Christophe Tricaud
YangQuan Chen

Optimal Mobile Sensing and Actuation Policies in Cyber-physical Systems

Springer

Optimal Mobile Sensing and Actuation Policies in Cyber-physical Systems

Christophe Tricaud • YangQuan Chen

Optimal Mobile Sensing and Actuation Policies in Cyber-physical Systems

Springer

Dr. Christophe Tricaud
Cummins Inc.
1900 McKinley Avenue
Columbus 47201, IN
USA
ctricaud@ieee.org

Dr. YangQuan Chen
Department of Electrical and Computer Engineering, CSOIS
Utah State University
Old Main Hill 4160
Logan 84322-4120, UT
USA
yqchen@ieee.org

ISBN 978-1-4471-2261-6 e-ISBN 978-1-4471-2262-3
DOI 10.1007/978-1-4471-2262-3
Springer London Dordrecht Heidelberg New York

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library

Library of Congress Control Number: 2011940676

Cover design: VTeX UAB, Lithuania

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

To our colleagues, friends, and families

Preface

Cyber-physical systems (CPSs) are an emerging research topic born from the ever increasing complexity of engineered systems. Future systems will have to interact with each other and with the physical world in a very tight and well-coordinated fashion, and designing such systems is the research challenge behind CPSs. CPSs have been defined as "computational thinking and integration of computation around the physical dynamic systems where sensing, decision, actuation, computation, networking, and physical processes are mixed". Given such a definition of CPSs, it is trivial to observe that there are two main entities in a CPS: the "cyber" end of the system that is composed of the hardware and software, and the "physical" end of the system that relates to part of the environment. The problem of designing the cyber part may not be trivial but can be solved from scratch. However, the physical part, usually a natural physical process, is inherently given and has to be identified in order to propose an appropriate cyber part to be adopted. Therefore, one of the first steps in designing a CPS is to identify its physical part. The physical part can belong to a large array of system classes. Among the possible candidates, we focus our interest on distributed parameter systems (DPSs) whose dynamics can be modeled by partial differential equations (PDEs). DPSs are by nature very challenging to observe as their states are distributed throughout the spatial domain of interest. Therefore, systematic approaches have to be developed to obtain the optimal locations of sensors to optimally estimate the parameters of a given DPS.

With this monograph, we wish to provide our reader with a comprehensive understanding of CPS and emphasize on our past experience in the topic. For the past five years, we have worked on the topic of optimal mobile sensing and actuation policies in CPSs and directed our research effort from purely theoretical results to more applicable results. That is why the reader will find not only sensing and actuation, but also remote sensing, an online solution to the problem of optimal sensing, and communication topologies.

We first review the recent methods from the literature as the foundations of our contributions. Then, we define new research problems within the above optimal parameter estimation framework. Two different yet important problems considered are the optimal mobile sensor trajectory planning and the accuracy effects and allocation of heterogeneous sensors. Under the remote sensing setting, we are able to determine the optimal trajectories of remote sensors. The problem of optimal robust estimation is then introduced and solved using an interlaced "online" or "real-time" scheme. Actuation policies are introduced into the framework to improve the estimation by providing the best stimulation of the DPS for optimal parameter identification, where trajectories of both sensors and actuators are optimized simultaneously. We also introduced a new methodology to solving fractional-order optimal control problems, with which we demonstrate that we can solve optimal sensing policy problems when sensors move in complex media, displaying fractional dynamics. We consider and solve the problem of optimal scale reconciliation using satellite imagery, ground measurements, and unmanned aerial vehicles (UAVs)-based personal remote sensing.

Finally, to provide the reader with all the necessary background, the appendices contain important concepts and theorems from the literature as well as the MATLAB codes used to numerically solve some of the described problems.

Columbus, IN, USA Christophe Tricaud
Logan, UT, USA YangQuan Chen

Acknowledgements

With this monograph, our intent is to provide a comprehensive summary of our research efforts during the past few years in the domain of optimal mobile sensing and actuation policies in cyber-physical systems. For that reason, this monograph contains materials from papers and articles that were previously published as well as the Ph.D. dissertation of the first author. We are thankful and would like to acknowledge the copyright permissions from the following publishers who have released our work on that topic:

Acknowledgement is given to the Institute of Electrical and Electronic Engineers (IEEE) to reproduce material from the following papers:

©2008 IEEE. Reprinted, with permission, from Christophe Tricaud, Maciej Patan, Dariusz Uciński, and YangQuan Chen. "D-Optimal Trajectory Design of Heterogeneous Mobile Sensors for Parameter Estimation of Distributed Systems", *Proc. of the 2008 American Control Conference*, Seattle, Washington, USA, June 11–13, 2008, pp. 663–668 (material found in Chap. 3). DOI: 10.1109/ACC.2008.4586568

©2009 IEEE. Reprinted, with permission, from Christophe Tricaud and YangQuan Chen. "Solution of Fractional Order Optimal Control Problems Using SVD-based Rational Approximations", *Proc. of the 2009 American Control Conference*, St. Louis, Missouri, USA, June 10–12, 2009, (material found in Chap. 6)

©2009 IEEE. Reprinted, with permission, from Christophe Tricaud and YangQuan Chen. "Optimal Mobile Actuator/Sensor Network Motion Strategy for Parameter Estimation in a Class of Cyber Physical Systems", *Proc. of the 2009 American Control Conference*, St. Louis, Missouri, USA, June 10–12, 2009 (material found in Chap. 4).

©2010 IEEE. Reprinted, with permission, from Christophe Tricaud and YangQuan Chen. "Optimal Trajectories of Mobile Remote Sensors for Parameter Estimation in Distributed Cyber-physical Systems", *Proc. of the 2010 American*

Control Conference, Baltimore, Maryland, USA, June 30–July 2, 2010 (material found in Chap. 7).

©2010 IEEE. Reprinted, with permission, from Christophe Tricaud and YangQuan Chen. “D-Optimal Trajectories of Mobile Sensors with Fractional Dynamics for Parameter Estimation of Distributed Parameter Systems”, *Proc. of the 8th World Congress on Intelligent Control and Automation*, Jinan, Shandong, P.R. China, July 7–9, 2010 (material found in Chap. 7).

©2010 IEEE. Reprinted, with permission, from Christophe Tricaud and YangQuan Chen. “Smart Remote Sensing of Environmental Systems Using Unmanned Air Vehicles”, *Proc. of the 8th World Congress on Intelligent Control and Automation*, Jinan, Shandong, P.R. China, July 7–9, 2010 (material found in Chap. 5).

Acknowledgement is given to the Society for Industrial and Applied Mathematics (SIAM) to reproduce material from the following papers:

©2008 SIAM. Reprinted, with permission, from Christophe Tricaud and YangQuan Chen. “Optimal Mobile Sensing Policy for Parameter Estimation of Distributed Parameter Systems: Finite Horizon Closed-loop Solution”, *Proc. of the 2008 SIAM Eighteenth International Symposium on Mathematical Theory of Networks and Systems*, Blacksburg, Virginia, USA, July 28–August 1, 2008 (material found in Chap. 5).

©2008 SIAM. Reprinted, with permission, from Christophe Tricaud and YangQuan Chen. “Optimal Mobile Actuation Policy for Parameter Estimation of Distributed Parameter Systems”, *Proc. of the 2008 SIAM Eighteenth International Symposium on Mathematical Theory of Networks and Systems*, Blacksburg, Virginia, USA, July 28–August 1, 2008 (material found in Chap. 6).

Acknowledgement is given to the International Federation of Automatic Control (IFAC) to reproduce material from the following paper:

©2008 IFAC. Reprinted, with permission, from Christophe Tricaud and YangQuan Chen. “Solving Fractional Order Optimal Control Problems in RIOTS 95—A General-Purpose Optimal Control Problems Solver”, *Proc. of the 2008 Conference on Fractional Differentiation and its Applications*, Ankara, Turkey, November 5–7, 2008 (material found in Chap. 7).

The research described in this monograph would not have been possible without the inspiration and help from the work of individuals in the research community, and we would like to acknowledge their help. We would like to thank Prof. Dariusz Uciński for his invaluable help during the early stages of our research. He introduced us to the framework of optimal observations in distributed parameter systems from the very basics up to the state of the art, and then beyond. His monographs and publications are the foundation of our research (Chap. 2). Our thanks are addressed to Dr. Maciej Patan for his help and insight on the implementation of our problems in MATLAB. Dr. Abdelhaq El Jai and El Hassan Zerrik’s research on sensors and actuators in distributed parameter systems has always kept us aware of important notions in the field of distributed parameter systems (Chap. 2). Moreover, the works of Drs. Michael A. Demetriou and Islam I. Hussein are a source of stimulation to our research efforts.

We would like to express our thanks to Dr. Mac McKee from the Utah Water Research Laboratory. His collaboration on remote sensing using unmanned aerial vehicles was the trigger for some of our later research directions and a constant reminder to short-term applications of our work (Chaps. 4 and 7). Our sincere thanks go to Dr. Om Prakash Agrawal for his work and guidance in the field of fractional optimal control problems (Chap. 7).

This research has been supported by the USU Presidential Fellowship (2006–2007), Utah Water Research Lab (UWRL) MLF Seed Grant (2006–2010), and the NSF International Research and Education in Engineering (IREE) Grant #0540179 “DDDAS/SEP: Optimal Interlaced Distributed Control and Distributed Measurement with Networked Mobile Actuators and Sensors”. In Summer 2010, Christophe Tricaud was also supported by NSF Grant #0851709 “REU Site: Mobile Actuator and Sensor Networks”. Christophe Tricaud would like to express his sincere thanks to his wife Dr. Qianru Li, his son Dorian, and his parents Chantal and Alain for their constant support. He would also like to thank former and current CSOIS members: Dr. Haiyang Chao, Austin Jensen, Long Di, Jinlu Han, Calvin Coopmans, Shayok Mukhopadhyay, Yiding Han, Dr. Yan Li, and Dr. Ying Luo for their support during his Ph.D. studies in CSOIS at Utah State University.

YangQuan Chen would like to thank his wife Dr. Huifang Dou and his sons Duyun, David, and Daniel for their patience, understanding, and complete support throughout this work. This monograph was finalized during his sabbatical semester at University of Pretoria, South Africa, in Spring 2011, and he appreciates Professors Xiaohua Xia and Ian Craig for hosting his productive stay. He also wishes to thank Dr. Kecai Cao for his assistance in carefully proof reading this monograph.

Finally, we would like to give our thanks to Oliver Jackson of Springer for his interests in this book project and to Charlotte Cross, Editorial Assistant (Engineering), Springer London, for much help. Finally, we are thankful to the three anonymous reviewers for their detailed comments and recommendations that helped make this monograph more complete and technically sound.

Contents

1 Introduction 1
1.1 Background on Cyber-physical Systems and Distributed Parameter Systems 1
1.1.1 Cyber-physical Systems 1
1.1.2 Distributed Parameter Systems 7
1.2 Motivations for Monograph Research and Application Scenarios . 7
1.2.1 Optimal Measurements in DPS 8
1.2.2 Scenarios for Optimal Operations of a Mobile Actuator/Sensor Network 9
1.2.3 Fractional-Order Cyber-physical Systems (FOCPS) 12
1.3 Summary of Monograph Contributions 13
1.4 Preview of Chapters 14

2 Distributed Parameter Systems: Controllability, Observability, and Identification 17
2.1 Mathematical Description 17
2.1.1 System Definition 17
2.1.2 Actuator Definition 18
2.1.3 Sensor Definition 19
2.2 Regional Controllability 21
2.3 Regional Observability 22
2.4 Parameter Identification and Optimal Experiment Design 24
2.4.1 System Definition 24
2.4.2 Parameter Identification 25
2.4.3 Sensor Location Problem 25
2.4.4 Sensor Clustering Phenomenon 26
2.4.5 Dependence of the Solution on Initial Parameter Estimates 27
2.5 Chapter Summary 29

3 Optimal Heterogeneous Mobile Sensing for Parameter Estimation of Distributed Parameter Systems 31
3.1 Introduction 31

3.2 Optimal Sensor Location Problem 32
3.3 Mobile Sensor Model 35
3.3.1 Node Dynamics 35
3.3.2 Pathwise State Constraints 35
3.3.3 Parameterization of Vehicle Controls 36
3.4 Characterization of Optimal Solutions 37
3.5 Optimal Control Formulation of the Search for the Candidate Support Point 40
3.6 Illustrative Example 41
3.7 Optimal Measurement Problem in the Average Sense 43
3.7.1 A Limitation of the Design of Optimal Sensing Policies for Parameter Estimation 43
3.7.2 Problem Definition 45
3.7.3 An Illustrative Example 46
3.8 Chapter Summary 49

4 Optimal Mobile Remote Sensing Policies 51
4.1 Introduction 51
4.1.1 Literature Review 51
4.1.2 Problem Formulation for PDE Parameter Estimation 52
4.2 Optimal Measurement Problem 54
4.2.1 Mobile Sensor Model 54
4.2.2 Problem Definition 56
4.3 Optimal Control Formulation 56
4.4 An Illustrative Example 58
4.5 Chapter Summary 59

5 Online Optimal Mobile Sensing Policies: Finite-Horizon Control Framework 63
5.1 Introduction 63
5.2 Optimal Mobile Sensing Policy: Finite-Horizon Closed-Loop Solution 64
5.2.1 A DPS and Its Mobile Sensors 64
5.2.2 Interlaced Optimal Trajectory Planning 65
5.2.3 Illustrative Simulations 67
5.2.4 A Second Illustrative Example 67
5.3 Communication Topology in Online Optimal Sensing Policy for Parameter Estimation of Distributed Parameter Systems 71
5.3.1 The Interlaced Scheme with Communication Topology . . 71
5.3.2 An Illustrative Example 72
5.4 Convergence of the Interlaced Scheme 75
5.5 Chapter Summary 77

6 Optimal Mobile Actuation/Sensing Policies for Parameter Estimation of Distributed Parameter Systems 79
6.1 Introduction 79
6.1.1 Problem Formulation for PDE Parameter Estimation 80

6.2 Optimal Actuation Problem 81
6.2.1 Mobile Actuator Model 81
6.2.2 Problem Definition 82
6.2.3 An Illustrative Example 83
6.3 Optimal Measurement/Actuation Problem 85
6.3.1 Mobile Sensor/Actuator Model 85
6.3.2 Problem Definition 91
6.3.3 An Illustrative Example 92
6.4 Chapter Summary 94

7 Optimal Mobile Sensing with Fractional Sensor Dynamics 97
7.1 Introduction 97
7.2 Fractional Optimal Control Problem Formulation 98
7.3 Oustaloup Recursive Approximation of the Fractional Derivative Operator 99
7.4 Fractional Optimal Control Problem Reformulation. I 101
7.5 Impulse-Response-Based Linear Approximation of Fractional Transfer Functions 102
7.5.1 Approximation Method 102
7.5.2 Suboptimal Approximation of the Fractional Integrator . . 104
7.6 Fractional Optimal Control Problem Reformulation. II 105
7.7 Illustrative Examples 107
7.7.1 A Linear Time-Invariant Problem 107
7.7.2 A Linear Time-Variant Problem 108
7.8 Optimal Mobile Sensing Policies with Fractional Sensor Dynamics 110
7.8.1 Sensor Dynamics 110
7.8.2 Optimal Measurement Problem 111
7.8.3 Optimal Control Problem Reformulation 112
7.8.4 An Illustrative Example 113
7.9 Chapter Summary 115

8 Optimal Mobile Remote Sensing Policy for Downscaling and Assimilation Problems 117
8.1 Background on Downscaling and Data Assimilation 117
8.1.1 Downscaling 117
8.1.2 Data Assimilation 120
8.2 Downscaling and Assimilation Problems for Surface Soil Moisture 124
8.2.1 Introduction 124
8.2.2 Kaheil and McKee's Algorithm 124
8.3 Introduction of UAV-Based Remote Sensors 128
8.4 Optimal Trajectories for Data Assimilation 128
8.4.1 Description of the Problem 128
8.4.2 Problem Formulation 130
8.4.3 Numerical Method to Find the Solution 130
8.5 An Illustrative Example 131
8.5.1 System's Description 131

8.5.2 Results 131
8.6 Chapter Summary 133

9 Conclusions and Future Work 135
9.1 Conclusions 135
9.2 Future Research Directions 136
9.2.1 Communication Topology Influence on Regional Controllability and Observability for DPS 137
9.2.2 Directed Communication Topologies 137
9.2.3 Regional Identifiability of a DPS 137

Appendix A Notation 139
A.1 General Notation 139
A.2 Special Notation in Chap. 2 140
A.3 Special Notation in Chap. 3 140
A.4 Special Notation in Chap. 4 140
A.5 Special Notation in Chap. 6 140
A.6 Special Notation in Chap. 7 141

Appendix B RIOTS Tutorial 143
B.1 Introduction 143
B.2 Features of RIOTS_95 144
B.3 Class of Optimal Control Problems Solvable by `RIOTS_95` 145

Appendix C Implementations 147
C.1 Remote Sensors Trajectory Optimization 147
C.2 Online Scheme for Trajectory Optimization 152
C.3 Fractional-Order Trajectory Optimization 157

References 161

Index 169

Acronyms

CPS	Cyber-physical System
DPS	Distributed Parameter System
EKF	Extended Kalman Filter
EnKF	Ensemble Kalman Filter
ESD	Empirical/Statistical Downscaling
FHC	Finite Horizon Control
FIM	Fisher Information Matrix
FOCP	Fractional Optimal Control Problem
FOCPS	Fractional Optimal Cyber-physical System
GCM	Global Circulation Model
HAB	Harmful Algal Bloom
HVAC	Heating, Ventilating, and Air Conditioning
IOOC	Integer Order Optimal Control
IT	Information Technology
ITSE	Integral Time Squared Error
LCFD	Left Caputo Fractional Derivative
LTI	Linear Time Invariant
LTV	Linear Time Varying
LRLFD	Left Riemann–Liouville Fractional Derivative
MAS-net	Mobile Actuator and Sensor Networks
MIMO	Multiple-Input Multiple-Output
MLE	Maximum Likelihood Estimation
MPC	Model Predictive Control
OED	Optimum Experimental Design
ORA	Oustaloup's Recursive Approximation
PDE	Partial Differential Equation
QoS	Quality of Service
RCM	Regional Climate Model
RIOTS	Recursive Integration Optimal Trajectory Solver
SISO	Single-Input Single-Output
SVM	Support Vector Machine
UAV	Unmanned Aerial Vehicle
WSN	Wireless Sensor Network

Chapter 1
Introduction

1.1 Background on Cyber-physical Systems and Distributed Parameter Systems

1.1.1 Cyber-physical Systems

1.1.1.1 What is a Cyber-physical System?

The term cyber-physical systems is one of the new buzzwords in the engineering community. It originates from the need to have a denomination for a new category of embedded systems where the emphasis was made on the increased interactions between the physical part and the computational part of the system [95]. It was loosely defined by the National Science Foundation (NSF) as "the tight conjoining of and coordination between computational and physical resources" [72]. Since its emergence, the term CPS has been given a lot of definitions, and most of these definitions depend on the field of research of the people giving them. For example, CPSs are defined in [134] in the following way: "Cyber-physical Systems are a next-generation network-connected collection of loosely coupled distributed cyber systems and physical systems monitored/controlled by user defined semantic laws". This definition reflects the point of view of the computer engineering community. The emphasis is made on the software and hardware and not equally on the physical part in itself. The system considered is mostly "cyber" and does not take into full account the "tight conjoining" mentioned by the NSF. This vision of a CPS is illustrated in Fig. 1.1 [134]. Similar definitions can be found in [121, 135].

The definition of CPS for the computer science community has a larger scale than the original definition from NSF:

"The Internet has made the world 'flat' by transcending space. We can now interact with people and get useful information around the globe in a fraction of a second. The Internet has transformed how we conduct research, studies, business, services, and entertainment. However, there is still a serious gap between the cyber world, where information is exchanged and transformed, and the physical world in

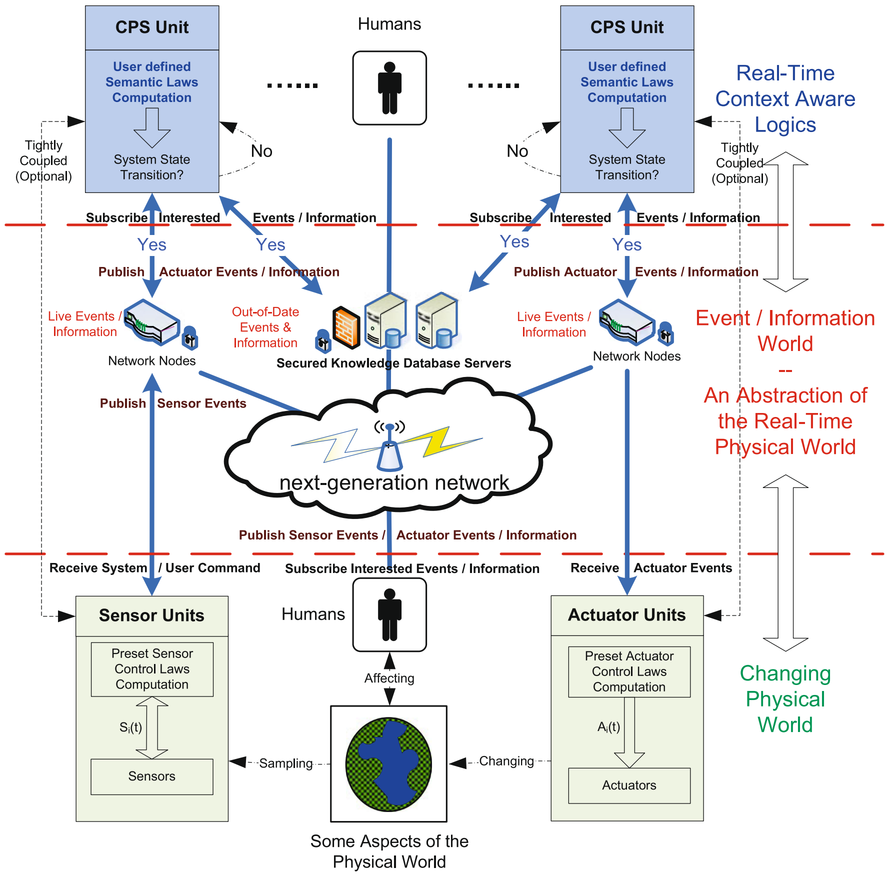

CPS Unit
User defined Semantic Laws Computation
System State Transition?
Humans
CPS Unit
User defined Semantic Laws Computation
System State Transition?
Real-Time Context Aware Logics
Tightly Coupled (Optional)
No
No
Tightly Coupled (Optional)
Subscribe Interested Events / Information
Subscribe Interested Events / Information
Yes
Yes
Yes
Yes
Publish Actuator Events / Information
Publish Actuator Events / Information
Live Events / Information
Out-of-Date Events & Information
Live Events / Information
Network Nodes
Secured Knowledge Database Servers
Network Nodes
Event / Information World -- An Abstraction of the Real-Time Physical World
Publish Sensor Events
next-generation network
Publish Sensor Events / Actuator Events / Information
Receive System / User Command
Subscribe Interested Events / Information
Receive Actuator Events
Sensor Units
Humans
Actuator Units
Preset Sensor Control Laws Computation
Si(t)
Sensors
Affecting
Sampling
Changing
Preset Actuator Control Laws Computation
Ai(t)
Actuators
Changing Physical World
Some Aspects of the Physical World

Fig. 1.1 A prototype architecture of a CPS [134]

which we live. The emerging cyber-physical systems shall enable a modern grand vision for new societal-level services that transcend space and time at scales never possible before". Chap. 1 in [145].

