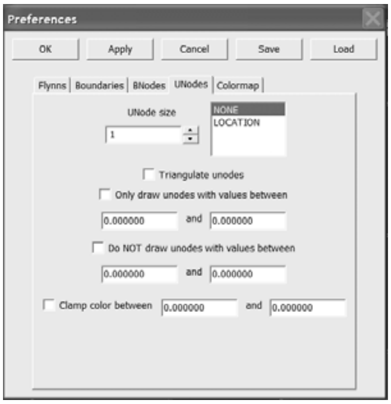
2
Show Nodenumbers

NONE
NEIGHBOURS
TRIPLE
CONC_A

Clamp color between 0.000000 and 0.000000

Fig. C5. Preferences page of bnodes settings

C.5.4 Unode Preferences



UNode size 1

NONE
LOCATION

Triangulate unodes
Only draw unodes with values between
0.000000 and 0.000000
Do NOT draw unodes with values between
0.000000 and 0.000000
Clamp color between 0.000000 and 0.000000

Fig. C6. Preferences page of unodes settings

Unode size: Adjust the size of the unodes here. This will simply draw every unode with the given size, there is no triangulation or recalculation of unode values!
List of attributes: As with the other pages, the attributes can be chosen here that is used to colour the unodes.

Triangulate Unodes: This will display the unodes as a continuous image, rather than showing each unode separately.
Only draw unodes with values between: This will only colour values within the given range. If you select an attribute other than “none” or “location”, the min and max values of this attribute will be shown in the two text fields.
Do NOT draw unodes with values between: The two fields (draw unodes and NOT draw unodes) can be combined. The following is important to know to get exact results and not to misinterpret the display: unodes that have the values that were entered in the boxes WILL be displayed (unodes with values equal to max or min).
Clamp color between: Limits the range of the colour look up table to the lower and upper values defined to the right of this check box.

C.5.5 The colour map

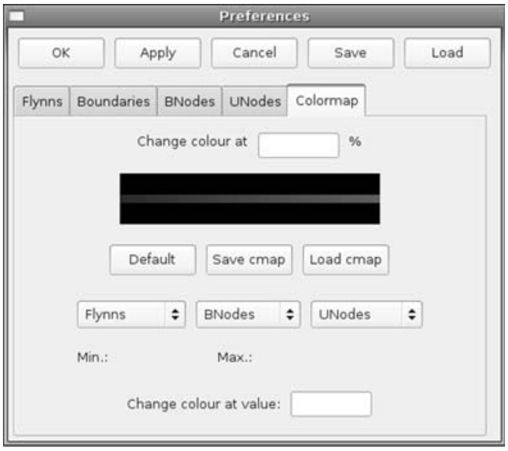


Fig. C7. Preferences page of colour settings

In the colour map window, the colours used to display attributes can be adjusted. The upper text box expects a percentage. An input of 0 will change the colour for the lowest value, an input of 100 will change the highest value. It is not necessary to know the min and max values of the attribute.
Default: The colours can be reset to the original by pressing the default button.
Save/Load cmap: The colour map can be saved or loaded for later reuse. The file saved will be a regular text file with 256 colours. See below for an example file of default.cmap:

```
Colormap for Elle-files... Please do not edit
          0 0 255 1
          1 0 254 0
          [...]
          [...]
          [...]
          254 0 1 0
          255 0 0 1
```

The first number indicates the number of the colour, the next are RGB values of the colour.
Flynns/BNodes/Unodes: If the user wants to change a specific colour value, he has to know the min and max values of the attributes. These might (or better most certainly will) have changed during a simulation. These three drop-down lists can be used to choose an attribute, the min and max values of this attribute will be calculated and displayed. The lower text box can be used to enter a specific value at which the colour will be changed.

C.6 The zooming window

Once the zooming window is opened, you have a few options to choose from: zooming in and out of the structure, moving the structure, stop zooming entirely, get info on bnodes and unodes, number all bnodes.

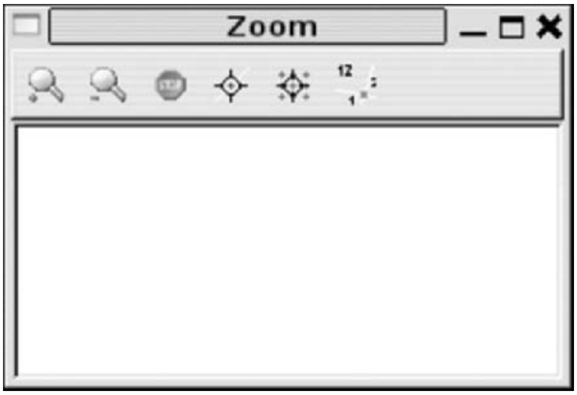


Fig. C8. The zooming window. For explanation of the buttons see text

Zoom into the view. Zooming will always use upper left corner as the the origin.

Zoom out of the view.

Stop zooming. The screen is redrawn to its full size.

Get info on bnodes. With this tool, bnodes can be chosen (by clicking on them in the canvas. The pointer to do so changes to black so you might have a hard time seeing it when no flynn-colours are displayed!) and information about the values of the current bnode are displayed. If you click to far away from a node, the window displays the message No Hit!. If there are no bnode-attributes defined, the window will say so EVEN IF YOU HIT A BNODE!

Get info on unodes. Same as above, only for unodes.

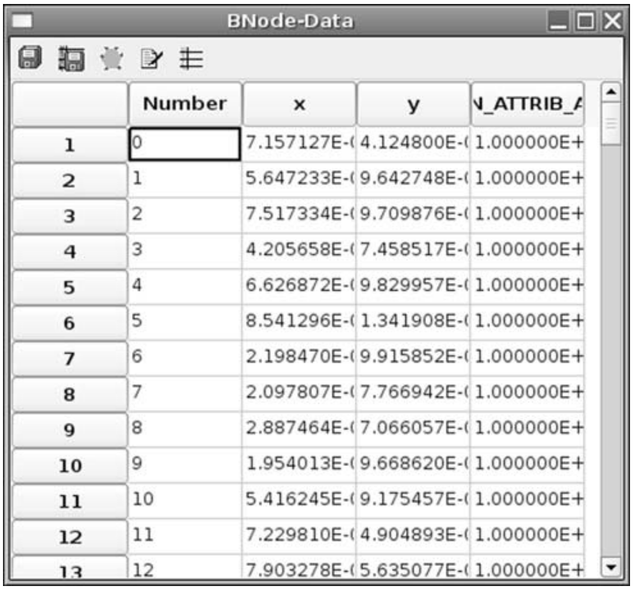
Labels all the flynns and bnodes. Since there might be a lot of bnodes, this only makes sense if the zoom is large.

Another feature of the zoom is that you can zoom in on any area you are interested in and start your simulation. During the simulation, the canvas will only show the area you have zoomed in. It is possible to change the viewing area of the zoom and/or to stop zooming entirely during the simulation.

C.7 The data-windows

The table windows will open if the respective menu under Data is chosen. They will display either flynn, bnode or unode attributes, if there are any defined. The values can be changed by simply clicking on the cell: a small box will pop-up and ask for the new value.

It is not possible to change the numbers of flynns, bnodes or unodes or the location of bnodes and unodes. It is also not possible to insert or delete flynns or bnodes or unodes from the file. If you click on the row-labels, the respective flynn, bnode or unode will be highlighted. Clicking on a row of a previously selected item will deselect it.



	Number	x	y	N_ATTRIB_A
1	0	7.157127E-(	4.124800E-(	1.000000E+
2	1	5.647233E-(	9.642748E-(	1.000000E+
3	2	7.517334E-(	9.709876E-(	1.000000E+
4	3	4.205658E-(	7.458517E-(	1.000000E+
5	4	6.626872E-(	9.829957E-(	1.000000E+
6	5	8.541296E-(	1.341908E-(	1.000000E+
7	6	2.198470E-(	9.915852E-(	1.000000E+
8	7	2.097807E-(	7.766942E-(	1.000000E+
9	8	2.887464E-(	7.066057E-(	1.000000E+
10	9	1.954013E-(	9.668620E-(	1.000000E+
11	10	5.416245E-(	9.175457E-(	1.000000E+
12	11	7.229810E-(	4.904893E-(	1.000000E+
13	12	7.903278E-(	5.635077E-(	1.000000E+

Fig. C9. Example of a data window (here bnode-data are displayed). If no attributes for nodes or flynns have been set, the window will not show anything. This is a *feature*, not a bug!

Save all data as text file. The floppy disk icon lets you save all the data.

Save selected data as text file. This icon lets you only save the selected data.

Change multiple values. If more than one point should be changed, it is possible to mark the respective area by clicking on the icon. You will then be able to mark an area in the canvas by drawing a polygon around the area you want to change. Note that only flynns, bnodes or unodes that are completely within the polygon will be highlighted and hence changed. The area that is marked can not have more than 1000 vertices. The last vertex is set by double-clicking on the desired position (left-double-click).

If multiple entries are selected (using the polygon tool described above) you can simultaneously change all of their values with this tool. If you have lots of flynns or nodes highlighted you may not see some of them. In that case, either scroll down until you find one, click on the desired cell and change the value for all of them or press this icon: a message box will appear asking you for the new value.

Deselect Rows. You can deselect the rows by clicking on this icon.

Appendix D Different ways to run an *Elle* experiment

Mark W. Jessell

At a minimum, an *Elle* experiment consists of a single input file (an *Elle* file) that describes the starting microstructure, and a single *Elle* processes, that describes the effect of a single process on that microstructure. Single-process experiment may be run in graphics mode, with continual update of the microstructure visualisation. Multi-process experiments must always be run via a shell script (called a batch script on the Windows platform), and the results of the experiment may be followed by opening output files using the *showelle* microstructure display tool.

D.1 Via the *Experiment Launcher*

The *Experiment Launcher* is a simple program that allows you to launch any of the example experiments in this Appendix. Once started up, the Launcher offers a single menu and two panels: the **File menu**, the **Examples panel** and the **Utilities panel**. The **Experiment panel** and its submenus list all the experiments in this Appendix. To run a particular example, select it from the **Experiment menu** and you will be provided with a brief overview of the experiment. If you wish to start the experiment, simply click on the **Go! button**.

Single-Process Experiments If a single-process was started, an *Elle* window will open up showing the starting microstructure. To enact the experiment select **Run→Run** from within the *Elle* Window. See Appendix C for details on graphical user interface for single processes.

Multi-Process Experiments If a multi-process was started, the experimented will be started automatically in the background, and to view any of the output *Elle* files, select *showelle* from the **Utilities menu** of the Launcher, and load in the desired file. See Appendix C for details on graphical user interface for *showelle*. All output files will be created in the same directory as the experiment, and have a naming convention described in Appendix E.

Starting Utilities The Experiment Launcher may also be used to launch utilities such as *showelle*, without any input file pre-loaded.