CPS is defined in [34] in the following way: "Computational thinking and integration of computation around the physical dynamic systems form CPS where sensing, decision, actuation, computation, networking and physical processes are mixed." This vision of a CPS is illustrated in Fig. 1.2.

Evidence of the misunderstanding of the term CPS is the emergence of terms such as "networked" CPS [97], or "distributed" CPS [6], or "wireless" CPS [106], or "complex" CPS [28].

CPS is foreseen to become a highly researched area in the years to come with its own conferences [3, 4], books [131], and journals [5].

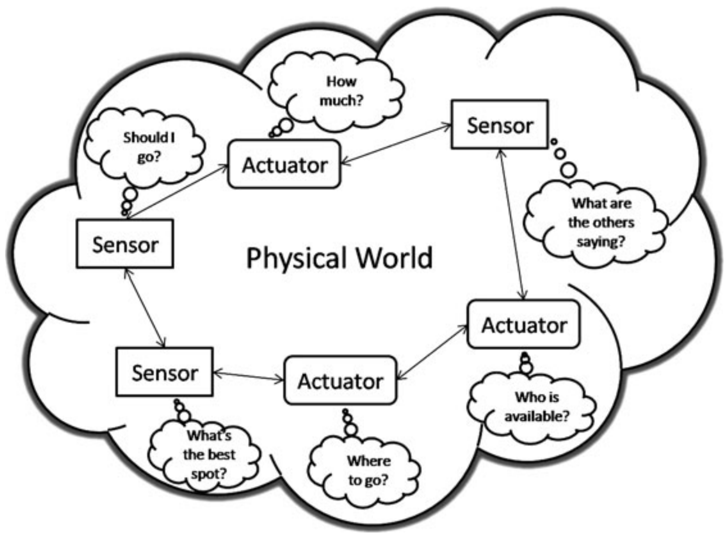


Fig. 1.2 Measurement and control architecture of a CPS

1.1.1.2 CPS Applications

The "Applications of CPS arguably have the potential to dwarf the 20th century IT revolution" [94]. CPS applications can be found in

- tele-physical services [75, 91]
- medical devices and systems [29, 96]
- aerospace [15]
- automotive and air traffic control [175]
- advanced automotive systems [46, 66, 70]
- infrastructure management [68, 99]
- environmental monitoring [53, 172]
- water usage control
- cooperative robotics
- smart buildings
- etc.

Because of the vastness of applications for embedded systems, the area of applications for CPSs is even larger. Here, we describe some of those envisioned by CPS pioneers.

Automotive Transportation Communication between vehicles will make possible the cooperation of nearby vehicles. Many functions of the vehicles will be able to be executed in a distributed manner enhancing its performance, emission reduction, and safety [168]. For example, the braking system not only will ensure the car stops but will also avoid incoming obstacles. If a collision is unavoidable, the system will choose the best trajectory to minimize the impact on the passengers.

By having information about the neighboring vehicles, it will be possible to have a consensus while changing lanes or, by maintaining platooning with small spaces between vehicles, reduced traffic congestion, and improved commute time. We can envision increased communication with the road itself and traffic signs to tell cars about the location of traffic signs, and vice-versa, tell traffic signs that a vehicle with priority is incoming.

Buildings CPS-enhanced buildings are usually called “smart buildings”. Many building functions (such as HVAC and lighting) could significantly improve energy efficiency and lower the overall energy consumption and, consequently, our greenhouse gas emissions. A network of sensors (temperature, humidity, presence detectors) and actuators (HVAC, fans, water heater) embedded into the building could make sense of all the information and operate the building in an optimal way (with respect to energy consumption and uniformity of comfort, for example).

Communication Systems A lot of people see cognitive radios as the CPS of communications [95]. “Cognitive radio signifies a radio that employs model based reasoning to achieve a specified level of competence in radio related domains” [101], but most of the time, a cognitive radio has to fulfill three main functions. It should sense the spectral environment over a wide bandwidth, detect the presence/absence of primary users, and adapt the parameters of their communication scheme only if the communication does not interfere with primary users. Using a CPS infrastructure between radios and cooperative control techniques would allow cognitive radios to use distributed consensus about available bandwidth, improving their overall performance.

Medical Systems There is a growing need for communication between medical devices in modern healthcare systems [88]. In recent years, the quantity of devices for health monitoring and diagnostics has drastically increased, and because the devices lack communication capabilities, healthcare employees have to gather data and make sense of it. One of the main specifications in medical systems is the need for failsafe systems as the malfunction of one system could result in harmful consequences to patients. The main justification for the need for improvement can be seen in statistical reports such as [71] in which numbers say that, of the 284,798 deaths that occurred among patients who developed one or more patient safety incidents between 2003 and 2005, 247,662 could have been avoided (89%).

Water Distribution Systems When assuming the enhancement of the current infrastructure of water distribution systems with networked flow-meter, water quality sensors, and gates, one can foresee improvements in water conservation and efficient power management [78]. Sensor data can be assimilated into a global hydraulic model that can predict the hydraulic state of the system or optimize pumping operations. Potential leaks will also be easier to detect and locate, allowing quick repair of the infrastructure. Chemical attacks to the network could also be detected early and allow a quick response of the authorities.

1.1.1.3 Research Challenges

There are two main directions for research in CPSs; the first one consists of dealing with the increased complexity of embedded systems [27], and the other is to build CPSs from scratch [6]. Because of the relatively young age of CPSs, a lot of research challenges have been identified, each of them related to one of the engineering fields that CPSs belong to.

Security The emergence and growth of CPSs will lead them to be used in critical infrastructure and industry. It is therefore necessary to develop hardware and software solutions to protect them from attacks. Among the potential attackers, several profiles have been identified [30]. Cybercriminals attack blindly any networked system as long as they can enter its operating system. Even though the attacks mean no harm, they can leave the system infected with malware and may modify its functionalities. Disgruntled employees constitute the largest threat in CPSs, the reason being their authorized access to the system's network. Terrorists, activists, and organized criminal groups can be identified as a threat to CPSs as attacks on them are cheaper, less risky, and not constrained by distance. A CPS usually communicates on a simple network: There is usually a single server, the number of nodes is known, the communications are poorly encrypted, and the number of protocols is limited. The amount of work required to prepare an intrusion on such a network may be small, but so would be the implementation of security measures. The identified research directions for security in CPSs are low-cost security, intrusion detection, redundancy, and recovery. Many research directions in CPS security can be found in [7, 30, 136].

Communication and Data Fusion In general, control loops are designed so that all the measurements from the sensors within the network (usually vast in space) are transmitted to the actuators and the actuator node does the computation of the control law. Such a method is very cumbersome for the network and usually results in long communications from sensor node to sensor node up to the actuator. However, not all the sensor data are necessary for the control purpose, and most of the sensor data could be fused to reduce the quantity of information flowing through the network. By performing small computations at each node, only the valuable part of the measurement data should be transmitted to the actuator node [133].

Software The software of CPSs will be very challenging to design. There are many reasons for these challenges, and here is a list of the most important ones [164, 170]:

- CPSs will be composed of a large variety of hardware platforms, and hence they will require the implementation of distributed and embedded applications. These applications themselves will have to be diverse in order to work with all the platforms.
- There will be a need for a unified component model. CPSs are globally virtual and locally physical. It will be required for the component to reflect this characteristic and provide a unified view from local components to global systems.

- The gaps in semantics of programming languages will have to be reduced or closed. In current embedded systems, semantics of physical, hardware, and software components are significantly different. To combine all these components in a system setting, researchers will have to bridge those semantic gaps and come up with a unified programming language.

Scalability Scalability will also be a big research challenge in CPSs. Software and hardware should be developed in such a manner that the design of a CPS with 1000 nodes should be as simple as one with 10 nodes.

Resilience Because of the potentially large scale of CPSs, their maintenance could be costly (especially for nature monitoring CPSs). Resilient CPSs would be of interest to reduce those costs but also to avoid absence of data in critical infrastructures. Three main research directions for resilience in CPSs have been identified [157]:

- Network self-organization to preserve/increase resilience. If the network is designed with self-healing and reconfiguration methodologies, its resilience would be increased. The information could be routed in an organic manner to naturally avoid problems.
- Risk mitigation via eNetworks. The network could be given the capability of quickly evaluating the system vulnerability with respect to new threats and react accordingly to remedy the vulnerability.
- Study of the impact of interdependencies. By identifying the critical parts of the system (the ones whose failure leads to the system's failure), a strategy reorganizing or shutting down major hubs could improve the robustness of the overall system.

Quality of Service (QoS) Since the array of applications of CPSs is extremely large, it is not hard to envision that CPSs will be omnipresent in our daily lives. Therefore, it will be necessary for them to provide QoS support because they will have to fulfill requirements from various sources (specifications, users, etc.) [169]. In CPSs, QoS can be achieved in several ways; communication protocols need to be aware of the QoS requirements and need to be designed with constraints on the platform heterogeneity to optimize the flow of information. The CPS will have to manage its resources (computation time, memory, bandwidth, energy) in a dynamic way and will probably need a resource managing application taking into account QoS specifications.

Modeling Most of the research directions in CPSs have been introduced by embedded systems engineering and computer science scholars. Therefore, most of the literature eludes the problems of modeling for CPSs. We believe that most of the literature on CPSs in the near future will be limited to single-input single-output (SISO) and multiple-input multiple-output (MIMO) finite-dimensional physical systems. In this monograph, we believe that the "physical" part of a CPS should be as complex as its "cyber" part. Therefore, we concentrate our efforts in modeling the physical part using a PDE, which is infinite dimensional in nature. Before creating a CPS, a mandatory step will be to understand its physical part and therefore develop a model for its dynamics. Next, we describe the system structure we consider in this monograph.

1.1.2 Distributed Parameter Systems

1.1.2.1 Definition of a DPS

DPSs are dynamical systems in which the states depend on not only time but also space or spatial variables, which makes the system infinite dimensional. In the literature, DPSs are also called spatio-temporal dynamic systems. They are usually used in opposition to lumped parameter systems. The usual model of a DPS involves partial differential equations (PDEs). In many cases, the physical part of a CPS cannot be modeled with a lumped parameter approach, and a DPS would be the best fit.

There are several, well-identified research directions [129] in the study of DPSs including optimal control, measurement, model reduction, and numerical methods [50–52].

Mathematical definitions and general results about DPSs are given in Chap. 2.

1.1.2.2 Applications of DPSs

Numerous fields of engineering make use of DPSs for modeling. The following is a short collection of them:

- Fluid dynamics [21]
- Signal transmission lines dynamics [67]
- Soil dynamics [113]
- Electromagnetic dynamics [139]
- Heat dynamics [20]

A more complete review of DPS applications can be found in [67]. With the technology advances, many new aspects in DPS research involving CPS contexts are identified and addressed in this monograph while these aspects were not discussed in previous DPS research.

1.2 Motivations for Monograph Research and Application Scenarios

When dealing with lumped parameter systems (SISO or MIMO), the decision on where to implement the sensors and the actuators is a rather straightforward process that is seldom discussed. However, if the system is of a distributed nature, their properties, location, and communication topologies have a big impact on the way the system is operated. For example, a mobile sensor will be able to ambulate in the domain of interest, and the design of its optimal sensing trajectory becomes a research problem. Similarly, the global performance of a control strategy will be improved if the sensors can communicate with the actuators. Now, we will present motivations for this monograph research and list some motivating application scenarios in this section.

1.2.1 Optimal Measurements in DPS

States in DPSs vary both spatially and temporally, but it is generally impossible to measure them over the whole spatial domain. Consequently, we are faced with the design problem on how to locate a limited number of measurement sensors so as to obtain as much information as possible about the process at hand. The location for the available sensors is not necessarily dictated by physical considerations or by intuition, and, therefore, some systematic approaches must be developed in order to reduce the cost of instrumentation and to increase the efficiency of measurement.

There are several lines of research linked with optimal location of sensors in DPSs—observability, state estimation, parameter estimation, detection of unknown sources, and model discrimination—and each of them is linked to some specific application scenarios.

1.2.1.1 Observability

In a DPS, the notion of observability is linked to the possibility to reconstruct the state of the system in a finite duration using sensor measurements. It is obvious that the location and coverage of the sensors are going to affect the observability of a given DPS. Therefore, using the observability as a performance criterion, it is possible to optimize the location/trajectories of the sensors to maximize the observability of the system [124].

1.2.1.2 State Estimation

Similar to the optimal measurement for best observability, the problem of optimal sensor location for state estimation consists of finding the best location so as to reconstruct the state of the system with minimum estimation error variance. However, it may not be necessary to seek the reconstruction of the state over the whole domain but, instead, to look at the reconstruction on the boundary [93].

1.2.1.3 Parameter Estimation

The parameter estimation problem is usually linked with a forecast problem. When faced with a DPS, the general form of its dynamics usually may be known (diffusion, advection, hyperbolic), but the parameters may not be. It is therefore necessary to look into systematic methodologies to determine the optimal locations/trajectories of stationary/mobile sensors for parameter estimation.

A relevant example is the design of optimal sensor location for air quality monitoring. The purpose of sensor networks in air quality monitoring is to measure pollutant concentration but, more important and practical, to produce information regarding the expected finite levels of those pollutants. Such a forecast can only be obtained by using a fog diffusion model. In general, fog can be modeled by an advection–diffusion partial differential equation. For the forecast to be accurate, a calibration is required by estimating the spatially varying turbulent diffusivity tensor of the model based on the data collection obtained by sensors. Because of limited resources, the problem arises on where to install those sensors to obtain the most precise model [152].

1.2.2 Scenarios for Optimal Operations of a Mobile Actuator/Sensor Network

The main motivation and application scenario driving the research effort in this monograph comes from our research center's own project called MAS-net [2]. This project envisions the use of networked mobile sensors and mobile actuators to identify, estimate, forecast, and control a DPS with the following scenario. More details about the MAS-net project can be found in [33, 39, 102]. An illustration of the scenario is given in Fig. 1.3.

1. A plume of a harmful chemical or biological agent is released into an urban environment. The dynamics of the plume in the air can be modeled by a diffusion process with the addition of transport because of the wind and specific boundaries due to the surrounding buildings.
2. A fraction of the harmful plume is detected by one sensor within a widespread array of networked static sensors.
3. The detection of a harmful agent triggers the deployment of a team of unmanned aerial vehicles (UAVs) equipped with chemical concentration sensors and communication capabilities that flies into the plume to estimate its parameters and the evolving boundary.
4. The group of UAVs send back to the ground station all the data they gather as well as their current locations.
5. Based on the original assumptions on the dynamics of the plumes, the data received by the main station help the estimation of the parameters of the diffusion plus transport process. Reciprocally, new destinations are assigned to the UAVs to gather more sensible data with respect to parameter estimation and/or state estimation.

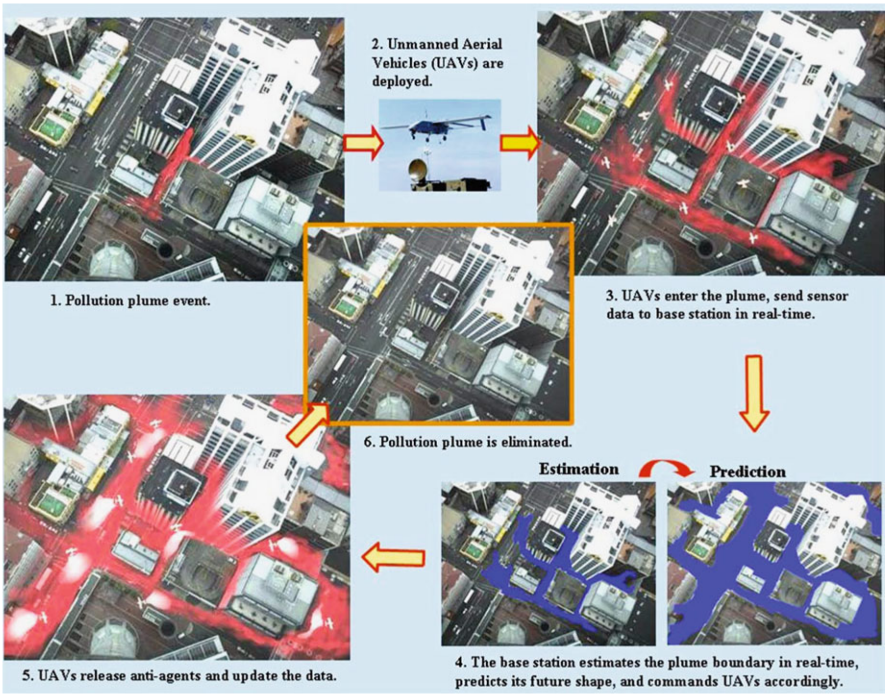

2. Unmanned Aerial Vehicles (UAVs) are deployed.
1. Pollution plume event.
3. UAVs enter the plume, send sensor data to base station in real-time.
6. Pollution plume is eliminated.
Estimation
Prediction
5. UAVs release anti-agents and update the data.
4. The base station estimates the plume boundary in real-time, predicts its future shape, and commands UAVs accordingly.

Fig. 1.3 Application scenario for the MAS-net project [131]

6. Once the estimation of the parameters has converged and the base station is confident with its identification, the UAVs are sent into the optimal locations within the plume to release an anti-agent to mitigate or neutralize the harmful effects.
7. Once the plume has been eliminated, the UAVs return to the base station.

From our experience in the framework of optimal operations of a mobile actuator/sensor network, many potential applications exist in the field of environmental science. Here, we describe a few we have identified so far.

1.2.2.1 Algal Blooms Monitoring and Control Using Mobile Actuator/Sensor Networks

Harmful algal blooms (HABs) are a menace to water wildlife as the release of toxins into habitats can generate a large population death count. However, the lack of systematic approach to detect, forecast, and control HABs means that most scientists study their aftermaths rather than their prevention. So far, scientists have used poorly calibrated tools for their problems. They either used satellite images, which are too low in resolution, both spatially and temporally, to accurately observe the dynamics of algaes, or used data collected by monitoring stations, which lack enough spatial information. We believe that the solution to monitor algae effectively lies in

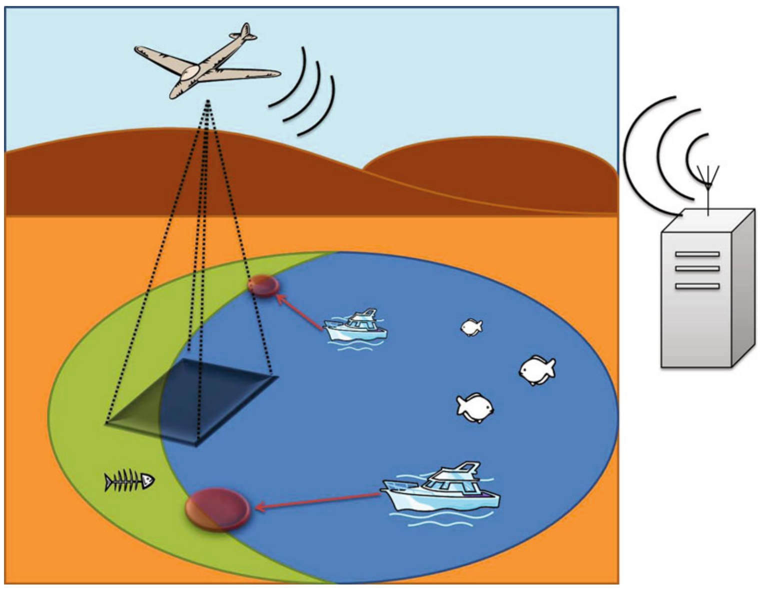


Fig. 1.4 Application scenario for algal blooms monitoring and control

the emergence of new remote sensing platforms that are UAV multispectral imaging [32], which improve the resolution of the measurements while still covering an area large enough for dynamic modeling. In addition, there is an increasing number of techniques for mitigation of HABs. Using information from sensors, the actuators could be sent where the release of mitigating agents would have the most impact on either the population of algae or the harmful chemical. An illustration for these scenarios is depicted in Fig. 1.4.

1.2.2.2 Wildfire Control Scenario

Another scenario where the consideration of optimal sensor and actuator location could be very beneficial resides in wildfire control. Imagine the following scenario (Fig. 1.5):

1. During the dry season, the monitoring of forests is increased to detect potential wildfire.
2. Thanks to the acute monitoring, a wildfire is detected in its early stage.
3. A group of UAVs is sent to detect the boundary of the fire, and fire fighters are dispatched in the area surrounding the fire, waiting for instructions.
4. Using a mathematical model combined with information such as wind speed and direction, humidity, forest density, and current location of the fire, an algorithm provides information to the fire marshal on where he should send the different resources available. For example, fire fighters could fight the fire at its boundary while water bombers release fire retardant or water inside the blaze.

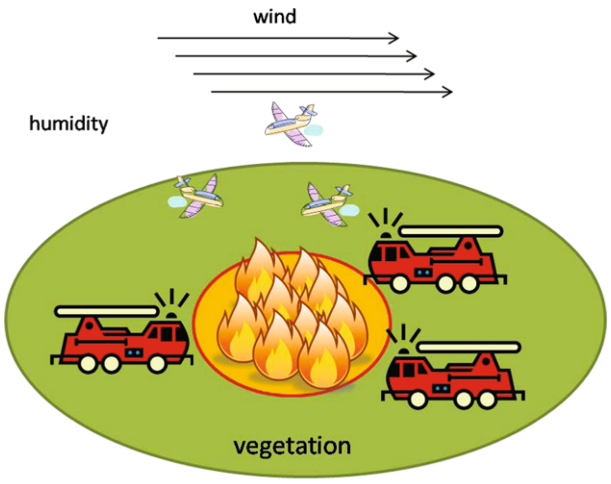


Fig. 1.5 Application scenario for wildfire control

5. The wildfire is quickly under control, and the resources can be sent to another location if necessary.
6. During the wet season, the data gathered during the wildfire season are analyzed to improve the models and their calibration.

For more scenarios on CPSs, refer to Dr. Chen's CPS talk slides in [38].

1.2.3 Fractional-Order Cyber-physical Systems (FOCPS)

A large number of real-world physical systems can be more properly described by fractional-order dynamics, meaning that their behavior is governed by fractional-order differential equations [110]. As an example, it has been illustrated that materials with memory and hereditary effects, and dynamical processes, including gas diffusion and heat conduction, in fractal porous media can be more adequately modeled by fractional-order models than integer-order models [173].

During the past decade, a new category of systems has developed interest in fractional dynamics: scale-free networks. The concept of a scale-free network was introduced because it allows the merging of the theories of complex systems in biology and in physical and social studies. The most peculiar property of a scale-free network is its invariance to changes in scale. The term scale-free refers to a system defined by a functional form $f(x)$ that remains unchanged within a multiplicative factor under a rescaling of the independent variable x. Effectively, this means power-law forms, since these are the only solutions to $f(ax) = bf(x)$ for all $x \in \mathbb{R}$. The scale-invariance property is often interpreted as self-similarity. Any part of the scale-free network is stochastically similar to the whole network, and parameters are assumed to be independent of the system size. Other mathematical laws that might fit to describe similar qualitative properties of the network degree distribution will not satisfy an important condition of the scale invariance. Therefore, a network is defined as scale-free if a randomly picked node has k connections with other nodes with a probability that follows a power-law $p(k) \sim k^{\gamma}$, where γ is the power-law exponent.

The scale-free framework has been introduced because of the need to find a new type of model that can match the self-similarity properties of biological and social networks.

In case of an epidemic, gathering information about the infected people is crucial. The traditional source of information comes from healthcare practitioners (hospitals, ERs, physicians) and helps the determination of the stage of the epidemic. Nowadays, with the emergence of online social networks [1], information about people's health is also available by other means. Monitoring those networks could allow authorities to obtain increased information from people who do not have health insurance and do not go to the hospital. With such an elaborate picture of the state of the network, we can consider the problem of vaccination to fight the epidemic in the most efficient way, for example, by prioritizing the most important nodes of the network to limit the propagation of the virus. As mentioned earlier, because of the self-similarity of the social network, the decision of whom to be given the vaccine to with what priority becomes a fractional optimal control problem (FOCP).

An FOCP is an optimal control problem in which the criterion and/or the differential equations governing the dynamics of the system contain at least one fractional derivative operator. Integer-order optimal controls (IOOCs) have been discussed for a long time, and a large collection of numerical techniques have been developed to solve IOOC problems. The collection of optimal control solvers is rather large [82, 126, 161]. It is therefore of interest to make use of such a solver to solve FOCPs. To achieve this goal, we need to use rational approximations of the fractional differentiation operator and reformulate the FOCP into an IOOC problem accordingly [143].

1.3 Summary of Monograph Contributions

This monograph provides the following contributions to the state of the art of CPS research:

- An approach is proposed to joint optimization of trajectories and measurement accuracies of mobile nodes in a mobile sensor network collecting measurements for parameter estimation of a distributed parameter system.
- We propose a method to obtain the optimal trajectories of a team of mobile robots remotely monitoring a distributed parameter system for its parameter estimation.
- Given a DPS with unknown parameters, a numerical solution method for generating and refining a mobile sensor motion trajectory for the estimation of the parameters of DPS in the "closed-loop" sense is provided.
- We discuss the influence of the communication topology of mobile sensors on the estimation of the parameters of a distributed parameter system.

- We discuss the problem of determining optimal sensors' trajectories so as to estimate a set of unknown parameters for a system of a distributed nature where the bounds on the parameters' values are known.
- We introduce a numerical procedure to optimize the trajectory of mobile actuators to find parameter estimates of a distributed parameter system given a sensor configuration.
- We introduce a framework to solve the problem of determining optimal sensors and actuators' trajectories so as to estimate a set of unknown parameters in what constitutes a CPS.
- We discuss fractional-order optimal control problems and their solution by means of rational approximation. The original problem is then reformulated to fit the definition used in general-purpose optimal control problem (OCP) solvers.
- A different direction to approximately solving FOCPs is introduced. The method uses a rational approximation of the fractional derivative operator obtained from the singular value decomposition of the Hankel data matrix of the impulse response and can potentially solve any type of FOCPs.
- We propose a methodology to optimize the trajectories of mobile sensors whose dynamics contains fractional derivatives to find parameter estimates of a distributed parameter system.
- We introduce a methodology to obtain the optimal trajectories of a group of mobile remote sensors for scale reconciliation for surface soil moisture.

1.4 Preview of Chapters

The outline of this monograph follows a direction from introductory notions and definitions to the development of methodologies for optimal sensing and actuation under particular conditions. This work is divided into nine chapters described as follows.

Chapter 1 The important terms motivating this work are defined, and extensive literature review is conducted. Motivation and application scenarios are provided for various research areas. The contributions of the monograph are summarized.

Chapter 2 We provide important definitions from the field of DPSs. We introduce the dynamic equations of the system, the mathematical descriptions of a sensor and an actuator. The concepts of regional controllability and observability for DPSs are derived from those definitions. Definitions of the parameter estimation and optimal sensor location framework are given. We discuss two issues linked to the framework: the sensor clusterization phenomenon and the dependence of the solution on initial parameter estimates.

Chapter 3 We show that some methods from the optimum experimental design (OED) framework for linear regression models can be applied to the formulation of the mobile sensor trajectory design problem for DPS parameter estimation when it is desirable to simultaneously optimize the number of sensors, their trajectories, and their accuracies (noise characteristics).

Chapter 4 We extend the existing optimal sensor location to encompass the case of remote sensors. We introduce a remote sensing function linking the mobility domain and the sensing domain. We provide an example that can be linked with the optimal trajectories of UAVs carrying imaging payloads.

Chapter 5 To circumvent the issue of the dependence of the optimal location on the parameter estimates, we introduce the design of moving sensor optimal trajectories, which does not rely on initial estimates of the parameters but instead is based on knowledge of upper and lower bounds of the parameter values and on offline computation type of solution. We also introduce an online scheme where the parameter estimates are evaluated iteratively, which allows us to introduce the concept of communication topology into the framework.

Chapter 6 The "stimulation" of the system being an implicit variable to the parameter estimation problem, we introduce the optimization of the trajectories of a group of mobile actuators. We solve the problem of optimal actuation for parameter estimation with given sensor location/trajectories. We combine this new framework with the optimal sensor location framework to optimize both the trajectories of sensors and actuators together.

Chapter 7 We introduce a new formulation toward solving a wide class of fractional optimal control problems. The formulation made use of an approximation to model the fractional dynamics of the system in terms of a state space realization. This approximation creates a bridge with a fractional optimal control problem and a readily available optimal control solver. The methodology allows us to reproduce results from the literature as well as to solve the more complex problem of optimal trajectories of sensors with fractional dynamics.

Chapter 8 We focus on the downscaling problem in the framework of surface soil moisture measurement. Our purpose is to introduce a new methodology to transform or fuse low-resolution remote sensing data, ground measurements, and low-altitude remote sensing (typically images obtained from a UAV) into a high-resolution data set.

Chapter 9 The contributions of this monograph are summarized. Discussion of potential future research directions is presented.

Appendices A–C We provide a list of general notation used in this monograph as well as specific notation for several chapters. We give a short tutorial about the optimal control problem solver (RIOTS_95) used in the illustrative examples. We provide the MATLAB code for some of the illustrative examples used in this monograph.

Chapter 2
Distributed Parameter Systems: Controllability, Observability, and Identification

2.1 Mathematical Description

We introduce the class of systems to be considered in the framework of this monograph and definitions on configurations of sensors and actuators. Important concepts are defined for parameter identification and optimal experiment design.

This section introduces important concepts for the analysis of distributed parameter systems from the literature [8].

2.1.1 System Definition

Here, we consider a class of linear DPSs whose dynamics can be described by the given state equation

$$\begin{cases} \dot{y}(t) = Ay(t) + Bu(t), \quad 0 < t < T, \\ y(0) = y_0, \end{cases} \tag{2.1}$$

where the state space is given as $Y = L^2(\Omega)$, and the set Ω is a bounded open subset of $\mathbb{R}^n$ with a sufficiently regular boundary $\Gamma = \partial\Omega$. The considered domain Ω stands for the geometrical support of the system defined by (2.1). The operator A is a linear operator describing the dynamics of system (2.1). A generates a strongly continuous semigroup $(\Phi(t))_{t \geq 0}$ on Y. The operator $B \in \mathcal{L}(U, Y)$ (the set of linear maps from U to Y) is the input operator; $u \in L^2(]0, T[; U)$ (space of integrable functions $f :]0, T[\mapsto U$ such that $t \mapsto \|f(t)\|^p$ is integrable on $]0, T[$); U is a Hilbert control space. The considered system can be augmented by the output equation

$$z(t) = Cy(t), \tag{2.2}$$

where $C \in \mathcal{L}(L^2(\Omega), Z)$, and Z is a Hilbert observation space. While such a definition can be used for the analysis of distributed parameter systems, it is fairly abstract when considering controls. That is why we introduce the notions of actuators and sensors, as well as notions of spatial distribution. These notions allow one to study the system not only with respect to the operators A, B, and C, but also with respect to the spatial distribution, location, and number of the actuators and sensors.

Sensors and actuators have two separate roles in a DPS. The actuators provide an excitation on the system, and the sensors give information (measurements) about the state of the system. Both sensors and actuators can be of different natures: zone or pointwise, internal or boundary, stationary or moving, communicating or noncommunicating, collocated or noncollocated.

An important notion of the framework of a DPS is the region. It is generally defined as a subdomain of Ω in which we are especially interested. Instead of considering a problem on the whole domain Ω, it is possible to consider only a subregion ω of Ω. This has allowed the generalization of the concepts, theorems, and results of the analysis of DPSs to any subdomain of Ω. In the following, we give the mathematical definitions for actuators and sensors.

2.1.2 Actuator Definition

Let Ω be an open regular bounded subset of $\mathbb{R}^n$ with a sufficiently regular boundary $\Gamma = \partial\Omega$. The set Ω stands for the geometrical support of a considered DPS [79].

Definition 2.1

1. An actuator is a couple (D, g) where D is the geometrical support of the actuator, $D = \text{supp}(g) \subset \Omega$, and g is its spatial distribution.
2. An actuator (D, g) is defined as:
   - A zonal actuator if D is a nonempty subregion of Ω.
   - A pointwise actuator if D is reduced to a point $b \in \Omega$. In that situation, we have $g = \delta_b$ where δ_b is the Dirac function concentrated at b. The actuator is then denoted as (b, δ_b).
3. An actuator (zonal or pointwise) is called a boundary actuator if its support $D \subset \Gamma$.

An illustration of actuator's support is given in Fig. 2.1. In the previous definition, g is assumed to be in $L^2(D)$. For p actuators $(D_i, g_i)_{1 \le i \le p}$, the control space is $U = \mathbb{R}^p$, and

$$B : \mathbb{R}^p \to L^2(\Omega)$$

$$u(t) \to Bu(t) = \sum_{i=1}^{p} g_i u_i(t)$$

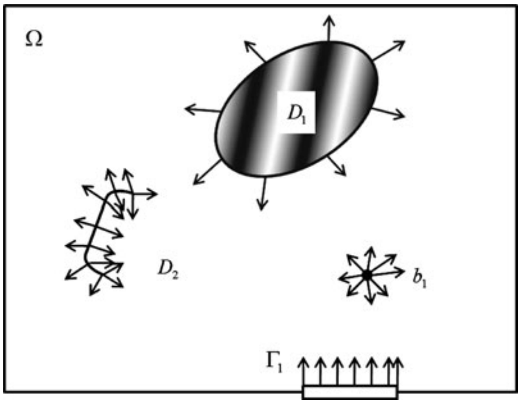


Fig. 2.1 Illustration of actuator's support

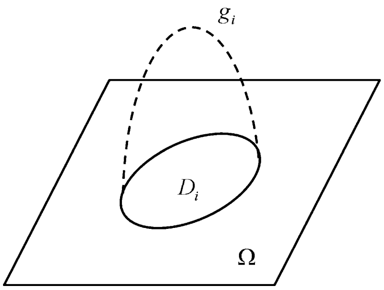


Fig. 2.2 Illustration of the geometrical support and spatial distribution of an actuator

where $u = (u_1, \dots, u_p)^T \in L^2(]0, T[; \mathbb{R}^p)$ and $g_i \in L^2(D_i)$ with $D_i = \text{supp}\,(g_i) \subset \Omega$ for $i = 1, \dots, p$ and $D_i \cap D_j = \emptyset$ for $i \neq j$, and we have

$$B^\star y = \big(\langle g_1, y\rangle, \dots, \langle g_p, y\rangle\big)^T \quad \text{for } z \in L^2(\Omega),$$

where M^T is the transpose of M, and $\langle \cdot, \cdot \rangle = \langle \cdot, \cdot \rangle_Y$ is the inner product in Y, and for $v \in Y$, if $\text{supp}\,(v) = D$, we have

$$\langle v, \cdot \rangle = \langle v, \cdot \rangle_{L^2(D)}.$$

If D is independent of the time instant t, the actuator (D, g) is defined as fixed or stationary. If D varies with t, it is called a moving or mobile actuator denoted by (D_t, g_t), where D_t and g_t are, respectively, the geometrical support and the spatial distribution of the actuator at time t. An illustration of the geometrical support and spatial distribution of an actuator is given in Fig. 2.2.

2.1.3 Sensor Definition

Let us provide some concepts and definitions for sensors in DPSs [79].

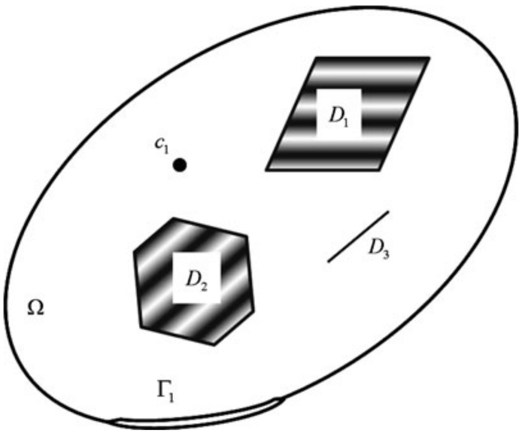


Fig. 2.3 Illustration of sensor's support

Definition 2.2 A sensor is defined as a couple (D, h), where D is the spatial support of the sensor, $D = \text{supp}(h) \subset \Omega$, and h is its spatial distribution.

An illustration of sensor's support is given in Fig. 2.3. It is generally assumed that $h \in L^2(D)$. In a similar fashion, we can define zonal, pointwise, internal, boundary, fixed, or moving sensors. If the output of the system is given by means of q zonal sensors $(D_i, h_i)_{1 \le i \le q}$ with $h_i \in L^2(D_i)$, $D_i = \text{supp}\,(h_i) \subset \Omega$ for $i = 1, \ldots, q$ and $D_i \cap D_j = \emptyset$ if $i \neq j$, then in the case of a zonal output, the DPS's output operator C is defined by

$$C : L^2(\Omega) \to \mathbb{R}^p$$

$$y \to Cy = \left(\langle h_1, y\rangle, \ldots, \langle h_q, y\rangle\right)^T$$

and the output of the sensors is given by

$$z(t) = \begin{bmatrix} \langle h_1, y\rangle_{L^2(D_1)} \\ \langle h_2, y\rangle_{L^2(D_2)} \\ \vdots \\ \langle h_q, y\rangle_{L^2(D_q)} \end{bmatrix}. \tag{2.3}$$

A sensor (D, h) is said to be a zonal sensor if D is a nonempty subregion of Ω. A sensor (D, h) is called pointwise if D is reduced to a point $c \in \Omega$, and $h = \delta_c$ is the Dirac function concentrated at c. The sensor is then denoted as (c, δ_c). For zonal or pointwise sensors, if $D \subset \Gamma = \partial\Omega$, a sensor (D, h) is said to be a boundary sensor. If D does not depend on time, the sensor (D, h) is said to be fixed or stationary; otherwise, it is said to be moving (or scanning) and is denoted as (D_t, h_t). In the case of q pointwise fixed sensors located in $(c_i)_{1 \le i \le q}$, the output function is a vector defined as

$$z(t) = \begin{bmatrix} y(t, c_1) \\ y(t, c_2) \\ \vdots \\ y(t, c_q) \end{bmatrix}, \tag{2.4}$$

where c_i is the position of the ith sensor, and $y(t, c_i)$ is the value of the state of the system in c_i at time t.

2.2 Regional Controllability

Let Ω be an open regular bounded subset of $\mathbb{R}^n$, and $Y = L^2(\Omega)$ be the state space. In what follows, we denote $Q = \Omega \times]0, T[$ and $\Sigma = \partial\Omega \times]0, T[$, and we consider the system described by the state equation

$$\begin{cases} \dot{y}(t) = Ay(t) + Bu(t), \quad 0 < t < T, \\ y(0) = y_0 \in D(A), \end{cases} \tag{2.5}$$

where $D(A)$ is the domain of the operator A. The operator A generates a strongly continuous semigroup $(\Phi(t))_{t \geq 0}$ on Z, $B \in \mathcal{L}(\mathbb{R}^p, Y)$, and $u \in L^2(0, T; \mathbb{R}^p)$. The mild solution y of (2.5), denoted $y(\cdot, u)$, is given by

$$y(t, u) = \Phi(t)y_0 + \int_0^t \Phi(t-s)Bu(s)\,\mathrm{d}s, \tag{2.6}$$

and we have $y(\cdot, u) \in C[0, T; Y]$.

We consider a given region $\omega \subset \Omega$ of positive Lebesgue measure and a given desired state $y_d \in L^2(\omega)$ [55].

Definition 2.3

1. System (2.5) is said to be exactly regionally controllable (or exactly ω-controllable) if there exists a control $u \in L^2(]0, T[; \mathbb{R}^p)$ such that

$$p_\omega y(T, u) = y_d. \tag{2.7}$$

2. System (2.5) is said to be weakly regionally controllable (or weakly ω-controllable) if, given $\epsilon > 0$, there exists a control $u \in L^2(]0, T[; \mathbb{R}^p)$ such that

$$\left\| p_\omega y(T, u) - y_d \right\|_{L^2_\omega} \leq \epsilon, \tag{2.8}$$

   where $y(\cdot, u)$ is given by (2.6), and $p_\omega y$ is the restriction of y to ω.

In the case of pointwise or boundary controls, $B \notin \mathcal{L}(\mathbb{R}^p, Z)$. We consider the operator

$$H : L^2\big(]0, T[; \mathbb{R}^p\big) \to Y$$

defined by

$$Hu = \int_0^T \Phi(T-\tau)Bu(\tau)\,\mathrm{d}\tau \tag{2.9}$$

and

$$p_\omega : L^2(\Omega) \to L^2(\omega) \tag{2.10}$$

defined by

$$p_\omega y = y|_\omega. \tag{2.11}$$

Then, from Definition 2.3, system (2.5) is exactly (respectively weakly) regionally controllable if

$$\mathrm{Im}(p_\omega H) = L^2(\omega)\big(\text{respectively } \overline{\mathrm{Im}\, p_\omega H} = L^2(\omega)\big). \tag{2.12}$$

We have equivalently

$$\overline{\mathrm{Im}(p_\omega H)} = L^2(\omega) \Leftrightarrow \mathrm{Ker}\big(H^\star i_\omega\big) = \{0\}, \tag{2.13}$$

where i_ω holds for the adjoint of p_ω. Characterizations (2.12) and (2.13) are often used in applications. We also have the following result [56].