D.2 Direct launching of experiments

Single-Process Experiments To run a single-process experiment, double click on the icon for the particular process (e.g. *elle_gg*), go to the **File menu** and **Open** an *Elle* file (e.g. exp3.elle) If there is a zip file with the same suffix as the *Elle* file in the same directory (e.g. exp3.zip) it will be loaded at this time, and it will define the experimental conditions and the graphical display properties of the experiment. To run the experiment, simply select **Run** from the **Run menu**. The graphical display will then be updated each time step, or less frequently, depending on the display options. A complete description of the *Elle* graphical interface, which is common to all processes, may be found in Appendix C.

Multi-Process Experiments To run a multi-process experiment, double click on the batch file (e.g. exp4.bat for windows) and a window will open up giving some runtime debugging information. Any output files will be saved in the same directory as the batch file, and the *Elle* files may be viewed by starting *showelle*, loading the *Elle* file. If there is a zip file containing the graphics preferences called defaults.zip, this will be loaded automatically. A web page to create new shell scripts/batch files is described in Appendix E. If *Basil* was used in the experiment, and its post-processor *Sybil* was installed, detailed information on the stress and strain state within the microstructure may also be displayed (see Appendix F).

D.3 Developer mode (Windows and Linux)

In addition to the GUI based modes described above, Windows and Linux Developers may also run *Elle* experiments from the command line. In Windows, you need to start up *MSys*, which will provide a shell, and in Linux, you need to open a new shell window.

Single-Process Experiments To run a single-process experiment, one command line call is needed with options specifying information such as the number of steps to be simulated, the input and output file and the simulation mode.

Synopsis: `process_name [-cefhinsuv]`
Options:

`f`	the next argument will set the frequency for saving the *Elle* data during a run; i.e. 10 will mean that the data is saved after every 10th stage
`h`	help, list the valid options
`i`	the next argument will be the filename of an *Elle* input data file
`n`	no display (run without the graphical display)
`s`	the number of stages to run (default 20)
`u`	may be followed by up to 9 numbers which allow the user to vary parameters defined for a particular process e.g. *elle_gbdiff* interprets the first value as the diffusion rate.
`v`	the boundary velocity (only used by *elle_manuel*)

These are the only options, which can currently be changed at runtime when executing in batch mode.

An example of calling an *Elle* process:

```
        elle_pblast –i start.elle –s 25 –f 25 –n
```

will run *elle_pblast* in batch mode (no display), reading the input file start.elle, processing 25 stages then saving the result. Only one file is saved as it is set to save every 25th stage. The options set by the command line will override the runtime and userdada setting read from a zip file (see App. D.2).

Multi-Process Experiments To run a multi-process experiment, change to the directory which contains the batch file and execute the batch file directly: ./exp4.shelle.

Any output files will be saved in the same directory as the batch file, and the *Elle* files may be viewed by starting ***showelle***, loading the *Elle* file. If there is a zip file containing the graphics preferences, this can be manually loaded by selecting **Preferences** from the **Graphics** menu, then clicking on the **Load button**. A web page to create new shell scripts/batch files is described in Appendix E. If ***Basil*** was used in the experiment, detailed information on the stress and strain state within the microstructure may also be displayed using the *Sybil* post-processor (see Appendix F).

Appendix E How to create multiple-process experiments

Lynn Evans

E.1 Shelle scripts/batch scripts

Running multiple processes requires the use of a control script, which we call a Shelle Script, this is in fact a shell script for under Linux and MSys, and a Batch Script under Windows, which defines the behaviour of the experimental run (Fig. E1):

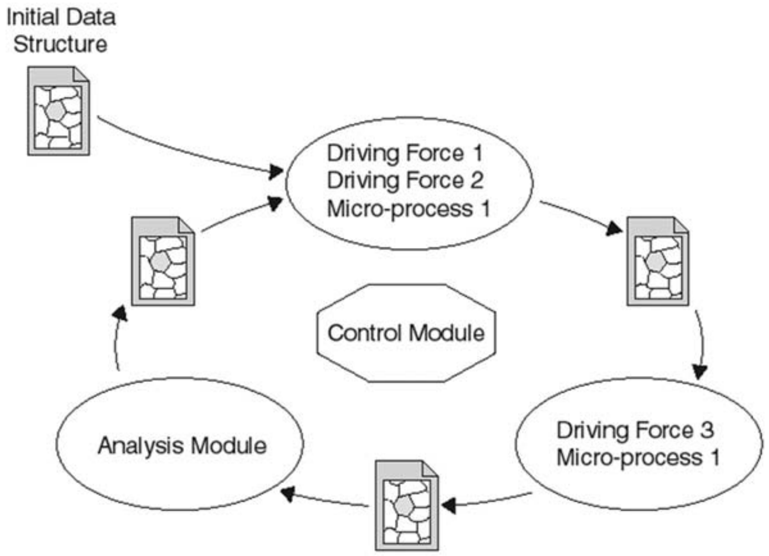


Fig. E1. *Elle* control module controls the way in which processes are executed and data is stored

We have simplified the task of creating these control scripts by developing a GUI that allows the user to specify the steps and parameters required for their simulation then generate and run the control program. Once created, it is possible for the user to write or edit the resulting text file that is a shell script similar to elle/examples/init/control.template. All the controlling variables, e.g. STAGES, LASTTIME, are set in the first section above the line that looks something like:

```
LIST="1 2 3 13 4 16 15"
```

LIST specifies the processes in the order in which they will be executed. This line may be edited if different processes are desired. If DEBUG is set to echo, the program will write the command line to the screen rather than running it. This is useful for checking that the program calling order and parameters are correct.
To run a simulation in the background type:

```
NOHUP CONTROLFILE >& DUMPFILE &
```

where controlfile is the name of the file created by the shelle script generator and dumpfile is the output redirected from the screen and stderr. The dumpfile can be deleted if the simulation runs successfully.

E.2 Shelle: An *Elle* Shell Script Generator (v1.24)

In order to run more than one process, it is necessary to create a shell script that will call each process in turn, taking into account all the file names and formats needed to ensure a smooth transfer of information.

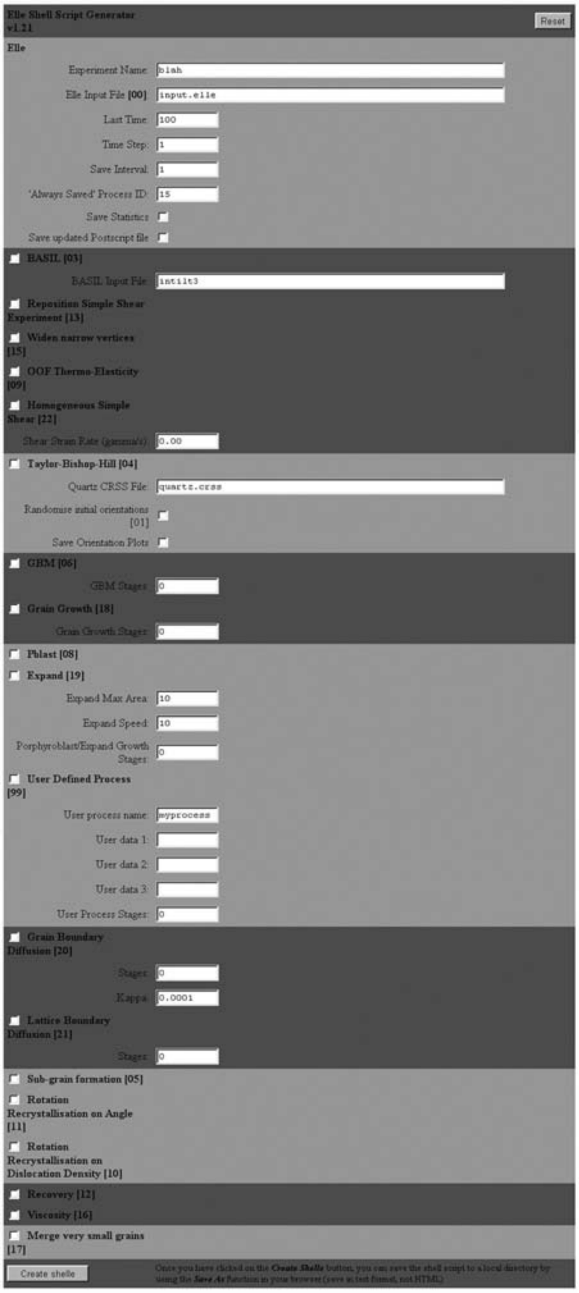
The *Elle* implementation page contains a link to a web page that lets you fill in a form that then automatically generates the appropriate shell script. It currently allows you to create a shell script for any combination of the processes listed in this manual.

E.2.1 How to use the Shelle Script Generator

Select the **Shelle Script Generator** from the **Utilities** section of the Elle Experiment Launcher and click on **Go**. Alternatively, start up a JavaScript compatible web browser (not as scary as it sounds, any Firefox browser or any Internet Explorer version from 3.0 will do) and select the Shell script generator:

- http://www.microstructure.info/elle/shelle.html
- or elle/utilitities/shelle/shellev24.html.

You will be presented with a form like Fig. E2.



Elle Shell Script Generator v1.21

Elle

Experiment Name:	blah
Elle Input File **[00]**:	input.elle
Last Time:	100
Time Step:	1
Save Interval:	1
'Always Saved' Process ID:	15
Save Statistics	☐
Save updated Postscript file	☐

☐ **BASIL [03]**

BASIL Input File: intilt3

☐ **Reposition Simple Shear Experiment [13]**

☐ **Widen narrow vertices [15]**

☐ **OOF Thermo-Elasticity [09]**

☐ **Homogeneous Simple Shear [22]**

Shear Strain Rate (gamma/s): 0.00

☐ **Taylor-Bishop-Hill [04]**

Quartz CRSS File: quartz.crss

Randomise initial orientations [01] ☐

Save Orientation Plots ☐

☐ **GBM [06]**

GBM Stages: 0

☐ **Grain Growth [18]**

Grain Growth Stages: 0

☐ **Pblast [08]**

☐ **Expand [19]**

Expand Max Area:	10
Expand Speed:	10
Porphyroblast/Expand Growth Stages:	0

☐ **User Defined Process [99]**

User process name:	myprocess
User data 1:	
User data 2:	
User data 3:	
User Process Stages:	0