Lemma 2.4 1. *System* (2.5) *is exactly regionally controllable if and only if*

$$\mathrm{Ker}(p_\omega) + \mathrm{Im}(H) = L^2(\Omega). \tag{2.14}$$

2. *System* (2.5) *is weakly regionally controllable if and only if*

$$\ker(p_\omega) + \overline{\mathrm{Im}(H)} = L^2(\Omega). \tag{2.15}$$

It is easy to show that (2.15) is equivalent to

$$\mathrm{Ker}(H^\star) \cap \mathrm{Im}(i_\omega) = \{0\}, \tag{2.16}$$

where $i_\omega = p_\omega^\star : L^2(\omega) \to L^2(\Omega)$ is given by

$$i_\omega z = \begin{cases} y(x), & x \in \omega, \\ 0, & x \in \Omega \backslash \omega. \end{cases}$$

2.3 Regional Observability

Let z be the state of a linear system with state space $Y = L^2(\Omega)$, and suppose that the initial state y_0 is unknown. Measurements are given by means of an output z depending on the number and the structure of the sensors. The problem to be studied here concerns the reconstruction of the initial state y_0 on the subregion ω. Let Ω be a regular bounded open set of $\mathbb{R}^n$ with boundary $\Gamma = \partial\Omega$, ω be a nonempty subset of Ω, and $[0,T]$ with $T>0$ be a time interval. We denote $Q = \Omega \times]0,T[$ and $\sigma = \partial\Omega \times]0,T[$, and we consider the autonomous system described by the state equation

$$\begin{cases} \dot{y}(t) = Ay(t), & 0 < t < T, \\ y(0) = y_0 & \text{supposed to be unknown,} \end{cases} \tag{2.17}$$

where A generates a strongly continuous semigroup $(\Phi(t))_{t\geq 0}$ on the state space Y. An output function gives measurements of the state y by

$$z(t) = Cy(t), \tag{2.18}$$

where

$$C : y \in L^2(]0,T[;Y) \to z \in L^2\big(]0,T[;\mathbb{R}^q\big) \tag{2.19}$$

depends on the sensors' structure. In the case where the considered sensor is pointwise and located in $b \in \Omega$, we have, with (2.18),

$$z(t) = \int_\Omega y(x,t)\delta(x-b)\,\mathrm{d}x = y(b,t). \tag{2.20}$$

The problem consists in the reconstruction of the initial state, assumed to be unknown, in the subregion ω. We consider the following decomposition:

$$y_0 = \begin{cases} y^e, & x \in \omega, \\ y^u, & x \in \Omega \backslash \omega, \end{cases} \tag{2.21}$$

where y^e is the state to be estimated, and y^u is the undesired part of the state. Then, the problem consists in reconstructing y^e with the knowledge of (2.17) and (2.18). As system (2.17) is autonomous, (2.18) gives

$$z(t) = C\Phi(t)y_0 = K(t)y_0, \tag{2.22}$$

where K is an operator $Y \to L^2(]0,T[;\mathbb{R}^q)$. The adjoint $K^\star$ is given by

$$K^\star y = \int_0^T \Phi^\star(s)C^\star z(s)\,\mathrm{d}s. \tag{2.23}$$

We recall that system (2.17) with the output (2.18) is said to be weakly observable if $\mathrm{Ker}(K) = \{0\}$. The associated sensor is then said to be strategic [56]. Consider now the restriction mapping

$$\chi_\omega : L^2(\Omega) \to L^2(\omega) \tag{2.24}$$

defined by

$$\chi_\omega z = z|_\omega, \tag{2.25}$$

where $z|_\omega$ is the restriction of z to ω. For simplification, along this section we denote $\gamma = \chi_\omega$. Then, we introduce the following definition [80]:

Definition 2.5 System (2.17)–(2.18) is said to be regionally observable on ω (or ω-observable) if

$$\operatorname{Im}\big(\gamma K^\star\big) = L^2(\omega). \tag{2.26}$$

System (2.17)–(2.18) is said to be weakly regionally observable on ω (or weakly ω-observable) if

$$\overline{\operatorname{Im}\big(\gamma K^\star\big)} = L^2(\omega). \tag{2.27}$$

From the above definition we deduce the following characterization [56]:

Lemma 2.6 *System* (2.17)–(2.18) *is exactly ω-observable if there exists $\omega > 0$ such that, for all $z_0 \in L^2(\omega)$,*

$$\|\gamma y_0\|_{L^2(\omega)} \le \nu \big\| K\gamma^\star y_0 \big\|_{L^2(]0,T[;\mathbb{R}^q)}. \tag{2.28}$$

2.4 Parameter Identification and Optimal Experiment Design

2.4.1 System Definition

Due to the nature of the considered parameter identification problem, the abstract operator-theoretic formalism used in (2.1) to define the dynamics of a DPS is not convenient. In this section, the following PDE-based general definitions are given. Consider a DPS described by n partial differential equations of the following form:

$$\begin{aligned}&\mathcal{F}_1(x,t)\frac{\partial y(x,t)}{\partial t} = \mathcal{F}_2\big(x,t,y(x,t),\nabla y(x,t),\nabla^2 y(x,t),\theta\big),\\ &(x,t)\in\Omega\times T\subset\mathbb{R}^{d+1}\end{aligned} \tag{2.29}$$

with initial and boundary conditions

$$\mathcal{B}(x,t,y) = 0, \quad (x,t)\in\partial\Omega\times T, \tag{2.30}$$

$$\mathcal{N}(x,t,y) = 0, \quad (x,t)\in\Omega\times\{0\}, \tag{2.31}$$

where

- $\Omega \subset \mathbb{R}^n$ is a bounded spatial domain with sufficiently smooth boundary $\Gamma = \partial\Omega$,
- t is the time instant,
- $T = [0, t_f]$ is a bounded time interval called observation interval,
- $x = (x_1, x_2, \dots, x_d)$ is a spatial point belonging to $\overline{\Omega} = \Omega \cup \Gamma$,
- $y = (y_1(x,t), y_2(x,t), \dots, y_n(x,t))$ stands for the state vector, and
- $\mathcal{F}_1$, $\mathcal{F}_2$, $\mathcal{B}$, and $\mathcal{N}$ are some known functions.

We assume that the system of equations (2.29)–(2.31) has a unique solution that is sufficiently regular. We can see that (2.29)–(2.31) contains an unknown set of parameters θ whose values belong to an admissible parameter space Θ_{ad}. Even though Θ_{ad} can have different forms, we make an assumption that the parameters are constant ($\theta \in \mathbb{R}^m$). The set of unknown parameters θ has to be determined based on observations made by N mobile pointwise sensors over the observation horizon T. We define $x_j : T \to \Omega_{\mathrm{ad}}$ as the trajectory of the jth mobile sensor, with $\Omega_{\mathrm{ad}} \subset \Omega$ being the region where measurements can be made. The observations are assumed to be of the form

$$z^j(t) = y\bigl(x^j(t), t\bigr) + \varepsilon\bigl(x^j(t), t\bigr), \quad t \in T,\ j = 1, \dots, N. \tag{2.32}$$

The collection of measurements $z(t) = [z^1(t), z^2(t), \dots, z^N(t)]^T$ is the N-dimensional observation vector, and ε represents the measurement noise assumed to be white, zero-mean, Gaussian, and spatial uncorrelated with the following statistics:

$$\mathrm{e}\bigl\{\varepsilon\bigl(x^j(t), t\bigr)\varepsilon\bigl(x^i(t'), t'\bigr)\bigr\} = \sigma^2 \delta_{ji}\delta(t - \tau), \tag{2.33}$$

where σ^2 stands for the standard deviation of the measurement noise, and δ_{ij} and $\delta(\cdot)$ are the Kronecker and Dirac delta functions, respectively.

2.4.2 Parameter Identification

According to this setup, the parameter identification problem is defined as follows. Given the model (2.29)–(2.31) and the measurements $z(t)$ along the trajectories (x^j), $j = 1, \dots, N$, obtain an estimation $\hat{\theta} \in \Theta_{\mathrm{ad}}$ minimizing the following weighted least-squares criterion as in [18] and [109]:

$$\mathfrak{J}(\theta) = \frac{1}{2}\int_0^T \bigl\| z(t) - \hat{y}(\mathbf{x}, t; \theta)\bigr\|^2 \, \mathrm{d}t, \tag{2.34}$$

where $\hat{y}(x, t; \theta)$ stands for the solution to (2.29)–(2.31) corresponding to a given set of parameters θ, and $\|\cdot\|$ stands for the Euclidean norm.

The estimated values of the parameters $\hat{\theta}$ are influenced by the sensors' trajectories $x^j(t)$, and our objective is to obtain the best estimates of the system parameters. Therefore, deciding on the trajectory based on a quantitative measure related to the expected accuracy of the parameter estimates to be obtained from the data collected seems to be practically logical.

2.4.3 Sensor Location Problem

The Fisher information matrix (FIM) [119, 132] is a well-known performance measure when looking for best measurements and is widely used in optimum experimental design theory for lumped systems. Its inverse constitutes an approximation of the covariance matrix for the estimate of θ [16, 60, 162]. Let us give the following definition of the experiment:

$$s(t) = \big(x^1(t), \dots, x^N(t)\big) \quad \forall t \in T, \tag{2.35}$$

and let $n = \dim(s(t))$. Under such conditions, the FIM can be written as [118]

$$M(s) = \sum_{j=1}^{N} \int_0^T g\big(x^j(t), t\big) g^T\big(x^j(t), t\big)\,\mathrm{d}t, \tag{2.36}$$

where $g(\mathbf{x}, t) = \nabla_\theta y(x, t; \theta)|_{\theta=\theta^0}$ is the vector made of the sensitivity coefficients, θ^0 being the previous estimate of the unknown parameter vector θ [146, 147].

By choosing s such that it minimizes a scalar function $\Psi(\cdot)$ of the FIM, one can determine the optimal mobile sensor trajectories. There are many candidates for such a function [16, 60, 162]:

- The A-optimality criterion suppresses the variance of the estimates

$$\Psi(M) = \operatorname{trace}\big(M^{-1}\big). \tag{2.37}$$

- The D-optimality criterion minimizes the volume of the confidence ellipsoid for the parameters

$$\Psi(M) = -\log\det(M). \tag{2.38}$$

- The E-optimality criterion minimizes the largest width of the confidence ellipsoid

$$\Psi(M) = \lambda_{\max}\big(M^{-1}\big). \tag{2.39}$$

- The sensitivity criterion' minimization increases the sensitivity of the outputs with respect to parameter changes

$$\Psi(M) = -\operatorname{trace}(M). \tag{2.40}$$

2.4.4 Sensor Clustering Phenomenon

The assumption on the spatial uncorrelation of the measurement noise can create a clustering of the sensors, which can be problematic in practice. We use an example from [147] to illustrate the sensor clustering problem.

Example 2.1 Consider the following parabolic partial differential equation:

$$\frac{\partial y(x,t)}{\partial t} = \theta_1 \frac{\partial^2 y(x,t)}{\partial x^2}, \quad x \in (0, \pi),\ t \in (0, 1),$$

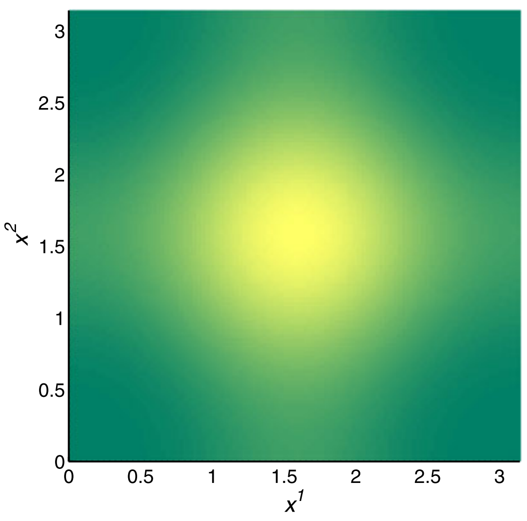


Fig. 2.4 Contour plot of $\det(M(x^1, x^2))$ versus the sensors' locations ($\theta_1 = 0.1$ and $\theta_2 = 1$)

with boundary and initial conditions

$$y(0,t) = y(\pi,t) = 0, \quad t \in (0,1),$$

$$y(x,0) = \theta_2 \sin(x), \quad x \in (0,\pi).$$

The two parameters θ_1 and θ_2 are assumed to be constant but unknown. In addition, we assume that the measurements are taken by two static sensors whose locations are decided by maximizing the determinant of the FIM. The analytical solution of the PDE can be easily obtained as

$$y(x,t) = \theta_2 \exp(-\theta_1 t) \sin(x).$$

The assumption is made that the signal noise statistic $\sigma = 1$ does not change the optimal location of the sensors. The determinant of the matrix is given by

$$\det\big(M\big(x^1, x^2\big)\big) = \frac{\theta_2^2}{16\theta_1^4}\big(-4\theta_1^2 \exp(-2\theta_1) - 2\exp(-2\theta_1) + \exp(-4\theta_1) + 1\big)$$
$$\times \big(2 - \cos^2(x_1) - \cos^2(x_2)\big)^2.$$

The results are shown in Fig. 2.4, and one quick observation allows one to determine that the best location for both sensors is at the center of the interval $(0,\pi)$.

2.4.5 Dependence of the Solution on Initial Parameter Estimates

Another serious issue in the FIM framework of optimal measurements for parameter estimation of DPS is the dependence of the solution on the initial guess on parameters. We illustrate the problem using an example from [111].

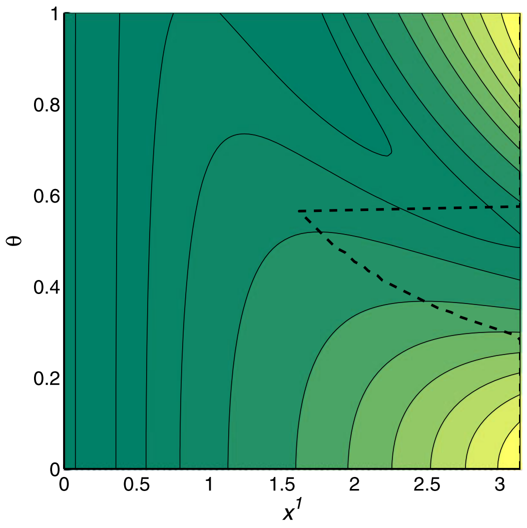


Fig. 2.5 Contour plot of $M(x^1;\theta)$

Example 2.2 Consider the following hyperbolic partial differential equation:

$$\frac{\partial^2 y(x,t)}{\partial t^2} = \theta \frac{\partial^2 y(x,t)}{\partial x^2}, \quad x \in (0,\pi),\ t \in (0,\pi),$$

with boundary and initial conditions

$$y(0,t) = \frac{1}{4}\cos(t), \qquad y(\pi,t) = \sin(\pi\theta)\sin(t) + \frac{1}{4}\cos(\pi\theta)\cos(t), \quad t \in (0,\pi),$$

$$y(x,0) = \frac{1}{4}\cos\theta x, \qquad \left.\frac{\partial y(x,t)}{\partial t}\right|_{t=0} = \sin(\theta x), \quad x \in (0,\pi).$$

The parameter θ is assumed to be constant and unknown. In addition, we assume that the measurements are taken by one static sensor located at $x^1 \in (0,\pi)$. The analytical solution of the PDE can be easily obtained and is given as

$$\begin{aligned} M\big(x^1\big) &= \int_0^\pi \left(\frac{\partial y(x^1,t;\theta)}{\partial\theta}\right)^2 \mathrm{d}t \\ &= \frac{1}{2}x^2\pi\cos^2(\theta x) + \frac{1}{32}x^2\pi\sin(\theta x). \end{aligned}$$

The results are shown in Fig. 2.5 (the optimal location of the sensor is represented by a dashed line), and it is easy to observe that the optimal sensor location depends on the value of θ.

The dependence of the optimal location on θ is very problematic; however, some techniques called “robust designs” have been developed to minimize or elude the influence [132, 162]. We propose similar methodologies in Chap. 5.

2.5 Chapter Summary

In this chapter, we gave very important definitions in the framework of DPSs. We defined the dynamic equations of the system, the mathematical descriptions of a sensor and an actuator. From those definitions we introduced the concepts of regional controllability and observability. Then, we described the dynamics of the system in an appropriate way for the FIM framework of optimal sensor location for parameter estimation. We gave the definitions of the parameter estimation and optimal sensor location. Finally, we discussed two of the important issues of the FIM framework: the sensor clustering phenomenon and the dependence of the solution on initial parameter estimates.

Chapter 3
Optimal Heterogeneous Mobile Sensing for Parameter Estimation of Distributed Parameter Systems

3.1 Introduction

States in distributed parameter systems (DPSs), i.e., systems described by partial differential equations (PDEs), vary both spatially and temporally, but it is generally impossible to measure them over the whole spatial domain. Consequently, we are faced with the design problem of how to locate a limited number of measurement sensors so as to obtain as much information as possible about the process at hand. The location of sensors is not necessarily dictated by physical considerations or by intuition, and, therefore, some systematic approaches should be developed in order to reduce the cost of instrumentation and to increase the efficiency of parameter estimation.

Although it is well known that the estimation accuracy of DPS parameters depends significantly on the choice of sensor locations, there have been relatively few contributions to the optimal experimental design for those systems. The importance of sensor planning has been recognized in many application domains, e.g., regarding air quality monitoring systems, groundwater-resources management, recovery of valuable minerals and hydrocarbon, model calibration in meteorology and oceanography, chemical engineering, hazardous environments, and smart materials [19, 45, 49, 84, 85, 104, 105, 116, 132, 147]. Over the past years, increasingly thorough research on the development of strategies for efficient sensor placement has been observed (for reviews, see papers [92, 158] and comprehensive monographs [147, 150]). Nevertheless, much still has to be done in this respect, particularly in light of recent advances in wireless sensor networks [31, 41, 48, 50–52, 81, 128, 174].

Nowadays, mobile platforms for sensors are available (mobile robots or unmanned air vehicles) that offer an appealing alternative to common stationary sensors with fixed positions in space [31, 41, 48, 107, 128]. The complexity of the resulting design problem is expected to be compensated by a number of benefits. Specifically, sensors are not assigned to fixed positions that are optimal only on the average, but are capable of tracking points that provide at a given time instant the best information about the parameters to be identified. Consequently, by actively reconfiguring a sensor system, we can expect the minimal value of an adopted design criterion to be lower than the one for the stationary case. Areas of direct application of such mobile sensing techniques include air pollutant measurements in the environment obtained from monitoring cars moving in an urban area, or atmospheric variables acquired using instruments carried in a satellite or aircraft [103]. Low-cost mobile platforms with wireless communication capabilities for sensor networks are now available. They get cheaper and cheaper, and more advanced ones are under development. With a group of such autonomous vehicles equipped with sensors, we can enhance the performance of the measurements.

The number of publications on optimized mobile observations for parameter estimation is limited. In a seminal article [119], Rafajłowicz considers the D-optimality criterion and seeks an optimal time-dependent measure, rather than the trajectories themselves. On the other hand, Uciński [146, 147, 156], apart from generalizations of Rafajłowicz's results, develops some computational algorithms based on the Fisher information matrix. He reduces the problem to a state-constrained optimal-control one for which solutions are obtained via the methods of successive linearizations that are capable of handling various constraints imposed on sensor motions. In turn, in [153] Uciński and Chen attempted to properly formulate and solve the time-optimal problem for moving sensors that observe the state of a DPS so as to estimate some of its parameters.

In the literature on mobile sensors, it is most often assumed that the optimal measurement problem occurs in the design of trajectories of a given number of identical sensors. In this chapter, we formulate it in a quite different manner. First of all, apart from sensor controls and initial positions, the number of sensors constitutes an additional design variable. Additionally, we can allow for different levels of measurement accuracies for individual sensors, which are quantified by weights steering the corresponding measurement variances. This leads to a much more general formulation that most often produces an uneven allocation of experimental efforts between different sensors. The corresponding solutions could then be implemented on a sensor network with heterogeneous mobile nodes. It turns out that these solutions can be determined using convex optimization tools commonly used in optimum experimental design [16, 62, 162]. As a result, much better accuracies of the parameter estimates can be achieved.

3.2 Optimal Sensor Location Problem

Let $\Omega \subset \mathbb{R}^n$ be a bounded spatial domain with sufficiently smooth boundary Γ, and let $T = (0, t_f]$ be a bounded time interval. Consider a distributed parameter system (DPS) whose scalar state at a spatial point $\boldsymbol{x} \in \bar{\Omega} \subset \mathbb{R}^n$ and time instant $t \in T$ is denoted by $y(\boldsymbol{x}, t)$. Mathematically, the system state is governed by the partial differential equation

$$\frac{\partial y}{\partial t} = \mathcal{F}(\boldsymbol{x}, t, y, \boldsymbol{\theta}) \quad \text{in } \Omega \times T, \tag{3.1}$$

where $\mathcal{F}$ is a well-posed, possibly nonlinear, differential operator that involves first- and second-order spatial derivatives and may include terms accounting for forcing inputs specified a priori. The PDE (3.1) has the following appropriate boundary and initial conditions:

$$\mathcal{B}(\boldsymbol{x}, t, y, \boldsymbol{\theta}) = 0 \quad \text{on } \varGamma \times T, \tag{3.2}$$

$$y = y_0 \quad \text{in } \varOmega \times \{t = 0\}, \tag{3.3}$$

respectively, where $\mathcal{B}$ is an operator acting on the boundary $\varGamma$, and $y_0 = y_0(\boldsymbol{x})$ is a given function. Conditions (3.2) and (3.3) complement (3.1) so that the existence of a sufficiently smooth and unique solution is guaranteed. We assume that the forms of $\mathcal{F}$ and $\mathcal{B}$ are given explicitly up to an m-dimensional vector of unknown constant parameters $\boldsymbol{\theta}$, which must be estimated using observations of the system. The implicit dependence of the state y on the parameter vector $\boldsymbol{\theta}$ will be reflected by the notation $y(\boldsymbol{x}, t; \boldsymbol{\theta})$.