☐ **Grain Boundary Diffusion [20]**

Stages: 0

Kappa: 0.0001

☐ **Lattice Boundary Diffusion [21]**

Stages: 0

☐ **Sub-grain formation [05]**

☐ **Rotation Recrystallisation on Angle [11]**

☐ **Rotation Recrystallisation on Dislocation Density [10]**

☐ **Recovery [12]**

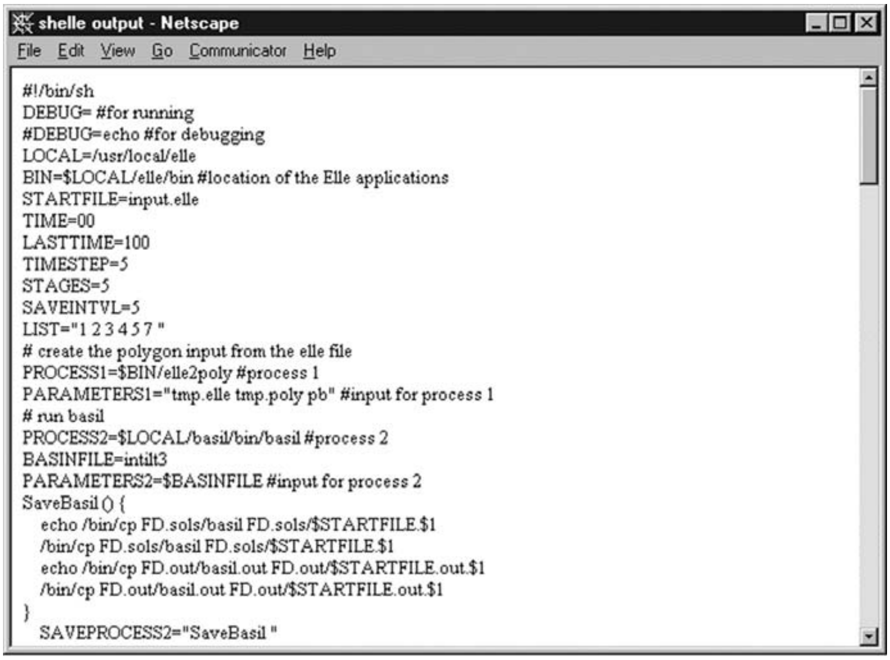
☐ **Viscosity [16]**

☐ **Merge very small grains [17]**

Once you have clicked on the ***Create Shells*** button, you can save the shell script to a local directory by using the ***Save As*** function in your browser (save as text format, not HTML)

Fig. E2. The shelle script generator

To activate a process, click on the check box to the left of its name, and fill in the appropriate information in the subsequent fields. Then click on **Create Shelle** for Linux, and Windows Developers using *MSys*, or **Create Batch** for Windows Users and a new window will appear that contains the generated shell script.



```
#!/bin/sh
DEBUG= #for running
#DEBUG=echo #for debugging
LOCAL=/usr/local/elle
BIN=$LOCAL/elle/bin #location of the Elle applications
STARTFILE=input.elle
TIME=00
LASTTIME=100
TIMESTEP=5
STAGES=5
SAVEINTVL=5
LIST="1 2 3 4 5 7 "
# create the polygon input from the elle file
PROCESS1=$BIN/elle2poly #process 1
PARAMETERS1="tmp.elle tmp.poly pb" #input for process 1
# run basil
PROCESS2=$LOCAL/basil/bin/basil #process 2
BASINFILE=intilt3
PARAMETERS2=$BASINFILE #input for process 2
SaveBasil () {
   echo /bin/cp FD.sols/basil FD.sols/$STARTFILE.$1
   /bin/cp FD.sols/basil FD.sols/$STARTFILE.$1
   echo /bin/cp FD.out/basil.out FD.out/$STARTFILE.out.$1
   /bin/cp FD.out/basil.out FD.out/$STARTFILE.out.$1
}
   SAVEPROCESS2="SaveBasil "
```

Fig. E3. Output of the shelle script generator (Shell Script)

If you like what you see, you can save this shell script by using the **File→Save As** menu from the browser window, remembering to change the save as type to Plain Text (**txt)**. If you don't like what you see, go back to the input form window, click on reset and try again.

In order to actually use this script under Linux or using Windows Developers using Msys you will have to change its privileges to executable: if the shell script is called myshell.txt, type in

```
        chmod +x myshelle.txt
```

in the same directory as the file (or use your file manager to access the Permissions section of the file properties and set Exec).

E.2.2 Notes on the Shelle Script

a) *File name convention:* Output elle files resulting from using this shelle script will be of the form **Experiment_Name.xxx.yy.elle** where xxx is the time step and yy is the process code (shown above in square brackets), as follows:

00 = input file
01 = following initial randomisation of crystal axis orientations
03 = following a BASIL step
04 = following a TBH step
05 = following a Subgrain Split step
06 = following a GBM step
08 = following a Porphyroblast growth step
09 = following an OOF step
10 = following a Nucleation/Rotation Recrystallisation step
11 = following a Misorientation/ Rotation Recrystallisation step
12 = following a Recovery calculation step
13 = following a Repositioning step
15 = following an Angle check step
16 = following a Viscosity calculation step
17 = following a Merge calculation step
18 = following a Grain Growth step
19 = following an Expand step
20 = following a Grain Boundary Diffusion step
21 = following a Lattice Diffusion step
22 = following a Homogeneous Simple Shear step
99 = following a User Defined Process step

b) *Randomise Orientations:* When checked, this option randomises the lattice orientations of any quartz grains, which have not already been defined, one time only, at the start of the experiment.

c) *Save Orientation Plots:* When checked, at the save interval frequency, c-, a- and r- axis pole figures of quartz grains will be saved out as post-script files with the naming convention *ellefilename_ax.ps* .

d) *Reposition Simple Shear Experiment:* This normalises the elle space to a square after each BASIL deformation step.

e) *Widen narrow vertices:* Angle check to removes narrow vertices from flynns to make triangulation better.

f) *Merge calculation:* Removes very small grains.

g) *Statistics:* Simply outputs information to file.

h) *Save updated Postscript file:* If this option is turned on, a postscript file of the current *Elle* microstructure called *tmp.ps* is created (using *showelle.in* defaults), if you use *ghostview* with the "watch file" option enabled, the window will update with the new file each save step.

E.2.3 Debugging Shell Scripts

It is advisable to test the shell script without actually running the processes first. To do this use your favourite text editor to remove just the first # from the third line, which reads #DEBUG=echo #for debugging, so it reads DEBUG=echo #for debugging. If you run the script now it will go through the motions of the script without actually doing anything. If the results look good, cross your fingers, put the # back and run the script.

Appendix F *Sybil* - The *Basil* post-processor

Lynn Evans, Terence Barr and Greg Houseman

F.1 Introduction

Sybil is the graphical post-processor for the *Basil* Finite Element code, and is a stand-alone program that reads in *Basil* solution files (which will be found in a directory called FD.sols within the directory containing the *Elle* experiment. It can also overlay grain boundary maps from *Elle* files.

Windows To start *Sybil* from Windows, first start up the *Experiment Launcher* and then select *Sybil* from the Utilities Menu.
Linux To start up *Sybil*, type **`./sybil`** in the same directory as your experiment. You will then be presented with a blank window, with a series of menus at the top

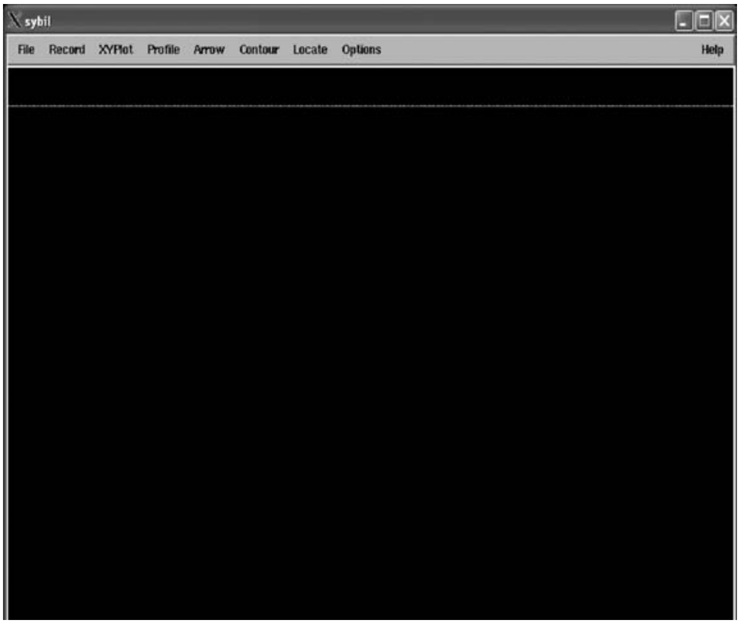


Fig. F1. *Sybil* window at start-up

You can use the various Menu Options, that are outlined below, to display the stress and strain state of the microstructure at different stages of the experiment.

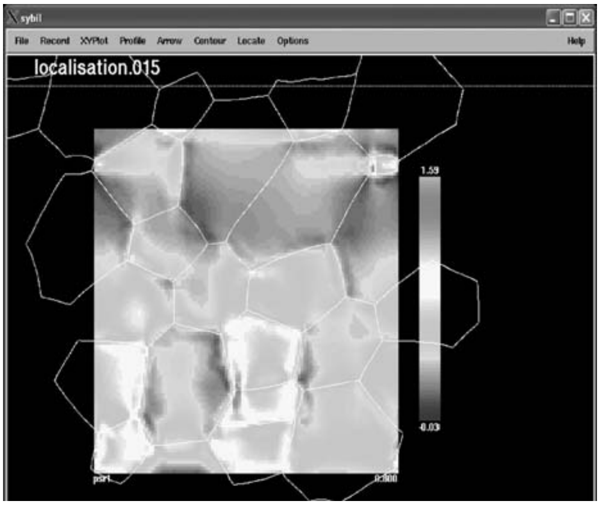

sybil
File Record XYPlot Profile Arrow Contour Locate Options Help
localisation.015
1.59
-0.03

Fig. F2. *Sybil* window showing strain state in a microstructure

F.2 Menu Options

F.2.1 File

1. **Open**: opens a file, either **Solution**: to open a solution file, or **Log**: to open and execute a previously saved log file
2. **Save**: writes a file, either **Log**: to write a log file that recreates the current drawing, or **Log+Postscript**: to write the log file, and then cause *sybilps* to execute and write a postscript file of the current drawing.
3. **Exit**: terminates execution of *Sybil*, after checking.

F.2.2 Record

1. **Current**: selects the solution record for subsequent drawing, **Next**: selects the next record in the current file, **Previous**: selects the previous record in the current file, **Last**: selects the last record in the current file, **Specify**: type input number of solution record (1 is the first record and the value of max in the info line at the bottom of the window is the last record)
2. **Reference**: selects the solution record to define an initial geometry relative to which subsequent deformations are measured, **First**: uses the first record of the current file, **Specify**: uses the record number you identify, as above.

F.2.3 XYPlot

1. **Mesh**: creates an X-Y plot of the Finite Element mesh, either **elements**: outline of all elements in the mesh **element+num**: outline as above plus element numbers of some elements **boundary**: external mesh boundary only **viscosity**: shows regions with anomalous viscosity
2. **Deform**: draws deformation indicators relative to a previously saved reference record (see **Record→Reference** above), **ellp**: draws finite deformation ellipses for a subset of triangles **tria**: draws the subset of triangles used for the ellipse calculations **trel**: draws ellipses and triangles, as specified by two preceding options.
3. **Bounding box**: draws a rectangular box around the current plotting area, excluding margins.
4. **LGMesh**: draws the Lagrangian mesh that may be used by *Basil* to track deformation, **elements**: draws the entire Lagrangian mesh **boundary**: draws the boundary only
5. **StrainMark**: draws any strain markers defined by *Basil* to track internal deformation
6. **Elle**: Draws the grain boundary network defined in an *Elle* File

F.2.4 Profile

1. **1_D**: constructs a 1-D profile of selected parameters between 2 points on the *x*-*y* plane. After selecting a quantity to profile from the lists below, input of the *x*-*y* coordinates of the two profile endpoints is requested. Min and Max values on the profile are then shown and, the user must input values for the Min and Max on the vertical axis of the profile plot. Default values are the min and max values of the function on the first call, and scale is preserved for subsequent profile plots. **Velocity**: select from the following parameters: **Ux**: *x*-component of velocity, **Uy**: *y* component of velocity, **Ur**: radial component of velocity (relative to local origin), **Uth**: tangential component of velocity. **Strain**: select from the following quantities related to strain-rate: **edxx**: the *xx*-component of strain-rate, **edyy**: the yy-component of strain-rate, **edzz**: the *zz*-component of strain-rate, **edxy**: the *xy*-component of strain-rate, **psr1**: maximum principal strain-rate in *x*-*y* plane, **psr2**: minimum principal strain-rate in *x*-*y* plane, **msst**: maximum shear strain-rate in the *x*-*y* plane, **cang**: orientation of principal compressional axis,, **tang**: orientation of principal extensional axis, **sang**: orientation of direction of maximum shear, **dblc**: parameter that indicates type of faulting, **vort**: vertical component of vorticity, **ed2i**: 2nd invariant of the strain-rate

tensor, **vota**: ratio of vorticity to shear strain-rate. **Stress**: select from the following quantities related to stress, **taud**: direction of maximum deviatoric stress, **taum**: maximum deviatoric shear stress, **taxx**: the *xx* component of deviatoric stress, **tayy**: the *yy*-component of deviatoric stress, **tazz**: the *zz*-component of deviatoric stress, **taxy**: the *xy*-component of deviatoric stress, **tau1**: maximum principal deviatoric stress, **tau2**: minimum principal deviatoric stress, **sixx**: the *xx*-component of total stress, **siyy**: the *yy*-component of total stress, **sizz**: the *zz*-component of total stress, **sig1**: maximum principal total stress, **sig2**: minimum principal total stress, **thdi**: thermal dissipation function (stress*strain-rate), **pres**: the pressure field, **brit**: parameter indicating type of faulting, **bri2**: parameter indicating type of faulting, **visc**: effective viscosity parameter, **Thickness**: layer thickness in the *z*-direction, **Density**: density parameter.

2. **2_D**: constructs a horizontal or vertical profile of selected quantities integrated in the orthogonal direction. The same set of parameters can be profiled, as listed above for 1_D. After choosing the quantity to be profiled, choose whether an *x*-direction, or a *y*-direction profile is required, and input the upper and lower limits of the orthogonal integra- tion variable. E.g. To show how the integral of *Ux.dy* varies as a function of *x*, choose the *x*-direction profile and the two limits are the extreme values of *y* for the integration. Avoid integrating outside the solution region if meaningful results are required. Input of vertical axis limits is then requested, as for 1_D profile.
3. **Mark**: is used to draw a line in the current cell showing the path of the 1-D profile in the *x*-*y* plane. The profile path can be plotted over a previously drawn contour plot of the variable that is to be plotted, in order to show where the profile has come from, and to check correlation with features in the 2D plane. After entering the coordinates of the 2 endpoints, use preview to check position of the profile that can then be amended as necessary. The Mark command can be used also for drawing any additional lines required on a *Sybil* drawing, regardless of whether a profile plot is required.

F.2.5 Arrow

1. **Velocity**: constructs an arrow plot, consisting of equally spaced arrow heads, whose direction is parallel to the velocity field and whose length is proportional to the magnitude of the velocity. The spacing of arrows can be set under **Options→Plot→mp and np** (refer ‘Options’ below).

2. **Strain:** constructs an arrow plot, consisting of equally spaced crosses, whose orientation indicates the directions of the principal axes of strain-rate. For direction-only plots all crosses are the same size, for magnitude plots, the length of each arm is proportional to the magnitude of the principal axis. Options are: **pstm**: Principal strain-rate magnitudes, **pstd**: Principal strain-rate directions only, **mssr**: planes and magnitudes of maximum shear strain-rate, **msft**: direction of strike-slip faulting planes,
3. **Stress**: constructs an arrow plot as for Strain (see above), but with magnitudes proportional to the relevant stress quantity. Options are: **taum**: principal deviatoric stress magnitudes, **taud**: principal deviatoric stress, directions only, **sigd**: principal stress (total), directions only, **sigm**: principal stress (total) magnitudes.

F.2.6 Contour

The Contour command is used to display 2-D quantities using a graphical display in which areas are coloured according to the value of the physical parameter being displayed, or equally spaced contours of this physical parameter are drawn, or both. The same set of physical parameters listed above under **Profile→1_D** is available for contouring. After choosing the parameter to be contoured, the user is asked to confirm or enter a set of parameters that affect the display. Shown at the top left are the actual minimum and maximum values of the quantity to be contoured. The two values at the top right, entered by the user, determine the mapping of colour to number. With the default colour scale, blue is mapped to the minimum, and magenta to the maximum. Contours, if present, are chosen in equal increments of 'Step' above and below 'Level'. A maximum number of contours is also available. The number labelled scale is only used to multiply the labels that appear on the colour scale. The settings are preserved for successive calls to 'Contour' so that the default setting usually is to keep the same scales as for the preceding plot. Various default parameters affecting the display may be set using the **Options** command (see below).

F.2.7 Locate

The Locate command is used to specify which of the current drawing cells is active, as follows:

1. **Next**: Move to the next cell, either the cell to the right, or if already at the right limit of the page, the left most cell of the next row.
2. **Prev**: Move to the previous cell.

3. **Specify**: Increment or decrement the row or column number of the current cell address using the buttons provided in order to move directly to a particular cell.

F.2.8 Options

1. **Label**: is used to add labels to the plot as follows:
2. **Edit**: enter a string to be drawn on the plot. Then click with the left mouse button at the point on the drawing where the lower left corner of the string is to be located. The same string can be added to the plot as many times as required and the string can be edited at any time in the labelling step.
3. **Font**: choose one of the 2 fonts (Helvetica, Symbol) and 5 or 6 font sizes (8, 10, 12, 14, 18, 24) that are provided as options.
4. **Colour**: choose one of the 8 standard colours to be used in subsequent drawing and labelling operations.
5. **Line**: choose solid, dashed, or dotted line for subsequent drawing operations.
6. **Rescale**: On making the first plot, the scale is set automatically so that the entire solution region will appear within the plotting cell, including a small margin on each side. Subsequent plots are drawn at the same scale, with the origin shifted to the new drawing cell. The scale will be automatically reset for the next plot, only if this option is first selected. Judicious choice of cell numbers and margin widths may be necessary to obtain a particular scale.
7. **Delete**: deletes previous drawings from the screen. Note that the first option is also available at any time by locating the cursor within the current cell and pressing the middle button of the mouse.
8. **Current cell**: only deletes contents of one cell
9. **All Cells**: deletes contents of all cells
10. **Plot**: requests input of certain options and parameters that affect the visual display, as follows: **nx3**: determines the number of interpolation points to be used in contouring and other operations. A number in the order of 100 is recommended. Greater numbers will improve the resolution of the plot, but will require more time to plot. **mp, np**: in constructing arrow plots (see above) an arrow is situated at every (mp)th interpolation point in the *x*-direction and every (np)th interpolation point in the *y*-direction. The size of the maximum arrow is equal to the distance between arrows. Numbers of 4 or 8 are recommended. Note that if nx3 is doubled, mp and np should be doubled to preserve the same arrow spacing. **profile_pts**: the number of sample points to take along a

given profile, **stipple**: if > 0, stippling is added to contour plots in specified regions, with reference to the Level parameter (see Contour above): stipple = 1 → stipple if below Level, stipple = 2 → stipple if above Level, stipple = 3 → stipple if abs value above Level, stipple = 4 → stipple if abs value below level. **solution_rot**: This parameter should be a number between 1 and 3, to rotate the solution by quanta of 90 degrees anticlockwise. The rotation is done immediately that the solution is input, and any reference to *x* and *y*- coordinates then applies to the solution as displayed in its new reference frame. **dble**: Causes most area plots to be displayed twice, with a horizontal offset equal to the width of the solution region. This option is only used for solutions with periodicity in the *x*-direction. **label**: causes all automatic labelling to be switched on or off. This option does not affect any labels added manually, as above. **flip**: causes the *x*-axis to be reversed on the display. **Contour Options**: either or both of the following may be selected. **Lines**: contour lines are added to all contour plots. **Shading**: shading by colour is added to all contour plots. **Bar**: the colour bar for contour plots may be vertical, to the right of the plot, or horizontal, underneath the plot, or may be omitted. **Tics**: ticks on profile plots may be internal to the plot, external, centred, or omitted.

11. **Verbose**: outputs numbers (lots of them) to standard out, from profiles, contour plots, arrow plots and strain-marker plots, in order that other external processing may be applied to these data. This option should only be turned on temporarily to extract a particular data set and is not yet implemented uniformly.
12. **Mark Cell**: shows the current plotting cell by a line drawn around its perimeter. This boundary line does not appear on the postscript plot.

Appendix G How to use *Elle_Latte*

Till Sachau and Daniel Koehn

G.1 Introduction

Elle_Latte (simply called *Latte* in this appendix) is a Lattice-Spring code that is included in *Elle*. Latte is written as a separate process that uses *Elle* functions and the *Elle* data set.

Latte is build up by three main classes, the particle class, the lattice class (lattice.cc, lattice.h, particle.cc and particle.h) and the experiment class (experiment.cc, experiment.h). A third basic class that inherits the lattice class is used as a base class for phase transformations (base_phase.cc, base_phase.h). This base phase class is inherited by the classes phase_lattice for fluid-solid reactions (phase_lattice.cc, phase_lattice.h) and min_trans_lattice for solid-solid phase transformations (min_trans_lattice.cc, min_trans.h). In addition a third class is used for heat diffusion (heat_lattice.cc, heat_lattice.h) and a class that includes the grain growth process (graingrowth.cc and graingrowth.h).

The User does not have to be concerned with these classes directly as we have developed a graphical interface where the user can choose one of the processes that is included in the experiment class and can change parameters without having to recompile.

Functions are basically separated into initialization functions that give different flynns, single particles or defined layers different properties, and run functions that define the proposed deformation or internal changes and dump statistics to plot for example stress-strain curves. The user can run an experiment in two possible ways: a) use the interface to select the desired process and its parameters or b) change parameters in the experiment.cc class, recompile and select the process when calling *Latte* using the **-u command**.

G.2 The experiment interface for Windows and Linux users

G.2.1 Introduction

As an attempt to simplify the handling of *Latte* and to avoid users to get into the *Latte* source code and recompile it every time to adapt it to their needs, a simple graphical user interface (GUI) has been developed. This GUI is not linked to the *Latte*-executable itself, but calls it as an external process. Thus it is designed as a graphical editor for a preference-file that *Latte* can read and execute. The examples given in the book can be started either from this interface or, alternatively, directly from the command-line. Functions that can be controlled from the interface are described by a simple tool-tip, while the mouse-pointer moves over a particular button.