We assume that the vector $\boldsymbol{\theta} \in \mathbb{R}^m$ is to be estimated from measurements made by N moving sensors over the observation horizon T. We call $\boldsymbol{x}_s^j : T \to \varOmega_{\text{ad}}$ the trajectory of the jth sensor, where $\varOmega_{\text{ad}} \subset \varOmega \cup \varGamma$ is a compact set representing the area where the mobile sensing measurements can be made. The observations are of the form

$$z^j(t) = y\big(\boldsymbol{x}_s^j(t), t\big) + \varepsilon\big(\boldsymbol{x}_s^j(t), t\big), \quad t \in T,\ j = 1, \dots, N, \tag{3.4}$$

where ε constitutes the measurement noise, which is assumed to be zero-mean, Gaussian, spatial uncorrelated, and white [14, 109, 118], i.e.,

$$\mathrm{e}\big\{\varepsilon\big(\boldsymbol{x}_s^j(t), t\big)\varepsilon\big(\boldsymbol{x}_s^i(\tau), \tau\big)\big\} = \delta_{ji}\delta(t - \tau)\frac{\sigma^2}{p_j}, \tag{3.5}$$

where σ^2/p_j defines the intensity of the noise, σ^2 is a constant, p_j stands for a positive scaling factor, and δ_{ij} and $\delta(\cdot)$ stand for the Kronecker and Dirac delta functions, respectively. Although white noise is a physically impossible process, it constitutes a reasonable approximation to a disturbance whose adjacent samples are uncorrelated at all time instants for which the time increment exceeds some value that is small compared with the time constants of the DPS. The white-noise assumption is consistent with most of the literature on the subject.

Note that instead of several mobile sensors whose accuracies are characterized by the equal variance σ^2, we use sensors for which the variance of measurement errors is σ^2/p_j. This means that a large weight p_j indicates that the jth sensor guarantees more precise measurements than sensors with lower weight values. With no loss of generality, we assume that the weights p_j satisfy the following normalization condition:

$$\sum_{j=1}^{N} p_j = 1, \qquad p_j \geq 0, \quad j = 1, \dots, N, \tag{3.6}$$

i.e., they belong to the probability simplex.

In the presented framework, the parameter identification problem is usually formulated as follows: Given the model (3.1)–(3.3) and the outcomes of the measurements z^j along the trajectories $\boldsymbol{x}_s^j$, $j = 1, \dots, N$, determine an estimate $\hat{\boldsymbol{\theta}} \in \Theta_{\mathrm{ad}}$ (Θ_{ad} being the set of admissible parameters) that minimizes the generalized output least-squares fit-to-data functional given by [18, 109]

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\vartheta} \in \Theta_{\mathrm{ad}}}{\arg\min} \sum_{j=1}^{N} p_j \int_T \big[z^j(t) - y\big(\boldsymbol{x}_s^j(t), t; \boldsymbol{\vartheta}\big)\big]^2 \,\mathrm{d}t, \tag{3.7}$$

where y now solves (3.1)–(3.3) for $\boldsymbol{\theta}$ replaced by $\boldsymbol{\vartheta}$.

We feel, intuitively, that the parameter estimate $\hat{\boldsymbol{\theta}}$ depends on the number of sensors N, the trajectories $\boldsymbol{x}_s^j$, and the associated weights p_j since the right-hand side of (3.7) does. This fact suggests that we may attempt to select these design variables so as to produce best estimates of the system parameters after performing the actual experiment. Note that the weights p_j can be interpreted here as sensor costs, which are inversely proportional to the variances of the corresponding measurement errors introduced by them. The weights must sum up to unity, which means that our budget on the experiment is fixed. Then, the problem is how to spend it, i.e., how many and how accurate sensors to buy so as to get the most accurate parameter estimates while assuming that their trajectories are also going to be optimized.

To form a basis for the comparison of different design settings, a quantitative measure of the "goodness" of particular settings is required. A logical approach is to choose a measure related to the expected accuracy of the parameter estimates to be obtained from the data collected (note that the design is to be performed offline, before taking any measurements). Such a measure is usually based on the concept of the Fisher information matrix (FIM) [119, 132], which is widely used in optimum experimental design theory for lumped systems [16, 62, 162]. When the time horizon is large, the nonlinearity of the model with respect to its parameters is mild, and the measurement errors are independently distributed and have small magnitudes, the inverse of the FIM constitutes a good approximation of the covariance matrix for the estimate of $\boldsymbol{\theta}$ [16, 62, 162].

The FIM has the following representation [118, 147]:

$$\boldsymbol{M} = \sum_{j=1}^{N} p_j \int_T \boldsymbol{g}\big(\boldsymbol{x}_s^j(t), t\big) \boldsymbol{g}^{\mathsf{T}}\big(\boldsymbol{x}_s^j(t), t\big) \,\mathrm{d}t, \tag{3.8}$$

where

$$\boldsymbol{g}(\boldsymbol{x}, t) = \nabla_{\boldsymbol{\vartheta}} y(\boldsymbol{x}, t; \boldsymbol{\vartheta})|_{\boldsymbol{\vartheta}=\boldsymbol{\theta}^0} \tag{3.9}$$

denotes the vector of the so-called sensitivity coefficients, $\boldsymbol{\theta}^0$ being a prior estimate to the unknown parameter vector $\boldsymbol{\theta}$ [146, 147].

The sought optimal design settings can be found by maximizing some scalar function Ψ of the information matrix. The introduction of the design criterion permits one to cast the sensor location problem as an optimization problem, and the criterion itself can be treated as a measure of the information content of the observations. Several choices exist for such a function [16, 62, 162], and the most popular one is the D-optimality criterion

$$\Psi[\boldsymbol{M}] = -\log\det(\boldsymbol{M}). \tag{3.10}$$

Its use yields the minimal volume of the confidence ellipsoid for the estimates. In what follows, we shall restrict our attention to this optimality criterion.

3.3 Mobile Sensor Model

3.3.1 Node Dynamics

Although measurement accuracies may vary from sensor to sensor, we assume that all sensors are conveyed by identical vehicles whose motions are described by

$$\dot{\boldsymbol{x}}_s^j(t) = \boldsymbol{f}\big(\boldsymbol{x}_s^j(t), \boldsymbol{u}_s^j(t)\big) \quad \text{a.e. on } T, \quad \boldsymbol{x}_s^j(0) = \boldsymbol{x}_{s0}^j, \tag{3.11}$$

where a given function $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^r \to \mathbb{R}^n$ is required to be continuously differentiable, $\boldsymbol{x}_{s0}^j \in \mathbb{R}^n$ defines an initial sensor configuration, and $\boldsymbol{u}_s^j : T \to \mathbb{R}^r$ is a measurable control function that satisfies

$$\boldsymbol{u}_{sl} \le \boldsymbol{u}_s^j(t) \le \boldsymbol{u}_{su} \quad \text{a.e. on } T \tag{3.12}$$

for some constant bound vectors $\boldsymbol{u}_{sl}$ and $\boldsymbol{u}_{su}$, $j = 1, \ldots, N$.

For each $j = 1, \ldots, N$, given any initial position $\boldsymbol{x}_{s0}^j$ and any control function, there is a unique absolutely continuous function $\boldsymbol{x}_s^j : T \to \mathbb{R}^n$ that satisfies (3.11) a.e. on T. In what follows, we will call it the state trajectory corresponding to $\boldsymbol{x}_{s0}^j$ and $\boldsymbol{u}_s^j$.

3.3.2 Pathwise State Constraints

In reality, some restrictions on the motions are inevitably imposed. First of all, all sensors should stay within the admissible region Ω_{ad} where measurements are allowed. We assume that it is a compact set defined as follows:

$$\Omega_{\text{ad}} = \big\{\boldsymbol{x} \in \Omega \cup \Gamma \mid b_i(\boldsymbol{x}) \le 0,\ i = 1, \ldots, I\big\}, \tag{3.13}$$

where b_is are given continuously differentiable functions. Accordingly, the conditions

$$b_i\big(\boldsymbol{x}_s^j(t)\big) \le 0 \quad \forall t \in T \tag{3.14}$$

must be fulfilled, where $1 \le i \le I$ and $1 \le j \le N$.

3.3.3 Parameterization of Vehicle Controls

From now on, we make the assumption that the controls of the available vehicles can be represented in the parametric form

$$\boldsymbol{u}_s^j(t) = \boldsymbol{\eta}\big(t, \boldsymbol{a}^j\big), \quad t \in T, \tag{3.15}$$

where $\boldsymbol{\eta}$ denotes a given function such that $\boldsymbol{\eta}(\,\cdot\,, \boldsymbol{a}^j)$ is continuous for each fixed $\boldsymbol{a}^j$, and $\boldsymbol{\eta}(t,\,\cdot\,)$ is continuous for each fixed t, the constant parameter vector $\boldsymbol{a}^j$ ranging over a compact set $A \subset \mathbb{R}^q$. An exemplary parameterization is using B-splines as employed in numerous optimal control solvers, e.g., RIOTS_ 95 [126] to be described in Appendix B.

Based on a specific parameterization, we can define the mapping $\boldsymbol{\chi}$ which assigns every $\boldsymbol{c}^j = (\boldsymbol{x}_{s0}^j, \boldsymbol{a}^j) \in \Omega_{\mathrm{ad}} \times A$ a trajectory $\boldsymbol{x}_s^j = \boldsymbol{\chi}(\boldsymbol{c}^j)$ through solving (3.11) for the initial position $\boldsymbol{x}_{s0}^j$ and control defined by (3.15).

Since only the controls and trajectories satisfying the imposed constraints are of our interest, we introduce the set

$$C = \big\{\boldsymbol{c} = (\boldsymbol{x}_{s0}, \boldsymbol{a}) \in A \times \Omega_{\mathrm{ad}} : \boldsymbol{\eta}(\,\cdot\,, \boldsymbol{a}) \text{ satisfies (3.12)}, \;\; \boldsymbol{\chi}(\boldsymbol{c}) \text{ satisfies (3.14)}\big\} \tag{3.16}$$

and assume that it is nonempty. A trivial verification shows that C is also compact.

Given N sensors, we thus obtain trajectories $\boldsymbol{x}_s^j$ corresponding to vectors $\boldsymbol{c}^j \in \mathbb{R}^{n+q}$, $j = 1, \dots, N$. The FIM can then be rewritten as

$$\boldsymbol{M}(\xi_N) = \sum_{j=1}^{N} p_j \int_T \boldsymbol{g}\big(\boldsymbol{x}(t), t\big)\boldsymbol{g}^{\mathsf{T}}\big(\boldsymbol{x}(t), t\big)\big|_{\boldsymbol{x}=\boldsymbol{\chi}(\boldsymbol{c}^j)}\, \mathrm{d}t, \tag{3.17}$$

where, for simplicity of notation, we represent the decision variables as the following table:

$$\xi_N = \begin{Bmatrix} \boldsymbol{c}^1, & \boldsymbol{c}^2, & \dots, & \boldsymbol{c}^N \\ p_1, & p_2, & \dots, & p_N \end{Bmatrix}. \tag{3.18}$$

Applying the terminology of optimum experimental design, we call this table a *discrete design*, while $\boldsymbol{c}^1, \dots, \boldsymbol{c}^N$ are termed the *support points*, and $p_1, \dots, p_N$ are referred to as the corresponding *weights*.

Observe that a design ξ_N can be interpreted as a discrete probability distribution on a finite subset of C, cf. (3.6). As is standard in optimum experimental design theory [61], we can extend this idea and regard a design as a probability measure ξ for all Borel sets of C, including single points. With such a modification, we can define the FIM analogous to (3.17) for the design ξ:

$$\boldsymbol{M}(\xi) = \int_C \boldsymbol{\Upsilon}(\boldsymbol{c})\, \xi(\mathrm{d}\boldsymbol{c}), \tag{3.19}$$

where

$$\boldsymbol{\Upsilon}(\boldsymbol{c}) = \int_T \boldsymbol{g}\big(\boldsymbol{x}(t), t\big)\boldsymbol{g}^{\mathsf{T}}\big(\boldsymbol{x}(t), t\big)\big|_{\boldsymbol{x}=\boldsymbol{\chi}(\boldsymbol{c})}\,\mathrm{d}t. \tag{3.20}$$

The integration in (3.19) is to be understood in the Lebesgue–Stieltjes sense. This leads to the so-called *continuous* designs that constitute the basis of the modern theory of optimal experiments and originate in seminal works by Kiefer and Wolfowitz [87]. It turns out that such an approach drastically simplifies the design, and the remainder of the chapter is devoted to this design issue.

3.4 Characterization of Optimal Solutions

For clarity, we adopt the following notational conventions. Here and subsequently, we will use the symbol $\Xi(C)$ to denote the set of all probability measures on C. Let us also introduce the notation $\mathfrak{M}(C)$ for the set of all admissible information matrices, i.e.,

$$\mathfrak{M}(C) = \big\{\boldsymbol{M}(\xi) : \xi \in \Xi(C)\big\}. \tag{3.21}$$

Then we may redefine an optimal design as a solution to the following optimization problem:

$$\xi^\star = \arg\max_{\xi \in \Xi(C)} \Psi\big[\boldsymbol{M}(\xi)\big]. \tag{3.22}$$

The theoretical results presented in this section constitute straightforward adaptations of their counterparts of Chap. 3 in [147]. We begin with certain convexity and representation properties of $M(\xi)$.

Lemma 3.1 *For any $\xi \in \Xi(C)$, the information matrix $\boldsymbol{M}(\xi)$ is symmetric and nonnegative definite.*

Lemma 3.2 *$\mathfrak{M}(C)$ is compact and convex.*

Lemma 3.3 *For any $\boldsymbol{M}_0 \in \mathfrak{M}(C)$, there always exists a purely discrete design ξ of the form (3.18) with no more than $m(m+1)/2+1$ support points such that $\boldsymbol{M}(\xi) = \boldsymbol{M}_0$. If $\boldsymbol{M}_0$ lies on the boundary of $\mathfrak{M}(C)$, then the number of support points is less than or equal to $m(m+1)/2$.*

The above lemma justifies that we can restrict our attention only to discrete designs with a limited number of supporting points, so the introduction of continuous designs being probability measures for all Borel sets of C is feasible technically. In this way, it greatly simplifies the solution process.

The next result provides a characterization of the optimal designs.

Theorem 3.4 *We have the following properties*:

(i) *An optimal design exists that is discrete and comprises no more than* $m(m+1)/2$ *support points* (*i.e., one less than predicted by Lemma* 3.3).
(ii) *The set of optimal designs is convex.*
(iii) *A design* $\xi^\star$ *is optimal if and only if*

$$\max_{\boldsymbol{c}\in C}\varphi\bigl(\boldsymbol{c},\xi^\star\bigr)=m, \tag{3.23}$$

where

$$\varphi(\boldsymbol{c},\xi)=\operatorname{trace}\bigl[\boldsymbol{M}^{-1}(\xi)\boldsymbol{\Upsilon}(\boldsymbol{c})\bigr]. \tag{3.24}$$

(iv) *For any purely discrete optimal design* $\xi^\star$, *the function* $\varphi(\,\cdot\,,\xi^\star)$ *has value zero at all support points.*

It is now clear that the function φ is of paramount importance in our considerations, as it determines the location of the support points in the optimal design $\xi^\star$ (they are among its points of global maximum). Moreover, given any design ξ, it indicates points at which a new observation contributes to the greatest extent. Indeed, adding a new observation at a single point $\boldsymbol{c}^+$ amounts to constructing a new design

$$\xi^+=(1-\lambda)\xi+\lambda\xi_{\boldsymbol{c}^+} \tag{3.25}$$

for some $\lambda\in(0,1)$. If λ is sufficiently small, then it may be concluded that

$$\Psi\bigl[\boldsymbol{M}\bigl(\xi^+\bigr)\bigr]-\Psi\bigl[\boldsymbol{M}(\xi)\bigr]\approx\lambda\varphi\bigl(\boldsymbol{c}^+,\xi\bigr), \tag{3.26}$$

i.e., the resulting increase in the criterion value is approximately equal to $\lambda\varphi(\boldsymbol{c}^+,\xi)$.

Analytical determination of optimal designs is possible only in simple situations, and for general systems, it is usually the case that some iterative design procedure will be required. The next theorem, called the *equivalence theorem*, is useful in checking for optimality of designs [117].

Theorem 3.5 *The following characterizations of an optimal design* $\xi^\star$ *are equivalent in the sense that each implies the other two*:

(i) *the design* $\xi^\star$ *maximizes* $\Psi[\boldsymbol{M}(\xi)]$,
(ii) *the design* $\xi^\star$ *minimizes* $\max_{\boldsymbol{c}\in C}\varphi(\boldsymbol{c},\xi)$, *and*
(iii) $\max_{\boldsymbol{c}\in C}\varphi(\boldsymbol{c},\xi^\star)=m$.

All the designs satisfying (i)–(iii) *and their convex combinations have the same information matrix* $\boldsymbol{M}(\xi^\star)$.

The above results provide us with tests for the optimality of designs. In particular:

1. If the sensitivity function $\varphi(\boldsymbol{c},\xi)$ is less than or equal to m for all $\boldsymbol{c}\in C$, then ξ is optimal.
2. If the sensitivity function $\varphi(\boldsymbol{c},\xi)$ exceeds m, then ξ is not optimal.

An interesting aspect of these results is that in addition to revealing striking minimax properties of optimal designs, they also provide sequential numerical design algorithms. That is, suppose that we have an arbitrary (nonoptimal) design ξ_k obtained after k iteration steps and let $\varphi(\,\cdot\,, \xi_k)$ attain its maximum (necessarily $> m$) at $\boldsymbol{c} = \boldsymbol{c}_k^0$. Then, the design

$$\xi_{k+1} = (1 - \lambda_k)\xi_k + \lambda_k \xi_{\boldsymbol{c}_k^0} \tag{3.27}$$

(here $\xi_{\boldsymbol{c}_k^0}$ stands for the unit-weight design concentrated at $\boldsymbol{c}_k^0$) leads to an increase in the value of $\Psi[\boldsymbol{M}(\xi_{k+1})]$ for a suitably small λ_k. This follows since the derivative with respect to λ_k is positive, i.e.,

$$\frac{\partial}{\partial \lambda_k} \Psi\big[\boldsymbol{M}(\xi_{k+1})\big]\big|_{\lambda_k = 0^+} = m - \varphi\big(\boldsymbol{c}_k^0, \xi_k\big) > 0. \tag{3.28}$$

Therefore, the procedure in using the above outlined gradient method can be briefly summarized as follows [57, 62, 120, 162]:

Step 1. Guess a discrete nondegenerate starting design measure ξ_0 (we must have $\det(\boldsymbol{M}(\xi_0)) \neq 0$). Choose some positive tolerance $\epsilon \ll 1$. Set $k = 0$.
Step 2. Determine $\boldsymbol{c}_k^0 = \arg\max_{\boldsymbol{c} \in C} \varphi(\boldsymbol{c}, \xi_k)$. If $\varphi(\boldsymbol{c}_k^0, \xi_k) < m + \epsilon$, then *STOP*.
Step 3. For an appropriate value of $0 < \lambda_k < 1$, set

$$\xi_{k+1} = (1 - \lambda_k)\xi_k + \lambda_k \xi_{\boldsymbol{c}_k^0},$$

increase k by one, and go to Step 2.

In the same way as for the classical first-order algorithms commonly used in optimum experimental designs for many years, it can be shown that the above algorithm converges to an optimal design, provided that the sequence $\{\lambda_k\}$ is suitably chosen. For example, the choices that satisfy one of the conditions below will ensure the convergence:

(i) $\lim_{k\to\infty} \lambda_k = 0$, $\sum_{k=0}^{\infty} \lambda_k = \infty$ (Wynn's algorithm),
(ii) $\lambda_k = \arg\min_{\lambda} \Psi[(1-\lambda)\boldsymbol{M}(\xi_k) + \lambda \boldsymbol{M}(\xi_{\boldsymbol{c}_k^0})]$ (Fedorov's algorithm).

Computationally, Step 2 is of crucial significance, but at the same time it is the most time-consuming step in the algorithm. Complications arise, among other things, due to the necessity of calculating a global maximum of $\varphi(\,\cdot\,, \xi_k)$, which is usually multimodal (getting stuck in one of local maxima leads to premature termination of the algorithm). Therefore, while implementing this part of the computational procedure, an effective global optimizer seems to be essential.

3.5 Optimal Control Formulation of the Search for the Candidate Support Point

Step 2 of the Wynn–Fedorov algorithm in the previous section is necessary in the determination of $\arg\max_{\boldsymbol{c}\in C} \varphi(\boldsymbol{c}, \xi_k)$. This formulation can be interpreted as a finite-dimensional approximation to the following optimization problem:

Find the pair $(\boldsymbol{x}_{s0}, \boldsymbol{u}_s)$ that maximizes

$$\begin{aligned} J(\boldsymbol{x}_{s0}, \boldsymbol{u}_s) &= \operatorname{trace}\left[\boldsymbol{M}^{-1}(\xi^k)\int_T \boldsymbol{g}\big(\boldsymbol{x}(t), t\big)\boldsymbol{g}^{\mathsf{T}}\big(\boldsymbol{x}(t), t\big)\,\mathrm{d}t\right] \\ &= \int_T \boldsymbol{g}^{\mathsf{T}}\big(\boldsymbol{x}(t), t\big)\boldsymbol{M}^{-1}(\xi^k)\boldsymbol{g}\big(\boldsymbol{x}(t), t\big)\,\mathrm{d}t \end{aligned} \tag{3.29}$$

over the set of feasible pairs

$$\begin{aligned} \mathcal{P} = \big\{(\boldsymbol{x}_{s0}, \boldsymbol{u}_s)\,|u_s : T \to \mathbb{R}^r \text{ is measurable,} \\ \boldsymbol{u}_{sl} \le \boldsymbol{u}_s(t) \le \boldsymbol{u}_{su} \text{ a.e. on } T,\ \boldsymbol{x}_{s0} \in \Omega_{\mathrm{ad}}\big\}, \end{aligned}$$

subject to the pathwise state inequality constraints (3.14).