As an alternative to run *Latte* directly from this window, the preference file in the generated zip-archive can be accessed from *Latte* by starting it directly from the command-line with

```
./latte_elle -u 0 -i inputfilename
```

The inputfilename is identical to the name of the zip-archive, apart from the file-ending, which will be '.zip' instead of '.elle'.

G.2.2 Installation notes

The interface is linked with wxWidgets - even though *Elle*/*Latte* can be installed using *lesstif* instead – therefore the Wx-package has to be installed on the system in order to use the interface.

On start-up the program needs to read an xml-file, called lattestart.xrc, containing mainly graphical information about the GUI. This file will be looked up in the directory, where the interface was started, and, if not found there, in the directories given by the 'PATH' variable (on *nix systems). Thus, if the executable will be copied to another location, either the xrc-file has to be copied to this location too, or the bin directory of the *Elle* package has to be included in the mentioned variable.

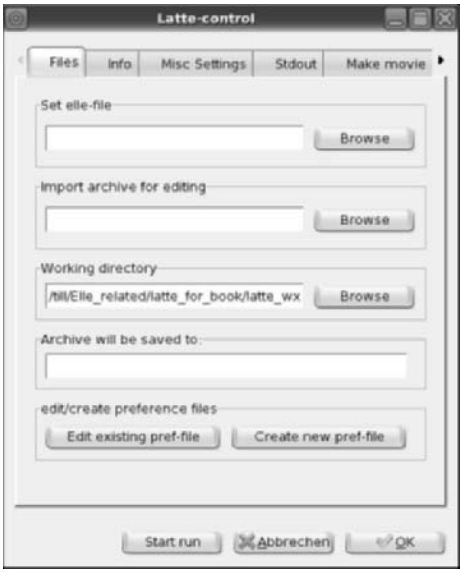


Fig. G1. Interface-window on program start-up. Different functionalities can be accessed via tabs. In the open 'file'-tab, the necessary settings for a single simulation can be done

G.2.3 The main-window

The main window uses tabs, which give access to different basic functionalities of the interface. The following description is thus subdivided according to those tabs. The **Start**, **Cancel** and **OK** buttons are related to the *Latte*-executable itself and can be accessed from every tab.

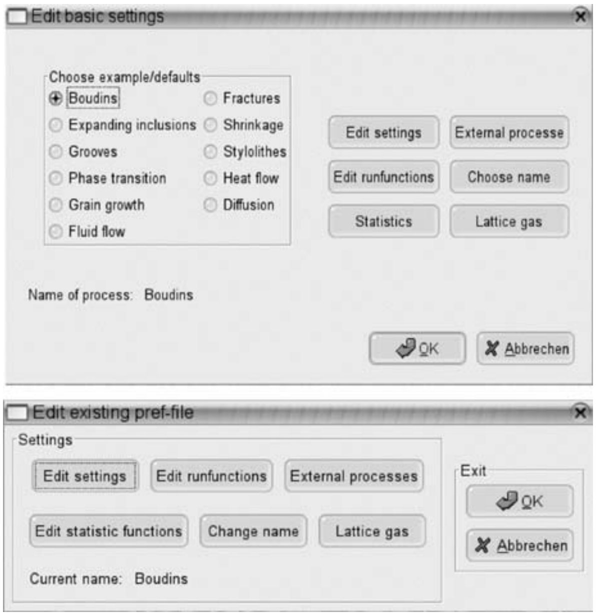
G.2.4 The 'Files' tab

Latte can be almost entirely controlled with the **Files** tab alone. Users can navigate to a working directory, where the input-file is located, and also edit and/or import preference-files from already existing zip-archives. The name of the created archive will be the same as the name of the input-file, except for the file ending.

To prepare the execution of *Latte*, two things have to be done: first an input-file has to be chosen by clicking the **Choose start-file** button, and browsing to an *Elle* file. Clicking **Create new pref-file** gives the opportunity to choose one of several predefined samples, which will set the preference file to useful default values, or, after importing a pref-file from a zip-archive, this can be edited with **Edit existing pref-file. Start run** will execute *Latte*.

The **Save-interval** spin box (in the **Misc settings** tab) allows to control how many pictures will be saved to disc. Output of the running simulation is displayed in the **Stdout** tab.

User-defined notes for a preference file will be displayed in the **Notes** tab, and can be edited there. They will be saved in the archive on finishing the interface.



Choose example/defaults: [✓ Boudins] Expanding inclusions Grooves Phase transition Grain growth Fluid flow Fractures Shrinkage Stylolithes Heat flow Diffusion

Name of process: Boudins

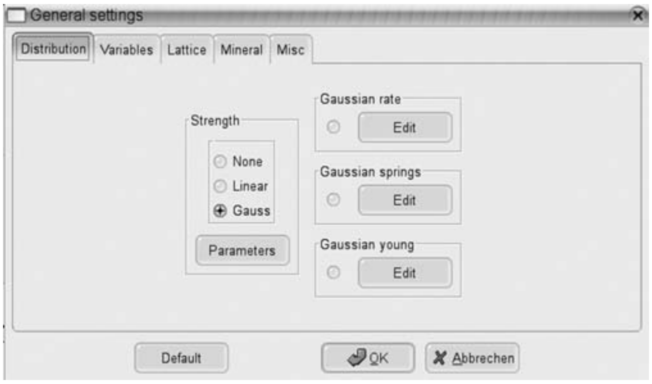
Current name: Boudins

Fig. G2. The 'New File'-window will set up a basic preference-file after choosing an example. The buttons will open other windows, which allow the refinement of the settings

G.2.5 The New file and the Edit file window

After having chosen to create a new preference-file, the dialogue on top of Fig. A16 will appear. It displays several predefined examples, which have been discussed in the book, and will give acceptable default-values. These settings, however, can be tweaked using the buttons on the right-hand side, which give control over the important functions and their parameters.

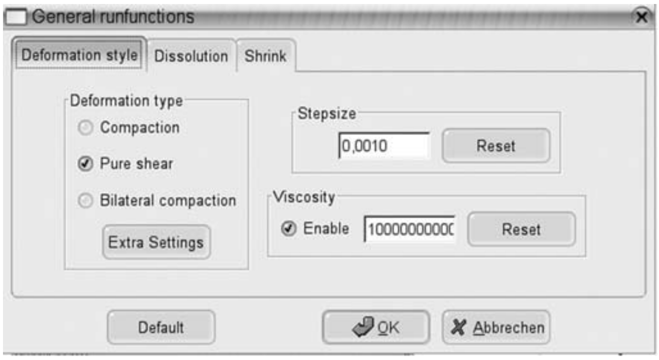
From this step on, the functionality of the **New file** and the **Edit file** window is identical. The edited variables will be saved, if the **OK** button is clicked in one of these windows.



Strength: None Linear [✓ Gauss]

Fig. G3. General, non-run-function-settings for a *Latte*-run

The **Edit settings** button will pop up a window, which allows the adjustment of basic properties of the lattice or the discrete elements. These settings range from basic distributions on several variables to settings concerning the number of the produced output-files.



Deformation type: Compaction [✓ Pure shear] Bilateral compaction

Stepsize: 0,0010

Viscosity: [✓ Enable] 1000000000C

Fig. G4. General run functions and the style of deformation

Similarly the run-functions dialogue covers functions, which are called during every single loop of the program and control the deformation behaviour. It consists of tabs for the deformation style, the dissolution and for shrinkage of either all particles of the system or single grains.

The **deformation style** tab gives control over the movement of the system-walls. **Compaction:** stands for unilateral compression, pressing the upper wall of the system down without moving the remaining walls. **Pure shear:** Pushes the top wall down and extends the systems on the right-wall-side. **Bilateral compaction:** Compresses the system by moving the top and bottom wall. Used in stylolite-simulations.

In addition it is possible to combine two deformation-types at a time or using different deformation-types at different time-steps using the **Extra settings** button. The user may also chose to have a visco-elastic lattice via a checkbox, and the viscosity can be entered in a text-field.

On the tab **Dissolution**, stylolite dissolution and dissolution strain can be applied. Please have a look for the corresponding function-descriptions and chapters. The same advice can be given for the **Shrink** tab, which allows the shrinkage of grains and/or the total system.

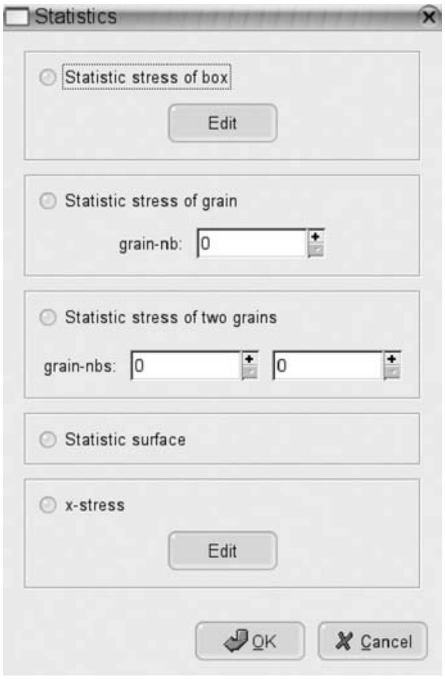


Fig. G5. The statistic options

G.2.6 The Statistics window

From the statistics window it is possible to enable the output of a file containing statistics. Note that these functions will be called every time the runtime-functions have been finished. See the list of functions in the appendix for details of the given functions.

G.2.7 Remaining tabs

The 'Info' tab: This includes a simple text control, where a description of the preference-file may be entered and saved. If importing a preference-file from a different zip-archive, this will be read and displayed.

The 'Misc settings' tab: The most important function of this tab, is the possibility to tell the interface where to look for the *Latte*-executable. If no distinct path to an executable is given, the current working directory will be the default, if the program finds it there. If there is still none, the program will have a look into the users 'PATH'-variable (on unix/linux), and, finally, it will try to find it two levels up in the file-hierarchy in the 'bin' directory. In addition, one can define in this tab after how many time steps an elle-file will be saved to the hard drive.

The 'Stdout' tab: Displays the standard-output of latte, which would be displayed on the command-line if started from there. (This doesn't display error-messages, if the program hangs! If you're developing and testing your own routines, don't start *Latte* from here, but use a shell). Unfortunately, the displayed output is not always synchronized with what really happens in latte (this seems to be an wxWidgets issue for which, however, the author didn't find a workaround). It is possible to save this output to a user-defined text-file.

The 'Make movie' tab: This is an attempt to provide a more or less convenient way of shooting movies from the *Elle* files produced by a run of latte. The usage is straightforward: choose the first and the last file of the sequence you want to shoot a film of, select a name for it and hit the start-button.

The interface-program relies on some external executables. Firstly it has to find *showelleps* on your system, which is in the bin/binx/'binwx directory within the *Elle* location. It is most advisable - not only for this particular purpose - to integrate this directory into your 'PATH'-variable anyway. Alternatively, the program will be looked up in the current working directory itself, and finally – if there is still no success – it tries to find the bin-directory relative to the working-directory (two levels up in the file hierarchy). To convert the generated postscript-files into an animated gif-file, the 'imagemagick'-package is needed – this should be found without problems, if installed on the system.

The 'Convert movie' tab: If you're not interested in the animated gif, which was created by **Make movie**, you can convert it into avi or mpeg, which will consume much less space on your hard drive and thus load much faster - and back to gif, if needed.