Evidently, its high nonlinearity excludes any possibility of finding closed-form formulas for its solution. Accordingly, we must resort to numerical techniques. A number of possibilities exist in this respect [73, 115], but since this problem is already in canonical form, we can solve it using one of the existing packages for numerically solving dynamic optimization problems, such as RIOTS_95 [126], DIRCOL [161], or MISER [82]. In our implementation, we employed the first of them, i.e., RIOTS_95, which is designed as a MATLAB toolbox written mostly in C and running under Windows 98/2000/XP and Linux. It provides an interactive environment for solving a very broad class of optimal control problems. The users' problems can be prepared purely as M-files, and no compiler is required to solve them. To speed up the solution process, the functions defining the problem can be coded in C and then compiled and linked with some prebuilt linking libraries. The implemented numerical methods are supported by the theory outlined in [115], which uses the approach of consistent approximations. Systems dynamics can be integrated with fixed step-size Runge–Kutta integration, a discrete-time solver, or a variable step-size method. The software automatically computes gradients for all functions with respect to the controls and any free initial conditions. The controls are represented as splines, which allows for a high degree of function approximation accuracy without requiring a large number of control parameters. There are three main optimization routines, each suited for different levels of generality, and the most general is based on sequential quadratic programming methods [26] (it was also used in our computations reported in the next section).

Note that in RIOTS_95 the controls are internally approximated by linear, quadratic, or cubic splines, and this immediately defines the parameterization (3.15).

3.6 Illustrative Example

In this section, we use a demonstrative example to illustrate our method. We consider the following two-dimensional diffusion equation:

$$\frac{\partial y}{\partial t} = \nabla \cdot (\kappa \nabla y) + F \tag{3.30}$$

for $\boldsymbol{x} \in \Omega = (0, 1)^2$ and $t \in [0, 1]$, subject to homogeneous zero initial and Dirichlet boundary conditions, where $F(\boldsymbol{x}, t) = 20 \exp(-50(x_1 - t)^2)$. The spatial distribution of the diffusion coefficient is assumed to have the form

$$\kappa(x_1, x_2) = \theta_1 + \theta_2 x_1 + \theta_3 x_2. \tag{3.31}$$

In our example, we select the initial estimates of the parameter values as $\theta_1^0 = 0.1$, $\theta_2^0 = -0.05$, and $\theta_3^0 = 0.2$, which are assumed to be nominal and known prior to the experiment. The excitation function F in (3.30) simulates a source with a vertical line support along the x_2-axis, which moves like a plane wave with constant speed from the left to the right boundary of Ω within the observation interval [0, 1].

The determination of the Fisher information matrix for a given experiment requires the knowledge of the vector of the sensitivity coefficients $\boldsymbol{g} = \mathrm{col}[g_1, g_2, g_3]$ along sensor trajectories. The FIM can be obtained using the direct differentiation method [147] by solving the following system of PDEs:

$$\begin{aligned}
\frac{\partial y}{\partial t} &= \nabla \cdot (\kappa \nabla y) + F, \\
\frac{\partial g_1}{\partial t} &= \nabla \cdot \nabla y + \nabla \cdot (\kappa \nabla g_1), \\
\frac{\partial g_2}{\partial t} &= \nabla \cdot (x_1 \nabla y) + \nabla \cdot (\kappa \nabla g_2), \\
\frac{\partial g_3}{\partial t} &= \nabla \cdot (x_2 \nabla y) + \nabla \cdot (\kappa \nabla g_3),
\end{aligned} \tag{3.32}$$

in which the first equation represents the original state equation, and the next three equations are obtained from the differentiation of the first equation with respect to the three unknown parameters θ_1, θ_2, and θ_3, respectively. The initial and Dirichlet boundary conditions for all four equations are homogeneous.

System (3.32) has been solved numerically using the routines from the MATLAB PDE toolbox and stored g_1, g_2, and g_3 interpolated at the nodes of a rectangular grid in a four-dimensional array (we applied uniform partitions using 21 grid points per each spatial dimension and 31 points in time), see Appendix I in [147] for details. Since values of g_1, g_2, and g_3 may have been required at points that were not necessarily nodes of that grid, the relevant interpolation was thus performed using cubic splines in space (for this purpose, MATLAB's procedure `interp2` has been applied) and linear splines in time. Since, additionally, the derivatives of g with respect to spatial variables and time were required during the trajectory optimization process, these derivatives were approximated numerically using the central difference formula.

Next, we used RIOTS_95 to determine D-optimal sensor trajectories in accordance with the Wynn–Fedorov algorithm. The dynamics of the sensor mobility platform follow the following single integrator kinematic model:

$$\dot{\boldsymbol{x}}_s^j(t) = \boldsymbol{u}_s^j(t), \qquad \boldsymbol{x}_s^j(0) = \boldsymbol{x}_{s0}^j, \tag{3.33}$$

and additional constraints

$$\left|u_{si}^j(t)\right| \le 0.7, \quad t \in T, \; i = 1, \ldots, 6, \tag{3.34}$$

restricting the maximum mobile sensor velocity components that are imposed on the controls. Our goal is to design their trajectories so as to obtain the best possible estimates of θ_1, θ_2, and θ_3.

A program was implemented using a low-end PC (AMD Athlon 3800+, 2GB RAM) running on Windows XP and MATLAB 701 (R2006a). We ran the program twice with 4 iterations and 200 randomly chosen initial positions for each iteration. Each run took between 10 and 45 seconds for each initial position. This is necessary if we wish to get an approximation to a global maximum in Step 2 of the Wynn–Fedorov algorithm. This is a trade-off between the computation time and the number of possible initial positions.

Figures 3.1 and 3.3 present the results obtained for these two simulations. The initial sensor positions are marked with open circles, and the sensors' positions at the consecutive points of the time grid are marked with dots. When available, weights are inserted inside the figures, each weight being positioned by its respective trajectory.

The first run gives two different trajectories with weights of 0.54807 and 0.45193. Based on the generalized weighted least-squares criterion, each weight can be interpreted in terms of an experimental cost, which is inversely proportional to the variance of the observation error along a given trajectory. Thus, we may think of the weights as the cost related, for example, to the sensitivity of the measurement devices. Following this interpretation, we should spend approximately 55% of total experimental costs to assure a more accurate sensor for the first trajectory, and approximately 45% to the second trajectory, which requires a less sensitive sensor. On the contrary, the second run results in three distinct trajectories with weights of 0.44464, 0.34726, and 0.2081 (see Fig. 3.3). However, combining second and third trajectories together with the total weight 0.55536, we can observe that this solution is quite similar to the previous one with only two distinct sensor paths. The differences can be explained in terms of the suboptimality of the solutions for the internal problem in Step 2 of the Wynn–Fedorov algorithm (in order to assure the compromise between the computational burden and the quality of solution, in practice we are satisfied with fairly good approximation to the global optimum). Thus, in both simulations we come up with only different suboptimal solutions to our problem,

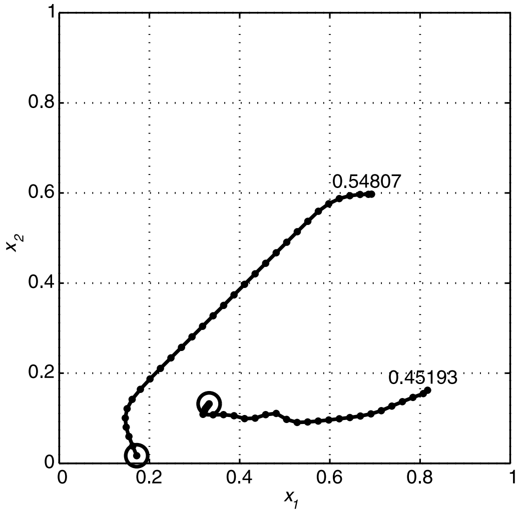


Fig. 3.1 Optimal trajectory of two mobile sensors using weighted D-optimality criterion ($\Psi = 7.4888$)

but with acceptable quality in the practical sense. The obtained Fisher information matrices are

$$M_{(1)} = \begin{pmatrix} 124.3815 & 68.0614 & 25.7666 \\ 68.0614 & 41.5653 & 13.4240 \\ 25.7666 & 13.4240 & 8.7691 \end{pmatrix} \tag{3.35}$$

and

$$M_{(2)} = \begin{pmatrix} 130.0149 & 72.3503 & 26.6154 \\ 72.3503 & 44.2181 & 14.1798 \\ 26.6154 & 14.1798 & 8.6267 \end{pmatrix} \tag{3.36}$$

with the criterion values Ψ equal to 7.4888 and 7.3672, respectively.

For comparison, we also present the results obtained using the technique described in [147] for D-optimum trajectories of moving sensors. This strategy is similar to ours but does not use weights in the computation of the FIM (or more precisely, the weights are fixed and assumed to be equal for each trajectory). Results are shown in Fig. 3.2 (two mobile sensors) and Fig. 3.4 (three mobile sensors).

3.7 Optimal Measurement Problem in the Average Sense

3.7.1 A Limitation of the Design of Optimal Sensing Policies for Parameter Estimation

As mentioned in Sect. 2.4, one of the main practical issues in optimal sensing policies is the dependence of the policy on the assumed values of the parameters' estimates. In most of the literature, the traditional approach is to consider a prior esti-

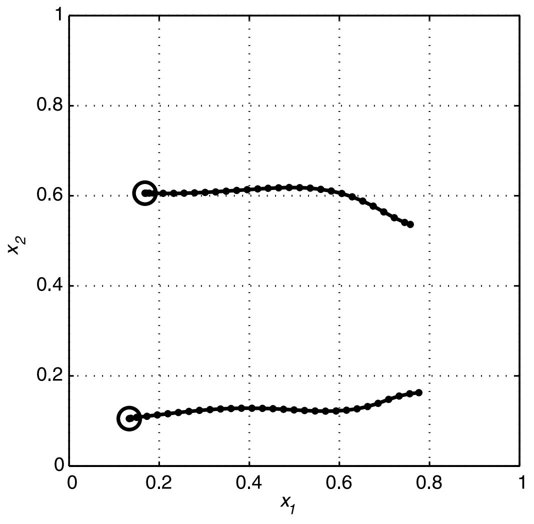


Fig. 3.2 Optimal trajectory of two mobile sensors using standard D-optimality criterion ($\Psi = 7.4017$)

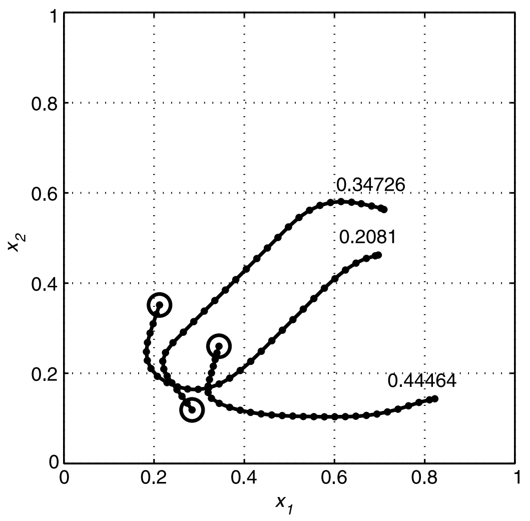


Fig. 3.3 Optimal trajectory of three mobile sensors using weighted D-optimality criterion ($\Psi = 7.3672$)

mate $\boldsymbol{\theta}^0$ of the true value of the parameters. But in practice, $\boldsymbol{\theta}^0$ can be very far from the true value $\boldsymbol{\theta}_{\text{true}}$, and a sensing policy designed for $\boldsymbol{\theta}^0$ can be a poor fit for $\boldsymbol{\theta}_{\text{true}}$.

One of the solutions described in the literature [111, 147] consists of creating an optimal sensing policy in the average sense. Such sensing policy is based on the fact that the true value of the parameters $\boldsymbol{\theta}_{\text{true}}$ belongs to the known compact set Θ_{ad}. An average sensing policy can be obtained such that its performance is good enough for any $\boldsymbol{\theta} \in \Theta_{\text{ad}}$. Another solution that will be presented in Chap. 5 is to create a finite-horizon control (FHC)-related method, where the sensing policy is divided into subpolicies. During each subexperiment, an optimal sensing policy is

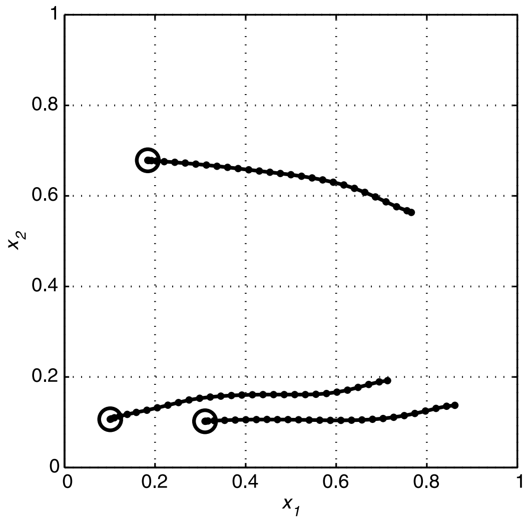


Fig. 3.4 Optimal trajectory of three mobile sensors using standard D-optimality criterion ($\Psi = 7.4959$)

determined based on the available parameter estimate, and the measurements taken are used to refine the value of the parameter estimates.

3.7.2 Problem Definition

When considering bounded parameter values, the optimal sensing policy problem can be defined by reformulating the FIM in the following way:

$$\boldsymbol{M} = \sum_{j=1}^{N} \int_{T} \boldsymbol{g}\big(\boldsymbol{x}_s^j(t), t\big) \boldsymbol{g}^{\mathsf{T}}\big(\boldsymbol{x}_s^j(t), t\big)\, \mathrm{d}t, \tag{3.37}$$

where

$$\boldsymbol{g}(\boldsymbol{x}, t) = \int_{\Theta_{\mathrm{ad}}} \nabla y(\boldsymbol{x}, t; \boldsymbol{\theta})\, \mathrm{d}\theta \tag{3.38}$$

denotes the vector of the so-called sensitivity coefficients in the average sense. We can observe that contrary to the previous definition (3.8), this one does not depend on a specific set of parameters θ^0 but on the whole set of possible parameter values.

The purpose of the optimal measurement problem is to determine the forces (controls) applied to each vehicle, which minimize the design criterion $\Psi(\cdot)$ defined on the FIMs of the form (3.37), which are determined unequivocally by the corresponding trajectories, subject to constraints on the magnitude of the controls and induced state constraints. To increase the degree of optimality, our approach considers $\boldsymbol{s}_0$ as a control parameter vector to be optimized in addition to the control function $\boldsymbol{u}_s$.

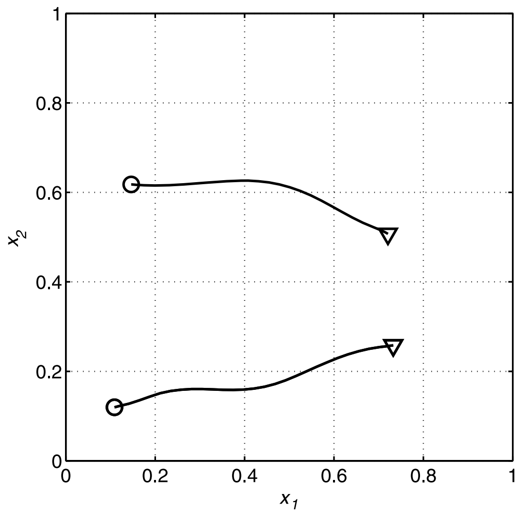


Fig. 3.5 Average D-optimal trajectories of a team of three sensors (two of them are collocated). The initial positions are marked with *open circles*, and the final positions are designated by *triangles*

Given the above formulation, we can cast the optimal measurement policy problem as the following optimization problem: Find the pair $(\boldsymbol{s}_0, \boldsymbol{u}_s)$ that minimizes

$$J(\boldsymbol{s}_0, \boldsymbol{u}_s) = \Phi\big[\boldsymbol{M}(\boldsymbol{s})\big] \tag{3.39}$$

over the set of feasible pairs

$$\begin{aligned}\mathcal{P} = \big\{(s_0, \boldsymbol{u}_s)\big|\boldsymbol{u}_s : T \to \mathbb{R}^r \quad &\text{is measurable,}\\ \boldsymbol{u}_{sl} \le \boldsymbol{u}_s(t) \le \boldsymbol{u}_{su} \text{ a.e. on } T, \boldsymbol{s}_0 \in \Omega_{\text{ad}}\big\},\end{aligned} \tag{3.40}$$

subject to the constraint (3.14).

3.7.3 An Illustrative Example

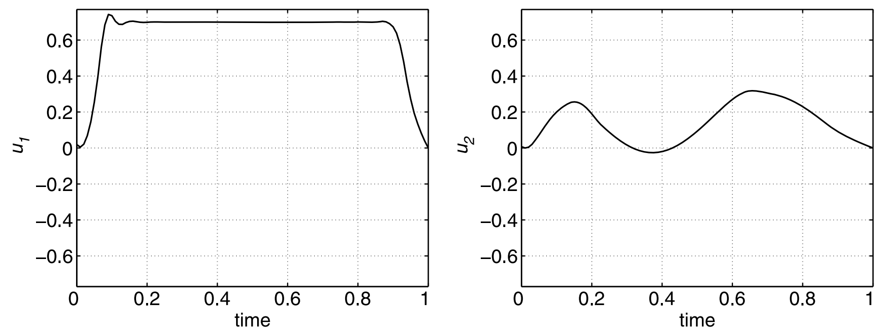
In this section, we consider the following two-dimensional diffusion equation similar to (3.30):

$$\frac{\partial y}{\partial t} = \nabla \cdot (\kappa \nabla y) + 20 \exp\big(-50(x_1 - t)^2\big) \tag{3.41}$$

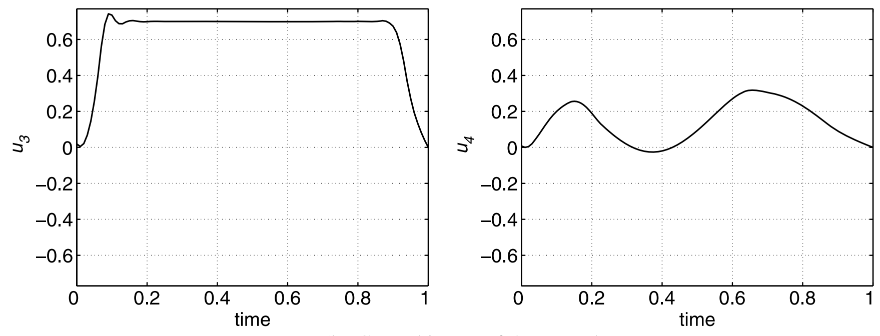
for $\boldsymbol{x} = [x_1\ x_2]^T \in \Omega = (0, 1)^2$ and $t \in [0, 1]$, subject to homogeneous zero initial and Dirichlet boundary conditions. The spatial distribution of the diffusion coefficient is assumed to have the form

$$\kappa(x_1, x_2) = \theta_1 + \theta_2 x_1 + \theta_3 x_2. \tag{3.42}$$

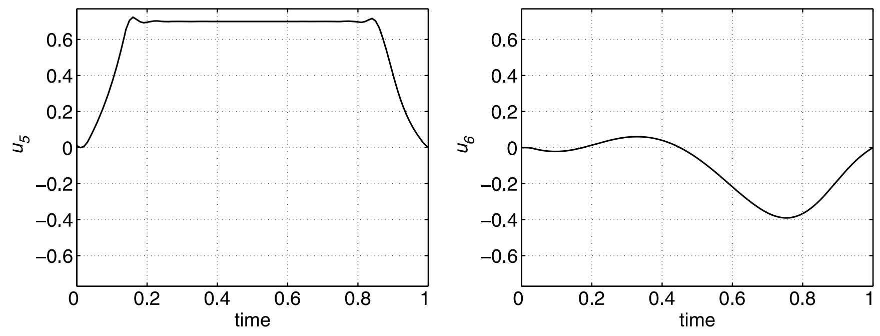
In this example, the chosen value intervals for the parameter are $\theta_1 \in [0.1, 0.7]$,



(a) Control inputs of the first sensor



(b) Control inputs of the second sensor



(c) Control inputs of the third sensor

Fig. 3.6 Control inputs of the mobile sensors

$\theta_2 \in [0.2, 0.6]$, and $\theta_3 \in [0.5; 1.0]$, which are assumed to be known prior to the experiment. The dynamics of the mobile sensors follow the single integrator kinematic model

$$\dot{\boldsymbol{x}}_s^j(t) = \boldsymbol{u}_s^j(t), \qquad \boldsymbol{x}_s^j(0) = \boldsymbol{x}_{s0}^j, \tag{3.43}$$

and additional constraints

$$\left|u_{si}^j(t)\right| \le 0.7, \quad t \in T, \ j = 1, \dots, N, \ i = 1, \dots, 2. \tag{3.44}$$

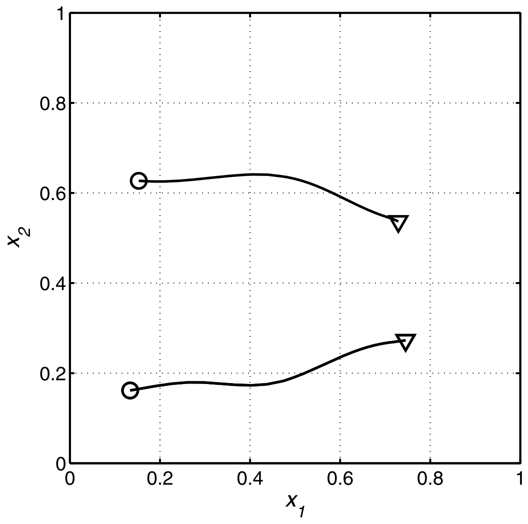


Fig. 3.7 Average D-optimal trajectories of a team of three sensors (two of them are collocated) for intermediate parameter values

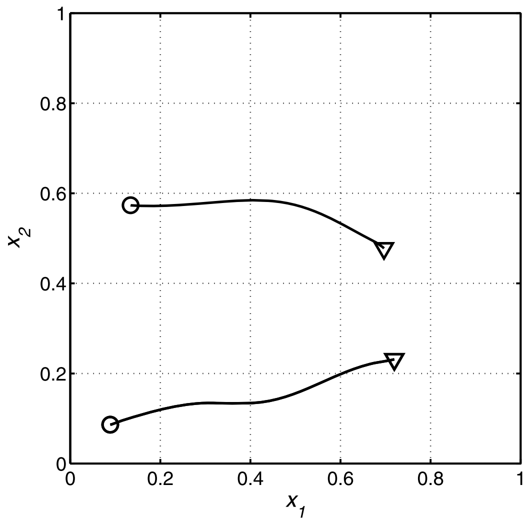


Fig. 3.8 Average D-optimal trajectories of a team of three sensors (two of them are collocated) for parameter values at the lower bound

Our goal is to design their trajectories so as to obtain possibly the best estimates of θ_1, θ_2, and θ_3 in the average sense.