As usual an external program is needed, in this case *mencoder*, which is in turn part of the mplayer-package. If installed on your system, there shouldn't be any difficulties to use it from within the interface.

G.3 The command line interface for Windows and Linux developers

A switch command is used in the initialization and the run routine in the experiment class to define processes. When the experiment is called the -u command followed by an Integer calls the right process:

```
./elle_tte -i file.elle -u 1
```

for example, calls a simple fracture experiment.
The following processes are included in Latte as examples:

-u 1	fracturing of a granular aggregate,
-u 2	fracture boudinage of horizontal layers,
-u 3	fractures around expanding grains,
-u 4	shrinkage cracks,
-u 5	visco-elastic deformation and fracturing,
-u 6	dissolution grooves,
-u 7	stylolite roughening,
-u 8	combine grain growth and fracturing,
-u 9	solid-solid phase changes under stress,
-u 10	heat diffusion,
-u 11	grain growth,
-u 12	Lattice-Gas example diffusion,
-u 13	Lattice-Gas example 2d fluid flow.

Parameters in functions and basic functions for processes can be changed within the experiment class. The file experiment.cc starts with a number of header calls followed by a constructor (Experiment::Experiment()). After the constructor the function Experiment::Init() is used to initialize the different processes. Within this function the *Elle* file is called, desired lattices are build and initial geometrical configurations are applied to the data set (statistical distributions, layering etc.). The initialization function is called if the **Run** button in the *Elle* interface is chosen at the beginning and also if the **ReRun** button is chosen. Please note that a rebuilding of some lattices with the **ReRun** command may not be possible, especially if the microstructure is already deformed.

The run functions are then executed in another switch call in the function Experiment::Run().

In this function the user can similar to the init function change parameters in existing functions or add new functions for different processes. Basically all functions that are included in the lattice, phase_base, phase_lattice, min_trans_lattice and graingrowth classes can be called within the experiment class.

Appendix H Miscellaneous processes and utilities

Mark Jessell

H.1 Introduction

In addition to the processes already documented in this chapter, we have also developed a number of other processes that are briefly described here for the sake of completeness, and because they may act as template for those wishing to develop their own codes. We also describe the functionality of the utility codes, which are not meant to represent specific geological processes, but instead perform useful conversion functions.

H.2 *elle_gbdiff*

This code performs a Finite Difference grain boundary diffusion calculation for a uniform-width grain boundary network and constant diffusivity. The code is currently set to diffuse the bnode property CONC_A, but could be easily modified.

H.3 *elle_diff*

This code performs a Finite Difference lattice diffusion calculation, which ignores the grain boundary network, and assumes constant diffusivity. The code is currently set to diffuse the flynn property CONC_A, but could be easily modified. This code performs a similar function to the lattice diffusion part of the *elle_exchange* code, except that it is not limited to intragranular diffusion.

H.4 *elle_met*

This is a prototype code derived from the *elle_melt* code that attempts to simulate polyphase solid-state metamorphic reactions. It uses a standalone array of thermodynamic data derived, for example from the *Perplex* code (http://www.perplex.ethz.ch), to simulate reactions of the type A+B=C+D.

H.5 *elle_recovery*

This code performs a simple time-based recovery of the flynn DISLOCDEN attribute (dislocation density) that allows the gradual decay of this property.

H.6 *elle_viscosity*

This code assigns each flynn a viscosity as a function of its properties (e.g. grain size and/or dislocation density). This code is typically rewritten for each new type of experiment.

H.7 *elle_expand* & *elle_pblast*

These two codes force any grains with their flynn attribute EXPAND set to 1 to slowly grow, consuming their matrix.

H.8 *elle_manuel*

This code applies a homogeneous simple shear to the entire microstructure.

H.9 *reposition*

This utility code transforms the deformed *Elle* microstructure from a Finite Element or *elle_manuel* calculation so that all nodes lie within the original bounding box. This is possible because *Elle* models have cyclic boundaries.

H.10 *tidy*

This utility code transforms the *Elle* microstructure in a number of different ways, including changing the bnode spacing, adding or removing unodes, randomising Euler angles of all grains etc. To modify an Elle file, you can use the tidy utility with the following user data settings, accessible from the Run Options Window:

- **User Data 1** Change default node spacing. Input value wil be the new switch distance. Setting the value to 0 keeps the current switch distance.
- **User Data 2** Set to anything but 0 to randomise the Euler angles.
- **User Data 3** If the value set here is equal to F_ATTRIB_A of a Flynn, this Flynn will get the mineral attribute "MICA". All other Flynns will be assigned the mineral attribute "QUARTZ". Set to zero to not use this function.
- **User Data 4** Setting the input to anything other than zero reorients the C-axis (Euler angle) of "MICA"-Flynns to become parallel to the long axis of the Flynn.
- **User Data 5** Add or delete unodes from file. Any valy smaller than zero will delete all unodes, while values bigger than zero determine the number of unodes.
- **User Data 6** Set the unode pattern: 0 = hexagonal array; 1 = square array; 2 = random; 3=semi_random.
- **User Data 7** If "User Data 6" was set to 3 (semi random), it defines the number of unodes per cell, otherwise set to 0.

H.11 *elle2poly*, *basil2elle*, *elle2oof*, *goof2elle*

These utility codes transforms the *Elle* microstructure into a format that the Finite Element codes *Basil* (*elle2poly*) and *OOF* (*elle2oof*) can understand, and similarly transforms the outputs from *Basil* (*basil2elle*) and *OOF* (*goof2elle*) back into the *Elle* format.

H.12 *ppm2elle*

This utility code takes a binary format *ppm* format raster image and transforms it into an *Elle* format file. For this to work the microstructure in the *ppm* file must consist of grains with completely uniform colours, with no lines demarking grain boundaries, and island grains completely within other grains are not allowed.

H.13 *ebsd2elle*, *elle2ebsd*

This utility code *ebsd2elle* takes the Euler angle information that describes the crystallographic orientations in an HKL format EBSD image and maps it onto the unodes of an *Elle* file. The grain boundary network information needs to be transferred as a separate step using *ppm2elle. elle2ebsd* maps the unode Euler angle information onto a regular grid and writes a file in HKL-format.

H.14 *plotaxes*

The utility code creates an equal angle upper hemisphere postscript plot of the c- a- and r- axes of the quartz grains in an *Elle* file.

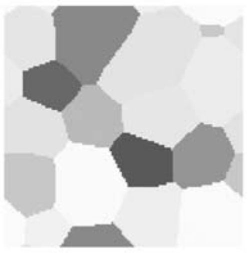
Appendix I How to Create an *Elle* file

Mark W. Jessell

We can create an arbitrary geometry *Elle* input file from any rectangular binary raster image. Any rectangular bounding box will be mapped to a square, so generally a square image is a good place to start, with dimensions of around 100x100 pixels. Each patch of pixels with a unique colour will be considered to be a single grain, so in order to minimise the number of individual grains, you should limit the number of different colours that appear in the image. Do not draw the grain boundaries, only the grains. In this example we use the Open Source drawing package *gimp* (http://www.gimp.org) to create the image, but any raster or vector package that can output a raster image (ppm-format) could also be used. We assume that there is a pre-existing raster that you want to modify, but you could equally start from a blank screen.

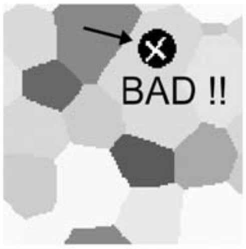
The example we use was actually created by saving out an existing *Elle* microstructure as a graphics file, with no grain boundaries displayed (this is important as otherwise the grain boundaries would be considered to be a different grain).

1. Start up *gimp* by typing in **gimp** (Linux) or click on the appropriate icon (Windows).
2. Select **Open** from the **File** menu and load the image foam.ppm (found in the ...elle/extras/ppm_files directory).

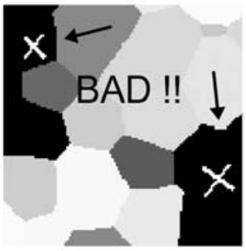


3. Use the **Pencil** tool to modify your microstructure, using a different colour for each neighbouring grain. Grains can wrap around horizontal and vertical boundaries, but:

Don't make island grains (i.e. grains completely surrounded other grains)



Don't make any single grain larger than about 30% of the length of the box (be careful with the wrapping resulting in grains being bigger than they appear).



4. Save the image by right-clicking in the image window and select **Save as**, and call you file any_name_you_like.ppm.
5. Quit from *gimp*.
6. Open up the *Experiment Launcher* and select and run *ppm2elle.*
7. Load up the ppm-format file you just created, which will be converted as it is loaded and then save the file as a new *Elle*-format file.
8. Next, go back to the *Experiment Launcher* and select and run **tidy**. Load the new Elle file, and go to **Run→Run Options** and change "Stages" to 1 and the **UserData field 1** to 0.01, and change all the other **Userdata** values to 0.0. Now select **Run→Run**. This has changed the bnode spacing to 0.01 *Elle* units.
9. Save out the new *Elle* file.
10. When the normal Elle window appears, open the *Elle* file you created in the previous steps and then go to **Run→Run Options** and change "Stages" to 100. Finally select **Run→Run** and you will see the new microstructure evolve through 100 time steps.

You will now have an *Elle* file that describes the geometry and topology of your microstructure, but will not have any specific physical or chemical attributes set as yet. See Appendix J to see how to add attributes with a text editor. Before you use the new data file for a simulation, it is often advisable to run ten or more steps of grain growth (step 10). This way you can see that the microstructure has no hidden problems and unwanted small (one or more pixel) grains are removed.

Appendix J The *Elle* file format

Mark W. Jessell

The *Elle* File is a text file that describes the general experimental conditions, followed by a description of the geometric, physical and chemical characteristics of the specimen, i.e. its microstructure. The file can be modified with any simple text editor, or even a word processor, as long as it is saved out again as a text file. If you have created a new microstructure using *ppm2elle*, you may add lists of new flynn, bnode or unode attributes at the end of the file (the order of the parameter blocks is not important). Often you will simply want to set default values, perhaps defining a few elements as having non-default values.