In order to avoid getting stuck in a local minimum, computations were repeated several times from different initial solutions. Figure 3.5 presents the resulting trajectories for the best run. Steering signals for both sensor and actuator are displayed in Fig. 3.6.

For illustration purposes, the problem is solved for several particular values of the parameters. The resulting trajectories for the median values ($\theta_1 = 0.4$, $\theta_2 = 0.4$, and $\theta_3 = 0.75$) can be observed in Fig. 3.7, lower values ($\theta_1 = 0.1$, $\theta_2 = 0.2$, and $\theta_3 = 0.5$) in Fig. 3.8, and upper values ($\theta_1 = 0.7$, $\theta_2 = 0.6$, and $\theta_3 = 1.0$) in Fig. 3.9.

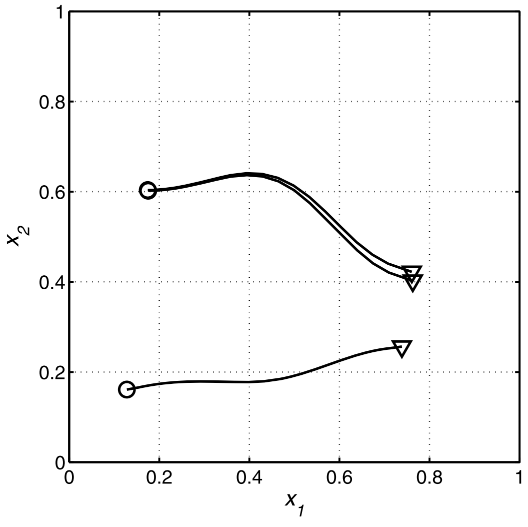


Fig. 3.9 D-optimal trajectories of a team of three sensors for parameter values at the upper bound

It is important to notice that the obtained results include cases where two sensors have the same trajectories. It is due to the uncorrelated nature of the measurement noise. From its definition, two collocated sensors could potentially provide more information than sensors with different trajectories.

3.8 Chapter Summary

The results in this chapter show that some well-known methods of optimum experimental design for linear regression models can be applied to the setting of the mobile sensor trajectory design problem for optimal parameter estimation of DPSs in case we wish to simultaneously optimize the number of sensors and their trajectories as well as to optimally allocate the experimental effort. The latter is understood here as allowing for different measurement accuracies of individual sensors, which are quantified by weights steering the corresponding measurement variances. This leads to a much more general setting that most frequently produces an uneven allocation of experimental effort between different sensors. This remains in contrast to the existing approaches. The corresponding solutions proposed in this chapter could obviously be implemented on a sensor network with heterogeneous mobile nodes. We demonstrate that these solutions can be determined using convex optimization tools commonly employed in optimum experimental design and show how to apply numerical tools of optimal control to determine the optimal solutions.

We also introduced the design of moving sensor optimal trajectories, which does not rely on initial estimates of the parameters but instead is based on knowledge of upper and lower bounds of the parameter values. In most research, the issue of initial estimates has been widely disregarded. Here, instead of using stochastic approximation algorithms for the search, we chose to rely on using the sensitivity coefficients in the average sense.

Chapter 4
Optimal Mobile Remote Sensing Policies

4.1 Introduction

We consider the case of an application where the use of mobile ground sensors is not practical or even feasible, for example, when the domain of interest is not smooth. Under those conditions, we are required to use mobile remote sensors, and, therefore, it is important to extend the framework of optimal mobile sensing policies to take into account the eventuality of remote sensing.

4.1.1 Literature Review

The juxtaposition of "real-life" physical systems and communication networks has brought to light a new generation of engineered systems, cyber-physical systems [72]. A definition of CPSs was given in [34] in the following way: "Computational thinking and integration of computation around the physical dynamic systems form CPS where sensing, decision, actuation, computation, networking and physical processes are mixed." Given its recent emergence and wide array of applications, the topic and study of CPSs are believed to become a highly researched area in the years to come including its conferences [3, 4] and journals [69]. "Applications of CPS arguably have the potential to dwarf the 20th century IT revolution" [94]. The applications of CPSs are numerous and include medical devices and systems, patient monitoring devices, automotive and air traffic control, advanced automotive systems, process control, environmental monitoring, avionics, instrumentation, oil refineries, water usage control, cooperative robotics, manufacturing control, smart buildings, etc.

Within these potential applications, the one we are interested in falls into the environmental monitoring category. It is believed that applying remote sensing can help determine the evapotranspiration of a given agricultural field and hence give improved information on crop condition and yield to perform better irrigation control. In the same vein of research, remote sensing can offer information correlated to the water stress level of the crops [83]. Remote sensing could provide important information to the farmers or even be used as feedback for a more real-time large scale irrigation control algorithm. Our ongoing project consists of developing UAVs equipped with multispectral aerial imagers to develop such a control algorithm [32].

In the considered framework, the system is a distributed parameter system, that is to say, the states are evolving along both time and space. Consequently, the traditional finite-dimensional input–output relationships have to be put aside, and partial differential equations have to be used to model the system. This increased complexity of the system leads to challenging problems. Whereas the location of sensors is rather straightforward when considering a finite-dimensional system, determining where measurement should be done is not a straightforward task in a DPS. One needs to consider the location of the sensors so that the gathered information best helps the parameter estimation. Therefore, it is a necessity to develop systematic approaches in order to increase the efficiency of PDE parameter estimation techniques.

The problem of sensor location in DPSs has been studied before as one can find in review papers [92, 147]. So far, the literature has limited the movements of the sensors within the domain of the distributed parameter system. However, with the emergence of remote sensing, we should extend the framework to mirror this new way of taking measurements. Our main motivation comes from our own projects [37]. With the help of small UAVs, we are capable of taking pictures and obtaining information on the amount of soil moisture on a specific plot of land. Such UAVs could also be used to gather information on soil water dynamics and help improve prediction of soil moisture. This approach is reflected in the illustrative example used later in this chapter.

4.1.2 *Problem Formulation for PDE Parameter Estimation*

Consider a DPS described by the partial differential equation

$$\frac{\partial y}{\partial t} = \mathcal{F}(\boldsymbol{x}, t, y, \boldsymbol{\theta}) \quad \text{in } \Omega_{\text{sys}} \times T, \tag{4.1}$$

with initial and boundary conditions

$$\mathcal{B}(\boldsymbol{x}, t, y, \boldsymbol{\theta}) = 0 \qquad \text{on } \Gamma_{\text{sys}} \times T, \tag{4.2}$$

$$y = y_0 \quad \text{in } \Omega_{\text{sys}} \times \{t = 0\}, \tag{4.3}$$

where $y(\boldsymbol{x}, t)$ stands for the scalar state at a spatial point $\boldsymbol{x} \in \bar{\Omega}_{\text{sys}} \subset \mathbb{R}^n$ and a time instant $t \in T$. $\Omega_{\text{sys}} \subset \mathbb{R}^n$ is a bounded spatial domain with sufficiently smooth boundary Γ, and $T = (0, t_f]$ is a bounded time interval. $\mathcal{F}$ is assumed to be a known, well-posed, possibly nonlinear, differential operator that includes first- and second-order spatial derivatives and includes terms for forcing inputs. $\mathcal{B}$ is a known operator acting on the boundary Γ, and $y_0 = y_0(\boldsymbol{x})$ is a given function.

We assume that the state y depends on the parameter vector $\boldsymbol{\theta} \in \mathbb{R}^m$ of unknown parameters to be determined from measurements made by N moving sensors. Those mobile sensors are assumed to ambulate in a spatial domain $\Omega_{\text{sens}} \neq \Omega_{\text{sys}}$. The sensors are able to remotely take measurements in $\Omega_{\text{meas}} \subset \Omega_{\text{sys}}$ over the observation horizon T. We call $\boldsymbol{x}_s^j : T \to \Omega_{\text{sens}}$ the position/trajectory of the jth sensor, where Ω_{sens} is a compact set representing the domain where the sensors can move. We call $\boldsymbol{z}_s^j : T \to \Omega$ the collection of measurements in Ω_{meas} where the jth sensor is observing. We assume that a function $\boldsymbol{f}_{\text{meas}} : \Omega_{\text{sens}} \to \Omega_{\text{meas}}$ linking the position of the sensor and measurements exists. The observations for the jth sensor are assumed to be of the form

$$\boldsymbol{z}_s^j(t) = \boldsymbol{y}\big(\boldsymbol{f}_{\text{meas}}\big(\boldsymbol{x}_s^j(t)\big), t\big) + \boldsymbol{\varepsilon}\big(\boldsymbol{f}_{\text{meas}}\big(\boldsymbol{x}_s^j(t)\big), t\big), \quad t \in T,\ j = 1, \ldots, N, \tag{4.4}$$

where ε represents the measurement noise assumed to be white, zero-mean, Gaussian, and spatial uncorrelated with the following statistics:

$$\mathrm{e}\big\{\varepsilon\big(\boldsymbol{f}_{\text{meas}}\big(\boldsymbol{x}_s^j(t)\big), t\big)\varepsilon\big(\boldsymbol{f}_{\text{meas}}\big(\boldsymbol{x}_s^i(t')\big), t'\big)\big\} = \sigma^2 \delta_{ji} \delta\big(t - t'\big), \tag{4.5}$$

where σ^2 stands for the standard deviation of the measurement noise, and δ_{ij} and $\delta(\cdot)$ are the Kronecker and Dirac delta functions, respectively.

With the above settings, similar to [147], the optimal parameter estimation problem is formulated as follows: Given the model (4.1)–(4.3) and the measurements $\boldsymbol{z}_s^j$ from the sensors $\boldsymbol{x}_s^j$, $j = 1, \ldots, N$, determine an estimate $\hat{\boldsymbol{\theta}} \in \Theta_{\text{ad}}$ (Θ_{ad} being the set of admissible parameters) of the parameter vector that minimizes the generalized output least-squares fit-to-data functional given by

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\vartheta} \in \Theta_{\text{ad}}}{\arg\min} \sum_{j=1}^{N} \int_T \big[\boldsymbol{z}_s^j(t) - \hat{\boldsymbol{y}}\big(\boldsymbol{f}_{\text{meas}}\big(\boldsymbol{x}_s^j(t)\big), t; \boldsymbol{\vartheta}\big)\big]^2 \, \mathrm{d}t \tag{4.6}$$

where $\hat{\boldsymbol{y}}$ is the solution of (4.1)–(4.3) with $\boldsymbol{\theta}$ replaced by $\boldsymbol{\vartheta}$.

By observing (4.6), it is possible to foresee that the parameter estimate $\hat{\boldsymbol{\theta}}$ depends on the number of sensors N and the mobile sensor trajectories $\boldsymbol{x}_s^j$. This fact triggered the research on the topic and explains why the literature so far focused on optimizing both the number of sensors and their trajectories. The intent was to select these design variables so as to produce best estimates of the system parameters after performing the actual experiment.

Since our approach is based on the methodology developed for optimal sensor location, we display it here as an introduction to the theory from [147] and [111]. In order to achieve optimal sensor location, some quality measure of sensor configurations based on the accuracy of the parameter estimates obtained from the observations is required. Such a measure is usually related to the concept of the Fisher information matrix, which is frequently referred to in the theory of optimal experimental design for lumped parameter systems [62]. Its inverse constitutes an approximation of the covariance matrix for the estimate of $\boldsymbol{\theta}$. Given the assumed statistics of the measurement noise, the FIM has the following representation [118, 147]:

$$\boldsymbol{M} = \sum_{j=1}^{N} \int_T \boldsymbol{g}\big(\boldsymbol{f}_{\text{meas}}(\boldsymbol{x}_s^j(t)), t\big)\boldsymbol{g}^{\mathsf{T}}\big(\boldsymbol{f}_{\text{meas}}(\boldsymbol{x}_s^j(t)), t\big)\,\mathrm{d}t, \tag{4.7}$$

where

$$\boldsymbol{g}(\boldsymbol{x}, t) = \nabla_{\boldsymbol{\vartheta}}\, y(\boldsymbol{x}, t; \boldsymbol{\vartheta})\big|_{\boldsymbol{\vartheta}=\boldsymbol{\theta}^0} \tag{4.8}$$

denotes the vector of the so-called sensitivity coefficients, $\boldsymbol{\theta}^0$ being a prior estimate to the unknown parameter vector $\boldsymbol{\theta}$ [146, 147].

As mentioned earlier, the FIM in its matrix format cannot be used directly in an optimization. Therefore, we have to rely on some scalar function Ψ of the FIM to perform the optimization. As described in Sect. 2.4, there are several candidates, and we choose the D-optimality criterion defined as

$$\Psi[\boldsymbol{M}] = -\log\det(\boldsymbol{M}). \tag{4.9}$$

4.2 Optimal Measurement Problem

4.2.1 Mobile Sensor Model

4.2.1.1 Sensor Dynamics

We assume that the sensing devices are equipped on vehicles whose dynamics can be described by the following differential equation:

$$\dot{\boldsymbol{x}}_s^j(t) = \boldsymbol{f}\big(\boldsymbol{x}_s^j(t), \boldsymbol{u}_s^j(t)\big) \quad \text{a.e. on } T,\ \boldsymbol{x}_s^j(0) = \boldsymbol{x}_{s0}^j. \tag{4.10}$$

With this nomenclature, the function $\boldsymbol{f} : \mathbb{R}^N \times \mathbb{R}^{r_s} \to \mathbb{R}^N$ has to be continuously differentiable, the vector $\boldsymbol{x}_{s0}^j \in \mathbb{R}^N$ represents the initial disposition of the jth sensor, and $\boldsymbol{u}_s : T \to \mathbb{R}^{r_s}$ is a measurable control function satisfying the following inequality:

$$\boldsymbol{u}_{sl} \le \boldsymbol{u}_s(t) \le \boldsymbol{u}_{su} \quad \text{a.e. on } T \tag{4.11}$$

for some known constant vectors $\boldsymbol{u}_{sl}$ and $\boldsymbol{u}_{su}$. Let us introduce

$$\boldsymbol{s}(t) = \big(\boldsymbol{x}_s^1(t), \boldsymbol{x}_s^2(t), \ldots, \boldsymbol{x}_s^N(t)\big)^T, \tag{4.12}$$

where $\boldsymbol{x}_s^j : T \to \Omega_{\text{sens}}$ is the trajectory of the jth sensor.

4.2.1.2 Mobility Constraints

We assume that all the mobile nodes equipped with sensors are confined within an admissible region Ω_{sensAD} (a given compact set) where the sensors are allowed to travel. Ω_{sensAD} can be conveniently defined as

$$\Omega_{\text{sensAD}} = \bigl\{\boldsymbol{x} \in \Omega_{\text{sens}} : b_{si}(\boldsymbol{x}) = 0,\ i = 1, \ldots, I\bigr\}, \tag{4.13}$$

where the b_{si} are known continuously differentiable functions. That is to say, the following constraints have to be satisfied:

$$h_{ij}\bigl(\boldsymbol{s}(t)\bigr) = b_{si}\bigl(\boldsymbol{x}_s^j(t)\bigr) \le 0 \quad \forall t \in T, \tag{4.14}$$

where $1 \le i \le I$ and $1 \le j \le N$. For simpler notation, we reformulate the conditions described in (4.14) in the following way:

$$\gamma_{sl}\bigl(\boldsymbol{s}(t)\bigr) \le 0 \quad \forall t \in T, \tag{4.15}$$

where γ_{sl}, $l = 1, \ldots, \nu$, tally with (4.14), $\nu = I \times N$. It would be possible to consider additional constraints on the path of the vehicles such as specific dynamics, collision avoidance, communication range maintenance, and any other conceivable constraints.

4.2.1.3 Remote Sensing Constraints

As mentioned earlier, we assume that the sensors are capable of taking measurements in Ω_{sys}, while being physically in Ω_{sens}. For that purpose, we introduce a remote sensing function f giving the location of the measurement based on the location of the sensor. Similar to path constraints, we assume that the remote sensing is only allowed within an admissible region Ω_{measAD} where the measurements are possible. The constraints on remote sensing can be defined as constraints on measurement location and then transformed into mobility ones. We can define Ω_{measAD} as

$$\Omega_{\text{measAD}} = \bigl\{\boldsymbol{x} \in \Omega_{\text{sens}} : b_{mi}\bigl(\boldsymbol{f}_{\text{meas}}(\boldsymbol{x})\bigr) = 0,\ i = 1, \ldots, I\bigr\}, \tag{4.16}$$

where the b_{mi} functions have the same properties as b_{si}. Similarly, we can regroup the remote sensing constraints into an inequality

$$\gamma_{ml}\bigl(\boldsymbol{s}(t)\bigr) \le 0, \quad t \in T. \tag{4.17}$$

Remark For our project [32], UAVs equipped with multispectral imagers are used for collecting aerial images of agricultural fields. The purpose of remote sensing is to gather data about the ground surface while avoiding contact with it. Multispectral imagers can generate an image for each different wavelength band ranging from visible spectra to infrared or thermal band for various applications. Having such a diverse and wide range of wavelengths allows for a better analysis of the ground surface properties. Under such circumstances, the domain where the sensors ambulate (space) is different from the domain where measurements are taken (ground). The constraints on mobility (such as collision avoidance between UAVs and/or environment) are different from the constraints on remote sensing (such as maintaining the images within the domain of interest that is the crop field).

4.2.2 Problem Definition

The purpose of the optimal measurement problem is to determine the forces (controls) applied to each vehicle, which minimize the design criterion $\Psi(\cdot)$ defined on the FIMs of the form (4.7), determined unequivocally by the corresponding trajectories, subject to constraints on the magnitude of the controls and the imposed state constraints. To increase the degree of optimality, our approach considers $\boldsymbol{s}(0) = \boldsymbol{s}_0$ as a control parameter vector to be optimized in addition to the control function $\boldsymbol{u}_s$.

Given the above formulation, we can cast the optimal measurement policy problem as the following optimization problem: Find the pair $(\boldsymbol{s}_0, \boldsymbol{u}_s)$ that minimizes

$$J(\boldsymbol{s}_0, \boldsymbol{u}_s) = \Phi\big[\boldsymbol{M}(\boldsymbol{s})\big] \tag{4.18}$$

over the set of feasible pairs

$$\begin{aligned}\mathcal{P} = \big\{(s_0, u_s) | u : T \to \mathbb{R}^r \quad &\text{is measurable,}\\ u_{sl} \le u_s(t) \le u_{su} \text{ a.e. on } T, &\boldsymbol{s}_0 \in \Omega_{\text{sens}}\big\},\end{aligned} \tag{4.19}$$

subject to the constraints (4.15) and (4.17).

This problem can hardly have an analytical solution. It is therefore necessary to rely on numerical techniques to solve the problem. A wide variety of techniques is available [115]. However, the problem can be reformulated as the classic Mayer problem where the performance index is defined only via terminal values of state variables.

4.3 Optimal Control Formulation

In this section, the problem is converted into a canonical optimal control one, making possible the use of existing optimal control problem solvers such as RIOTS_95.

To simplify our presentation, we define the function `svec`: $\mathbb{S}^m \to \mathbb{R}^{m(m+1)/2}$, where $\mathbb{S}^m$ denotes the subspace of all symmetric matrices in $\mathbb{R}^{m\times m}$, that takes the lower triangular part (the elements only on the main diagonal and below) of a symmetric matrix A and stacks them into a vector $\boldsymbol{a}$:

$$\boldsymbol{a} = \text{svec}(A) = \text{col}[A_{11}, A_{21}, \dots, A_{m1}, A_{22}, \dots A_{32}, \dots, A_{m2}, \dots, A_{mm}]. \tag{4.20}$$

Reciprocally, let $A = \mathrm{Smat}(\boldsymbol{a})$ be a symmetric matrix such that $\mathrm{svec}(\mathrm{Smat}(\boldsymbol{a})) = \boldsymbol{a}$ for any $\boldsymbol{a} \in \mathbb{R}^{m(m+1)/2}$.