J.1 Example *Elle* file

```
# Created by test_unodes: elle version 2.3.2  Fri Jun  8 13:38:25
2001
# LINES THAT START WITH A # ARE COMMENTS

# THE OPTIONS BLOCK DEFINES GLOBAL PARAMETERS

OPTIONS
SwitchDistance 2.50000000e-02
MaxNodeSeparation 5.50000000e-02
MinNodeSeparation 2.50000000e-02
SpeedUp 1.0
CellBoundingBox 0.00000000e+00 0.00000000e+00
                1.00000000e+00 0.00000000e+00
                1.00000000e+00 1.00000000e+00
                0.00000000e+00 1.00000000e+00
SimpleShearOffset 0.00000000e+00
CumulativeSimpleShear 0.00000000e+00
Timestep 3.15000000e+07
UnitLength 1.00000000e-02
Temperature 2.50000000e+01
Pressure 1.00000000e+00
MassIncrement 0.00000000e+00

#THE NEXT BLOCK DEFINES THE POLYGONAL STRUCTURES: THE FIRST NUMBER IS
THE ID <E.G. 3>, THE SECOND THE NUMBER OF NODES THAT DEFINE THE
POLYGON <E.G. 23>, ANS FOLLOWED BY AN ANTICLOCKWISE LIST OF NODE IDS
E.G. <133 272 140 ETC.>

FLYNNS
3 23 133 272 140 146 112 174 87 15 147 150 211 129 17 39 62 94 119 20
13 8 72 135 7
7 36 21 76 31 29 73 107 279 302 166 280 167 30 224 156 209 27 131 25
16 77 132 2 0 141 4 121 1 10 86 122 155 227 22 3 37 105
```

```
11 24 20 119 94 62 39 17 33 165 34 48 49 162 51 50 88 36 38 56 78 29
31 76 21 142
#etc.

#THE NEXT BLOCKS DEFINE FLYNN PAAMETERS SO TAHT FOR EXAMPLE, AN
EXPAND PARAMETER IS DEFINED FOR THESE FLYNNS, THE DEFAULT VALUE IS 1
AND THE FLYNN WITH ID 17 IS SET TO 1

EXPAND
Default 1
17 0
DISLOCDEN
Default 0.00000000e+00
MINERAL
Default QUARTZ
EULER_3
Default 0.00000000e+00 0.00000000e+00 0.00000000e+00

#THE LOCATION BLOCK DEFINES THE LOCATION OF BOUNDARY NODES. THE FIRST
NUMBER IS THE BNODE ID, THE SECOND IS X, THE THIRD IS Y

LOCATION
0 0.7157127300 0.0004124800
1 0.5647232500 0.9642747600
2 0.7517333600 0.9709875600
3 0.4205657800 0.7458517000
4 0.6626872400 0.9829956900
272 0.1584355500 0.7560001600
279 0.5814173200 0.6308649800
280 0.6592591400 0.6547707300
284 0.5687853700 0.4511538400
289 0.1280806400 0.3809446100
293 0.2484257500 0.1165237400
302 0.6079290500 0.6402233800
#etc.

#NEXT COMES BNODE PROPERTIES
CONC_A
Default 0.00000000e+00
29 4.0

#THE NEXT BLOCK DEFINES THE POSITION OF UNCONNECTED NODES. THE FIRST
NUMBER IS THE UNODE ID, THE SECOND IS X, THE THIRD IS Y

UNODES
0 0.000000 0.008660
1 0.020000 0.008660
2 0.040000 0.008660
3 0.060000 0.008660
4 0.080000 0.008660
5 0.100000 0.008660
6 0.120000 0.008660
#etc.

#NEXT COMES UNODE PROPERTIES
U_CONC_A
Default 1
0 1.2
```

J.2 *Elle* file blocks

The file format consists of a series of 7 sets of data blocks:

The **OPTIONS Block** - This defines the general experimental conditions:

```
SwitchDistance The distance between triple nodes before they are
      switched
MaxNodeSeparation The maximum distance between double nodes before a
      new node is inserted
MinNodeSeparation The minimum distance between double nodes before
      they are merged
SpeedUp Arbitrary rate multiplyer for some processes
CellBoundingBox Bottom Left, Bottom Right, Top Right and Top Left
      coordinates of bounding Box
SimpleShearOffset Used to determine the Bounding box shape for simple
      shear deformation experiments
CumulativeSimpleShear Used to determine the Bounding box shape for
      simple shear deformation experiments
Timestep Single timestep in seconds
UnitLength Real World width of unit bounding box
Temperature Temperature in Kelvin
Pressure Pressure
MassIncrement Used to add mass to grian boundaries for validation
      tests
```

The **FLYNNS Block** - The next block defines the polygonal grain structures: the first number is the id <e.g. 3>, the second the number of nodes that define the polygon <e.g. 23>, followed by an anticlockwise list of node ids e.g. <133 272 140 etc.> All flynns will be defined here. The bnode ids refer to the ids found in the Bnode Block.

```
FLYNNS
3 23 133 272 140 146 112 174 87 15 147 150 211 129 17 39 62 94 119 20
13 8 72 135 7
7 36 21 76 31 29 73 107 279 302 166 280 167 30 224 156 209 27 131 25
16 77 132 2 0 141 4 121 1 10 86 122 155 227 22 3 37 105
11 24 20 119 94 62 39 17 33 165 34 48 49 162 51 50 88 36 38 56 78 29
31 76 21 142
17 25 50 123 110 204 164 158 64 53 67 115 128 85 68 24 91 180 40 23
185 182 284 96 38 36 88
#etc.
#etc.
#etc.
```

The **FLYNN Parameters Block(s)** - The next set of blocks define flynn parameters so that, for example, an EXPAND parameter is defined for these flynns, the default value is 1 and the flynn with id 17 is set to 0. You can only define parameters that the *Elle* system recognises, however three dummy parameters are provided.

Each defined parameter consists of its name, followed by a Default value definition, followed by the list of flynns that do not have the default value and their values. All FYLNN parameters are optional, and all definitions apart from the default value are optional, if a parameter is defined as being present. In the example below all flynns would have a value for EXPAND of 1, expect flynn number 17, which has a value of 0.

```
EXPAND
Default 1
17 0
DISLOCDEN
Default 0.00000000e+00
3 21.9
17 23.32
MINERAL
Default QUARTZ
EULER_3
Default 0.00000000e+00 0.00000000e+00 0.00000000e+00
```

The current list of valid flynn parameters is:

F_ATTRIB_I, F_ATTRIB_J, F_ATTRIB_K (dummy integer attributes to be assigned by the user)
MINERAL (Currently, valid keywords are QUARTZ, FELDSPAR, MICA, GARNET, CALCITE, MINERAL_A, MINERAL_B, MINERAL_C)
EXPAND (toggles grain expansion for elle_pblast and elle_expand)
COLOUR (this attribute will become redundant, a number in the range 8-63)
ENERGY (Arbitrary volume energy term)
VISCOSITY (Viscosity term)
S_EXPONENT (Stress exponent for power-law viscosity)
DISLOCDEN (dislocation density)
F_ATTRIB_A, F_ATTRIB_B, F_ATTRIB_C (dummy floating point attributes to be assigned by the user)
CAXIS (polar coordinates of c-axis orientation)
EULER_3 (Euler space coordinates of lattice orientations)
FLYNN_STRAIN (current strain state as calculated by *Basil* [followed by at least one of E_XX, E_XY, E_YY, E_YX, E_ZZ, F_INCR_S, F_BULK_S])

The **BNODES Block** - The location block defines the location of boundary nodes. The first number it the bnode id, the second is *x*, the third is *y* coordinate.

```
LOCATION
0 0.7157127300 0.0004124800
1 0.5647232500 0.9642747600
2 0.7517333600 0.9709875600
3 0.4205657800 0.7458517000
4 0.6626872400 0.9829956900
5 0.0854129600 0.0134190800
255 0.1195778800 0.1301631600
272 0.1584355500 0.7560001600
279 0.5814173200 0.6308649800
280 0.6592591400 0.6547707300
284 0.5687853700 0.4511538400
289 0.1280806400 0.3809446100
293 0.2484257500 0.1165237400
302 0.6079290500 0.6402233800
#etc.
#etc.
#etc.
```

The **BNODES Parameters Block(s)** - The same logic applies as for flynn parameters.

```
CONC_A
Default 0.00000000e+00
29 4.0
```

The current list of valid bnode parameters is:

N_ATTRIB_A, N_ATTRIB_B, N_ATTRIB_C (user assigned node attributes)
VELOCITY VEL_X VEL_Y (the velocities in the *x* and *y* directions)
STRESS The local stress state as calculated using *Basil*. [TAU_XX TAU_YY TAU_ZZ TAU_XY TAU_1 PRESSURE] (at least one)
CONC_A (a dummy concentration attribute)
STRAIN (the local strain state as calculated using *Basil* [INCR_S BULK_S]; at least one)

The **UNODES Block** - The next block defines the position of unconnected nodes. The first number it the unode id, the second is *x*, the third is *y* coordinate. This block and the unodes parameter blocks are optional.

```
UNODES
0 0.000000 0.008660
1 0.020000 0.008660
2 0.040000 0.008660
3 0.060000 0.008660
4 0.080000 0.008660
5 0.100000 0.008660
6 0.120000 0.008660
7 0.140000 0.008660
#etc.
#etc.
#etc.
```

The **UNODES Parameters Block(s)** - The same logic applies as for flynn and bnode parameters.

```
U_CONC_A
Default 1
0 1.2
```

The current list of valid unode parameters is:

U_CONC_A (Attribute used to store chemical concentration)
U_ATTRIB_A, U_ATTRIB_B, U_ATTRIB_C (user assigned unode attributes)
U_STRAIN START_S_X START_S_Y PREV_S_X PREV_S_Y CURR_S_X CURR_S_Y (the *x*,*y* coordinates for the starting position, the position before the last increment of strain and the current position)
U_CAXIS (Polar coordinates of c-axis orientation)
U_EULER_3 (Euler space coordinates of lattice orientations)
U_ENERGY (Arbitrary volume energy term)
U_VISCOSITY (Viscosity term)
U_DISLOCDEN (dislocation density)

Index

Page numbers referring to the entries discussed in detail are given in **BOLD.**

Accuracy, 11, 37, 79, 121–123, 143, 177
ATG-instability, 187, 189, 191

Base libraries, 77, 78, 83
Basil, 78, 81, 84, **139–154**, 158, 159, 216, 226, 227, 251, 258–261, **294**, **295**, **314–317**, 318–320, 330, 333–335, 337, 347, 362, 363, 369, 371–377, 391, 398, 399
Binaries, 77, 78, 84, 294, 295, 391
Boudinage, **179, 180,** 241, 244, 245, **321–323**, 343, 344, 386
Boundary energy, 26, 110, 115, 121, 151, 209–214, 260, 311, 331
  anisotropy, 5, 13, 36, 66, 101, 102, 133, 140, 156, 159, 175, 209–214, 225
Boundary models, 16, **33–36**, 183, 224
Boundary node (bnode), 34, 78, **79, 80**, 91, 94, 118, 119, 149–152, 351, 353–359, 389, 391, 394–400
Brittle deformation, 10, 172–175, 197, **241**, 338–345

Cation exchange, **87–96**, 195, **307–311**
Cellular automata, 16, 21, 27–29, 81, 100, 110, 117, 175, 176, 187
CIP, 155
Cleavage, 232, 233
CPO, lattice-preferred orientation, 5–7
Critical resolved shear stress (CRSS), 156, 158, 159, 318–320
Crystal growth, 4, 7, 8, 54, 55, 127, 151, 198–206, **302, 303**
Crystallographic preferred orientation, 1, 4–7, 127, 128, 140, 155, 235

Data file, 83, 84, 149, 151, 363, 394, **395–400**
Data structure, **15–18**, 75, 77–84, 91–94, 149, 199
Deformation Mechanism map, 3
Delaunay triangulation, 16, 17, 81, **82**, 151, 301, 356, 369
DiffForm, 85, **166, 167**
DiffForm programs, 166, 167
Diffusion, 11, 20, 30, 33, 53, 87, 89–96, 116, 134, 161–169, 176, 185, 195–198, 202, **237–240**, **299, 300**, 307–309, 363, 369, 379, 386, 389
Diffusion coefficient (diffusivity), 93, 94, 164, 202, 204, 309
Diffusion creep, 59, 161–169
DiffView, 167
Dislocation creep, 2
Dislocation density, 18, 79, 99, 108–113, 116, 117, 158, **217–224**, 264, 333, 390, 398, 400
Dislocation dynamics, 57–63
Dislocation energy, 217
Dislocation glide, 155–157, 217
Dislocations, 6, 11, 12, 57–63, 97–99, 106–113, 156, 213
Dissolution, 233, 236, 237, 323–327, 383–386
Dissolution grooves, 184–193, 210, **323, 324**, 386
Dissolution-precipitation, 6, 7, 140, 156, 161–169, 183–193, 213

EBSD, 3, 98, 101, 310, 392
Elasticity (elastic), 46, 51, 60–73, 81, 99, 140, 174–181, 185–193, 197–201, 233, 237, 241–245, 306, 324–327, 342–344, 384, 386
Elle base libraries, 77, 78, 83
Elle binaries, 77, 78, 84, 294, 295, 391
Elle (data) file, 83, 84, 149, 151, 363, 394, **395–400**
Elle data structure, 15–18, 75, 77–84, 91–94, 149, 199
Elle experiment script, 84, 85, 178, 361–363, **366–371**
Elle_latte Programs
  elle_manuel, 363, **390**
  elle_melt, 135, 389
  elle_met, **389**
  elle_nucl_xx, **218**, 335
  elle_pblast, 259, 260, 330, 363, **390**, 398
  elle_recovery, 219, 335, 390
  elle_split, **112**, **217**, 335, 337
  elle_tbh, 140, 158, **217**, 319, 320, 335
  elle_viscosity, 158, **219**, 318–320, 335, 337, **390**
Elle processes
  elle_angle_rx, 111, **218**, 335
  elle_diff, **389**
  elle_exchange, 92, 307, 389
  elle_expand, 149, 150, 316, **390**, 398
  elle_gbdiff, 92, 94, 307, 363, **389**
  elle_gbm, 125, **219**, 331, 335
  elle_gg, 125, **259**, 260, 330, 337, 362
Elle utilities
  basil2elle, 149, 153, 316, 391
  elle2oof, **391**
  elle2poly, 149, 150, 153, 316, **391**
Error, 37, 41, 42, 122
Euler angles, 159, 160, 309, 319, 391, 392, 398, 400
Experiment script Elle terms, 84, 85
Experiment Launcher, 294, **297, 298**, **361**, 371, 394
  ebsd2elle, 392
  goof2elle, 391
  plotaxes, **392**
  ppm2elle, **391**, 392, 394, 395
  reposition, 149, 153
  showelle, 297, 298, 351, 361–363, 370
  showelleps, 385
  tidy, 391, 394
  triangle, 81, 147, 151, 152

FACET, 7, 8, 130
Failure, 10, 172–175, 197, 241, 338–345
Finite Difference (FD), 31, 34, **37–49**, 55, 90–94, 119, 199, 201, 202, 389
  Alternating Direct Implicit, 201
  Crank-Nicholson, 41, 46, **47**
  explicit, 31, **39–44**, 45–47, 202
  implicit, 41, **45, 46**, 47, 202
Finite Element (FEM), 6, 16, 27, 34, **50–53**, 55, 64, 78, 81, 118, 140, 143, 151, 152, 155–160, 175, 187, 199, 216, 225, 226, 250, 257–259, 293, 294, 347, 371–373, 390, 391
Flow law, 4, 18, 52, 53, 164, 227, 337
Fluid flow, 30, **300–302**, 386
Flynn, **78–85,** 135, 136, 159, 176–178, 217, 351–359, 369, 379, 389–391, 395–400
Foam texture, 1, 33, 34, 115, 123, 211, 226, 311, 318–320
Fracture, 10, 171–175, 197, 241, 338–345
Fringe, 7, 8, 10, 128, 131
Fringe Growth, 7, 85, 127, 130
Front-Tracking, 27, 33, 34, 54, 59–62, 110, 128, 118, 127

Gaussian distribution, 30, 178–180, 190–193, 234, 235, 244, 305, 306, 322, 324, 327

Grain boundary
  diffusion, 59, **87–96**, 307–309, 369, **389**
  energy, 26, 110, 115, 121, 151, 209–214, 260, 311, 331
  migration, 5, **97–103**, 109, **115–126**, 155, 196, 198, 200, 209–214, **219**, 226, 311, 330, 332–335, 337
  mobility, 6, 100, 115–126, 210, 213, 220–223, 259, 310, 333–335
  node (bnode), 34, 78, **79, 80**, 91, 94, 118, 119, 149–152, 351, 353–359, 389, 391, 394–400
  sliding, 5, 59, 140
  width, 79
Grain growth, 13, 25, 26, **34–36**, 115, 117, **123–125**, 133–137, 151, 198–200, **209–214**, 227, 257, 259
Griffith (theory), 71, 174
Grooves, 183–193, 210, **323, 324**, 386

Heat conduction, 40
Heat of crystallisation, 40, 303
Hydrofracture, 173

Ice, 5
Inclusion (in matrix), 8, 147, 149, **249–253**, **338, 339**, 346, 347
Installing Elle, **293–295**
Installing experiments, 297
Ising model, 20, **21–25**, 28

Lagrange (Lagrangian), 149, 175, 250, 346, 373
Latte (elle_latte), 176, 177, 187, 199, 200–203, 232, 237, 242, 299, 301, **379–387**
Lattice Boltzmann/-gas, **28–32,** 299–302, 386
Lattice preferred orientation (LPO), 1, 4–7, 127, 128, 140, 155, 235
Lattice rotation, 5–7, 107, 155, **156–160**, 215, 216, 317–320, 332, 335
Lattice spring, 16, **64–73**, 176, 187, 304, 321, 323
Libclip, 83
Libelle, 83
Libmatrix, 83
Libplot_wx, 83
Libplot_x, 83
Libps, 83
Localisation (deformation), 4, 9, 12, 60, **226–231**, **247–253**, 328–330, **335–338**, 345–347

Matlab, 167
Melt(ing), 54–56, **133–137**, 213, **302, 303**, **313, 314**, 389
MgO, 210
Mica, 11, 159, 308, 328, 345, 391, 398
Minerals
  albite, 106
  andalusite, 256
  aragonite, 196
  biotite, 1, **87–96**, 248, 307, 308
  calcite, 5, 106, 196, 398
  garnet, **87–96**, 307, 308, 315, 346, 398
Misorientation, 97, 98, 101, 102, 106, 111, 218, 219, 222, 310, 311, 369
Mobility (grain boundary), 6, 100, 115–126, 210, 213, 220–223, 259, 310, 332–335
Molecular dynamics, **57–63**, 64
Monte Carlo, 5, 6, **20–32**, 35, 47, 100, 110, 199
Mud cracks, 231, **340, 341**

Normal force model, 64, **65, 66**, 68, 175, 176
Nucleation (grain), **105–113**, 173, 195, 197–199, 200, 332–335, 369

OIM (Orientation Imaging Microscopy), 3
Olivine, 106, 134, 195–205, 238–240
OOF, 81, 369, 391
Oolites, 161, 162

Partial melt, 54–56, **133–137**, 213, **302, 303**, **313, 314**, 389
Peach-Koehler formula, 60, 61
Perplex, 13, 390
Phase field (model), 16, **54–56**, 59, 61, 62, 100, 110, 117, 187, 200, **302, 303**
Phase transition, 14, 58, 59, **195–206**, **237–240**, 379
Porphyroblast, 4, **8, 9**, 140, **147–153**, 159, 248–250, **314–316**, **327–330**, **345, 346**, 369
Porphyroclast, 4, **8, 9**, **147–153**, 241
Potts models, 16, 20, **25–29**, 33, 54, 81, **97–103**, 110, 117, 224, 309–311
Preferred orientation (lattice), 1, 4–7, 127, 128, 140, 155, 235
Pressure solution, 149, 161, 162, 183
Programs
  Basil, 78, 81, 84, **139–154**, 158, 159, 216, 225, 226, 250, 258–260, 293–295, **314–317**, 318–320, 330, 333–335, 337, 347, 362, 363, 369, 371–377, 391, 398, 399
  DiffForm, 85, **166, 167**
  DiffView, 167
  FACET, 7, 8, 130
  Fringe Growth, 7, 85, 127, 130
  Latte (elle_latte), 176, 177, 187, 199, 200–203, 232, 237, 242, 299, 301, **379–387**
  Matlab, 167
  OOF, 81, 369, 391
  perplex, 13, 390
  sybil, 148, 293–295, 362, 363, **371–377**
  Thermocalc, 13
  Vein Growth, 7, 8, 85, **127–131**
Pyrite, 8, 162

Quartz (quartzite), 1, 2, 5, 6, 11, 91, 94, 106, 112, 113, 116, 122, 127, 133, 134, 150, 155, 159, 161, 162, 186, 195–197, 218, 232, 248, 256, 308, 318, 320, 324, 327, 369, 391, 392, 398

Reactions, 7, 13, 30, **87–96**, 119, **183–193**, 195–206, **231–235**, 237–239, 255, 264–266, **307–309**, 379, 389
Region of Interest (ROI), 82, 83

Salt, 98, 106, 184, 191, 209–214, 311
Slip (system), 5, 60–62, 72, 156–159, 217, **318–320**
Spinel, **195–205,** 237–239
Staurolite, 248
Strain-rate partitioning, 4, 9, 12, 60, **255–265**, **327–330**, 335–338, 345–347
Stylolite, **183–193,** 232, 235, 236**,** **325–327**, 384, 386
Subgrain, 12, 64, 78, 81, 82, **97–113**, 117, 158, 215–219, 224, **309–311**, 332–335, 369
  boundary, 97–100, 109
  formation, 5, **105–113**, 155, 215–217, 224
Surface energy, 26, 110, 115, 121, 151, 209–214, 259, 311, 331
Sybil, 148, 293–295, 362, 363, **371–377**

Taylor-Bishop-Hill (TBH), 5–7, **155–160**, 217, 318
Texture, 1–6, 34, 88, 91, 115, 123, 211, 226, 311, 318, 331
Thermocalc, 13
Topological check/problems, 35, 79, 80, 316, 352
Triangulation, 16, 17, 79–83, 151, 301, 356, 369

Unconnected node (unode), **78–83,** 85, 91–94, 100, 149, 150, 199, 351–359, 391, 392, 395–400

Vacancy, 20, 162–164
Vein, 4, **7, 8**, 116, **127–131**, 172, 173, 231, 232
Vein Growth, 7, 8, 85, **127–131**
Vertex model, 118
Visco-elastic, 70, 81, **241–245**, **342–345**, 384, 386
Voronoi, 17, 18, 81–83

Printing: Krips bv, Meppel, The Netherlands
Binding: Stürtz, Würzburg, Germany