Consider the matrix-valued function

$$\Pi\big(\boldsymbol{s}(t), t\big) = \sum_{j=1}^{N} \boldsymbol{g}\big(\boldsymbol{f}_{\mathrm{meas}}\big(\boldsymbol{x}_s^j(t)\big), t\big)\boldsymbol{g}^T\big(\boldsymbol{f}_{\mathrm{meas}}\big(\boldsymbol{x}_s^j(t)\big), t\big). \tag{4.21}$$

Setting $r : T \to \mathbb{R}^{m(m+1)/2}$ as the solution of the differential equations

$$\dot{\boldsymbol{r}}(t) = \mathrm{svec}\big(\Pi\big(\boldsymbol{s}(t), t\big)\big), \quad r(0) = 0, \tag{4.22}$$

we obtain

$$M(\boldsymbol{s}) = \mathrm{Smat}\big(\boldsymbol{r}(t_f)\big), \tag{4.23}$$

i.e., minimization of $\Phi[M(\boldsymbol{s})]$ thus reduces to minimization of a function of the terminal value of the solution to (4.22). We introduce an augmented state vector

$$\boldsymbol{q}(t) = \begin{bmatrix} \boldsymbol{s}(t) \\ \boldsymbol{r}(t) \end{bmatrix} \tag{4.24}$$

with

$$\boldsymbol{q}_0 = \boldsymbol{q}(0) = \begin{bmatrix} s_0 \\ \boldsymbol{0} \end{bmatrix}. \tag{4.25}$$

Then, the equivalent canonical optimal control problem occurs in finding a pair $(\boldsymbol{q}_0, \boldsymbol{u}_s) \in \bar{\mathcal{P}}$ that minimizes the performance index

$$\bar{J}(\boldsymbol{q}_0, \boldsymbol{u}_s) = \phi\big(\boldsymbol{q}(t_f)\big) \tag{4.26}$$

subject to

$$\begin{cases} \dot{\boldsymbol{q}}(t) = \phi(\boldsymbol{q}(t), \boldsymbol{u}_s(t), t), \\ \boldsymbol{q}(0) = \boldsymbol{q}_0, \\ \bar{\gamma}_{sl}(\boldsymbol{q}(t)) \le 0, \\ \bar{\gamma}_{ml}(\boldsymbol{q}(t)) \le 0, \end{cases} \tag{4.27}$$

where

$$\bar{\mathcal{P}} = \big\{(\boldsymbol{q}_0, \boldsymbol{u}) | \boldsymbol{u} : T \to \mathbb{R}^r \quad \text{is measurable,} \\ \boldsymbol{u}_l \le \boldsymbol{u}(t) \le \boldsymbol{u}_u \text{ a.e. on } T, \boldsymbol{s}_0 \in \Omega_{\mathrm{sens}}^M\big\}, \tag{4.28}$$

and

$$\phi(\boldsymbol{q}, \boldsymbol{u}, t) = \begin{bmatrix} f(\boldsymbol{s}(t), \boldsymbol{u}(t)) \\ \mathrm{svec}(\Pi(\boldsymbol{s}(t), t)) \end{bmatrix}, \tag{4.29}$$

$$\bar{\gamma}_{sl}\big(\boldsymbol{q}(t)\big) = \gamma_{sl}\big(\boldsymbol{s}(t)\big), \tag{4.30}$$

$$\bar{\gamma}_{ml}\big(\boldsymbol{q}(t)\big) = \gamma_{ml}\big(\boldsymbol{s}(t)\big). \tag{4.31}$$

The problem formulated above is clearly in a normal form that can be solved with readily available software packages for solving dynamic optimization problems numerically. A nonexhaustive list of such packages includes RIOTS_95 [126], DIRCOL [161], and MISER [82]. Like in most of our work, we use RIOTS_95, which is designed as a MATLAB toolbox written mostly in C and runs under Windows 98/2000/XP/Vista and Linux. The theory behind RIOTS_95 and its numerical methods can be found in [125].

4.4 An Illustrative Example

In this section, we use a simple example to illustrate the method developed earlier in this chapter. The system we consider here consists of the following two-dimensional diffusion equation:

$$\frac{\partial y(\boldsymbol{x},t)}{\partial t} = \nabla \cdot \big(\kappa \nabla y(\boldsymbol{x},t)\big) + 20\exp\big(-50(x_1 - t)^2\big) \tag{4.32}$$

for $\boldsymbol{x} = [x_1\ x_2]^T \in \Omega_{\mathrm{sys}} = (0, 1)^2$ and $t \in [0, 1]$, subject to homogeneous zero initial and Dirichlet boundary conditions. The spatial distribution of the diffusion coefficient is assumed to have the form

$$\kappa(x_1, x_2) = \theta_1 + \theta_2 x_1 + \theta_3 x_2. \tag{4.33}$$

In this example, the guessed values of the diffusion coefficient parameters (which we want to estimate) are $\theta_1^0 = 0.1$, $\theta_2^0 = -0.05$, and $\theta_3^0 = 0.2$. They are assumed to be known prior to the experiment. The dynamics of the mobile sensors follow the given dynamical model

$$\dot{\boldsymbol{x}_s}^j(t) = \boldsymbol{u}_s^j(t), \quad \boldsymbol{x}_s^j(0) = \boldsymbol{x}_{s0}^j, \tag{4.34}$$

for $\boldsymbol{x}_s^j = [x_{s1}^j\ x_{s2}^j\ x_{s3}^j]^T \in \Omega_{\mathrm{sens}} = (0, 1)^3$ with additional constraints

$$\big|u_i^j(t)\big| \le 0.7, \quad t \in T,\ j = 1, \dots, N,\ i = 1, 2, \tag{4.35}$$

$$\big|u_i^j(t)\big| \le 0.2, \quad t \in T,\ j = 1, \dots, N,\ i = 3. \tag{4.36}$$

We can notice that Ω_{sens} is of dimension 3 and Ω_{sys} is of dimension 2 and that Ω_{sys} lies in the boundary of Ω_{sens}. The remote sensing function $\boldsymbol{f}_{\mathrm{meas}}$ is defined in a way that is very similar to a downward looking camera mounted on an unmanned aerial vehicle. We assume that the mobile node's attitude is determined by an orthogonal basis directed by the control input $\boldsymbol{u}_s^j$. $\boldsymbol{u}_s^j$ gives us the direction that the robot is facing, the second axis is taken parallel to the $x_3 = 0$ plane, and the third axis is obtained by completing the orthogonal basis in a direct way. The obtained basis is

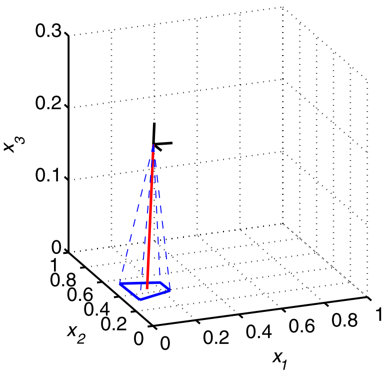


Fig. 4.1 Illustration of the remote sensing function

$\{\boldsymbol{e}_{j1}, \boldsymbol{e}_{j2}, \boldsymbol{e}_{j3}\}$, with $\boldsymbol{e}_{j1} = \boldsymbol{u}_s^j$. The view vector of the jth sensor is taken as $-\boldsymbol{e}_{j3}$, which can be seen as a camera facing downward. The vertical field of view is chosen as $\frac{\pi}{3}$ and the horizontal field of view is taken as $\frac{\pi}{2}$. Since we decided to model our remote sensor as a camera, we choose a resolution of 3×3. Measurements are taken at the intersection of the field of view and Ω_{sys}. To give the reader a better insight of the remote sensing function, we provide a visual description in Fig. 4.1. The orthogonal basis is in black, the view vector is represented by a red line, and the visual footprint is represented by a blue trapezoid.

The purpose of our optimization is to obtain the trajectories of a team of three sensors so as to determine the best possible estimates of the parameters θ_1, θ_2, and θ_3.

Since the sensing function is not pointwise, we reformulate (4.8) for our illustrative example:

$$\boldsymbol{g}(\boldsymbol{x}, t) = \sum_{i=1}^{\mathrm{res}} \sum_{j=1}^{\mathrm{res}} \nabla_{\boldsymbol{\vartheta}} y(\boldsymbol{x}_{ij}, t; \boldsymbol{\vartheta})|_{\boldsymbol{\vartheta}=\boldsymbol{\theta}^0} / \mathrm{res}^2, \tag{4.37}$$

where res stands for the resolution of the sensor (three in our case). In addition, to prevent the mobile nodes from intersecting with the system's domain Ω_{sys}, which would be equivalent to a crash, the optimality criteria are reformulated as

$$J(\boldsymbol{s}_0, \boldsymbol{u}) = \Phi\big[\boldsymbol{M}(\boldsymbol{s})\big] + \frac{1}{|x_3|}. \tag{4.38}$$

The resulting optimal trajectory of one mobile sensor can be observed in Fig. 4.2. The results for a team of two sensors are displayed in Fig. 4.3, and the case for three sensors is given in Fig. 4.4.

4.5 Chapter Summary

We have extended the existing framework of the design of mobile sensor trajectories that minimizes the volume of the confidence ellipsoid for the estimates to the

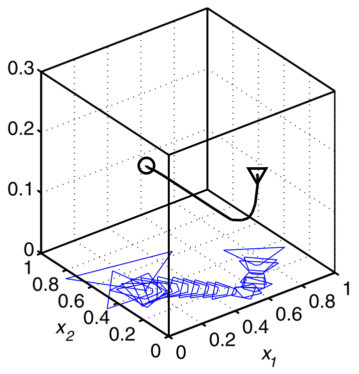


Fig. 4.2 D-optimal trajectory of one mobile remote sensor. The initial positions are marked with *open circles*, and the final positions are designated by *triangles*. The measured area is delineated by a *blue* trapezoid

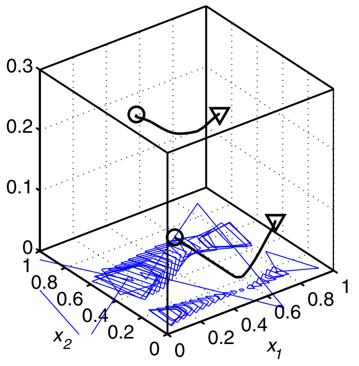


Fig. 4.3 D-optimal trajectories of two mobile remote sensors. The initial positions are marked with *open circles*, and the final positions are designated by *triangles*. The measured area is delineated by a *blue* trapezoid

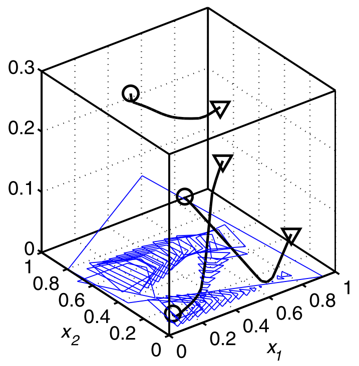


Fig. 4.4 D-optimal trajectories of three mobile remote sensors. The initial positions are marked with *open circles*, and the final positions are designated by *triangles*. The measured area is delineated by a *blue* trapezoid

emerging field of remote sensing. For that purpose, we introduced a remote sensing function linking the mobility domain and the sensing domain. It is important to notice that the introduced formulation can still be transformed into a canonical optimal control problem. This reformulation allows the problem to be solved by the MATLAB toolbox RIOTS_95, a collection of routines capable of solving a large class of finite-time optimal control problems, with the help of the MATLAB Partial Differential Equation Toolbox. The method was then applied to an illustrative example to demonstrate its applicability.

This remote sensing policy framework is becoming more important by the day because of the growing interest from the scientific community (especially in earth sciences) of using unmanned aerial platforms for collecting ground data. The remote sensing framework will be considered again in Chap. 8 but to solve the problem of downscaling surface soil moisture data.

Chapter 5
Online Optimal Mobile Sensing Policies: Finite-Horizon Control Framework

5.1 Introduction

This chapter is dedicated to the online solution to the problem of the sensitivity of optimal sensing policies to initial parameter estimates.

The work we present here enters the category of what is called "robust designs" [147]. The major problem with optimization of sensor locations is the dependence of the solution on the real values of the parameters to be estimated, as illustrated in Sect. 2.4. In general, this problem is solved by using a prior estimate of the parameter instead of the real value. In some cases, it may occur that this initial guess is very far from the real value and therefore the "optimal" solution obtained is far from the real optimum. Different approaches were introduced to remove this initial guess from the equation. The envisioned designs fall in four categories: sequential designs, optimal designs in the average sense, optimal designs in the minimax sense, and the use of randomized algorithms. For more information, please check Chap. 6 of [147]. Most work on the topic was based on stochastic approximation algorithms [148, 149, 151, 155] to limit the computational burden. With the rapid growth of computer power available, computationally intensive approaches are more and more viable. In addition, since those methods are based on offline computations, as long as the duration is reasonable, they do not present a major burden.

In [141], for the first time, we solved this problem by the proposed optimal interlaced mobile sensor motion planning and parameter estimation. The problem formulation is given in detail with a numerical solution for generating and refining the mobile sensor motion trajectories for parameter estimation of the distributed parameter system. The basic idea is to use the finite-horizon control type of scheme. First, the optimal trajectories are computed in a finite time horizon based on the assumed parameter values. For the following time horizon, the parameters of the distributed parameter system are estimated using the measured data in the previous time horizon, and the optimal trajectories are updated accordingly based on these estimated parameters obtained. Simulations are offered to illustrate the advantages of the proposed interlaced method over the noninterlaced techniques. We call the proposed scheme *online* or *real-time*, and it offers practical solutions to optimal measurement and estimation of a distributed parameter system when mobile sensors are used. It should be mentioned that this online problem has been recognized in the last chapter of [111] as an "extremely important" research effort.

We continue the type of research problem first introduced in [141]. We introduce communication topologies into the framework and study their influence on the behavior of the team of mobile sensors.

5.2 Optimal Mobile Sensing Policy: Finite-Horizon Closed-Loop Solution

5.2.1 A DPS and Its Mobile Sensors

To get ready for simulation demonstration, let us start with a generic DPS model describing a diffusion process with unknown parameters. Then, we define the mobile sensors used for taking measurements of this system. Our ultimate goal is to best identify the unknown DPS parameters using these mobile sensors.

The model used for a specific diffusion process is the same as in [130] except that the parameters are now assumed unknown. This allows us to compare the results between different estimation techniques.

The dynamics of the system under consideration are defined by

$$\begin{aligned}
\frac{\partial y(x_1,x_2,t)}{\partial t} &= \frac{\partial}{\partial x_1}\left(\kappa(x_1,x_2)\frac{\partial y(x_1,x_2,t)}{\partial x_1}\right) \\
&\quad + \frac{\partial}{\partial x_2}\left(\kappa(x_1,x_2)\frac{\partial y(x_1,x_2,t)}{\partial x_2}\right) \\
&\quad + 20\exp\bigl(-50(x_1-t)^2\bigr), \\
(x_1,x_2) \in \Omega &= (0,1)\times(0,1), \quad t \in T, \\
y(x_1,x_2,0) &= 0, \\
y(x_1,x_2,t) &= 0, \\
T &= \bigl\{t \mid t \in (0,1)\bigr\}, \\
\kappa &= \theta_1 + \theta_2 x_1 + \theta_3 x_2, \\
\theta_1 &= 0.1, \quad \theta_2 = 0.6, \quad \theta_3 = 0.8,
\end{aligned}$$

where $y(x_1,x_2,t)$ is the concentration of the considered diffusing substance, $\kappa(x_1,x_2)$ is the diffusion coefficient for the spatial coordinate (x_1,x_2), t is the time, and θ_1, θ_2, and θ_3 are the unknown values of the parameters to be estimated. The assigned values for θ_1, θ_2, and θ_3 are just for simulation comparison purposes.

5.2.2 Interlaced Optimal Trajectory Planning

5.2.2.1 Optimal Trajectory Planning

In order to solve the problem, we need to reformulate the problem in the optimal control framework. The solver used for this optimal control problem is called RIOTS [126]. RIOTS stands for "recursive integration optimal trajectory solver." It is a MATLAB toolbox programmed to solve a very broad class of optimal control problems. According to [126], our optimal trajectory planning problem can be solved using the RIOTS toolbox if rephrased as follows:

$$\min_{(u,\xi)\in L^2 N_\infty[t_0,t_f]\times\mathbb{R}^K} J(u,\xi), \tag{5.1}$$

where

$$J(u,\xi)=g_0\big(\xi,\boldsymbol{x}(t_f)\big)+\int_{t_0}^{t_f} l_0(\boldsymbol{x},t,u)\,\mathrm{d}t, \tag{5.2}$$

subject to the following conditions and constraints:

$$\begin{aligned}
&\dot{\boldsymbol{x}}=h(\boldsymbol{x},t,\tau),\\
&\boldsymbol{x}(t_0)=\xi,\quad t\in[t_0,t_f],\\
&u_{\min}^{(j)}(t)\le u^{(j)}(t)\le u_{\max}^{(j)}(t),\quad j=1,\dots,N,\ t\in[t_0,t_f],\\
&\xi_{\min}^{(j)}(t)\le \xi^{(j)}(t)\le \xi_{\max}^{(j)}(t),\quad j=1,\dots,K,\ t\in[t_0,t_f],\\
&l_{ti}\big(\boldsymbol{x}(t),t,\tau(t)\big)\le 0,\ t\in[t_0,t_f],\\
&g_{ei}\big(\xi,\boldsymbol{x}(t_f)\big)\le 0,\quad g_{ee}\big(\xi,\boldsymbol{x}(t_f)\big)=0.
\end{aligned}$$

In the case of our optimal trajectory planning problem, $\dot{\boldsymbol{x}}=h(t,\boldsymbol{x},u)=A\boldsymbol{x}+Bu$. Instead of defining $l_0(\xi,\boldsymbol{x}(t_f))=\Psi(\boldsymbol{M})$, we choose to define $g_0(\xi,\boldsymbol{x}(tf))=\int_{t_0}^{tf}\Psi(\boldsymbol{M})\,\mathrm{d}t$, in order to lower the amount of calculations. The reformulation is achieved by using the "Mayer equivalent problem" technique described in Sect. 4.3.

5.2.2.2 Measurements and Parameters Estimation

Once the optimal trajectories have been computed, the measurements are done as described in Sect. 3.2. However, the observations are completed until the end of the finite horizon for which the trajectory was computed. Instead, after a fraction of the horizon, the data gathered so far are used to refine the estimation of the parameters values.

In order to determine refined values of the parameters, we use the MATLAB command "`lsqnonlin`", a routine for solving nonlinear least-squares problems, and

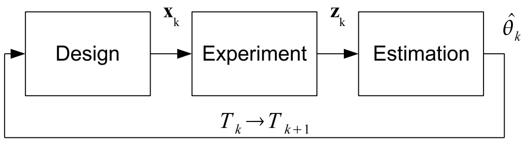


Fig. 5.1 The interlaced scheme illustrated

especially for our case, the least-squares fitting problems. "`lsqnonlin`" allows the user to incorporate one's own function to compute. In our problem, the input of the function is a set of parameters as well as the measurements, and the output is the error between the measurement and the simulated value of the measurement for the set of parameters:

$$\min_{\boldsymbol{\theta}} = \frac{1}{2}\sum_{i=1}^{N} f_i(\boldsymbol{\theta})^2 \tag{5.3}$$

with

$$f_i(\boldsymbol{\theta}) = z^i(t_0,\dots,t_k) \\ - \mathcal{H}\big(y\big(\boldsymbol{x}_s^i(t_0,\dots,t_k), t_0,\dots,t_k;\boldsymbol{\theta}\big), \boldsymbol{x}_s^i(t_0,\dots,t_k), t_0,\dots,t_k\big). \tag{5.4}$$

Prior to the experiment, we determine the value of the state $y(\boldsymbol{x},t,\boldsymbol{\theta})$ for a set of parameter values $\boldsymbol{\theta} \in \Theta_{ad}$ in an offline manner. We assume that the state variations between two values of a parameter are linear enough to allow interpolation. Using this database obtained offline allows faster computation of the function to be called by the optimization algorithm.

5.2.2.3 Summary of the Interlaced Scheme

Let us summarize the interlaced strategy step by step:

1. Given a set of parameters $\hat{\boldsymbol{\theta}}$ for the DPS (its initial value being given prior to the first iteration), we design an optimal experiment, i.e., optimal trajectories for the mobile sensors to follow.
2. The sensors take measurements along their individually assigned trajectories. Measurements are simulated, taking the real value of the state along the trajectory and adding zero-mean white noise.
3. Measurement data are used to refine the estimate of the parameters using an optimization routine such as "`lsqnonlin`". The optimization routine computes the parameters such that the difference between the measurements and the simulated values of the state along the trajectory is minimized. Go back to Step 1.

The above algorithm is illustrated in Fig. 5.1.

5.2.3 Illustrative Simulations

We focus our attention on the performance of the methodology. The experiment is run for different noise statistics, and for each case, results are given in the form of sensor trajectories and parameter estimates. For Case 1, $\sigma = 0.0001$, for Case 2, $\sigma = 0.001$, and for Case 3, $\sigma = 0.01$. In all cases, we consider three mobile sensors. The control of the mobile sensors u is limited between -0.7 and 0.7. All three sensors have fixed initial positions ($\boldsymbol{x}^1(0) = (0.1, 0.1)$, $\boldsymbol{x}^2(0) = (0.1, 0.5)$, and $\boldsymbol{x}^3(0) = (0.1, 0.9)$). The results for the previously defined case are respectively given in Fig. 5.2 for Case 1, in Fig. 5.3 for Case 2, and in Fig. 5.4 for Case 3. For each figure, subfigure (a) gives the sensor trajectories, the evolution of the estimates is shown in (b), and the measurements are given in (c).

From these figures, we have the following observations:

- In all the cases, the sensors have similar trajectories as they try to follow the excitation wave along the x_1 axis $20\exp(-50(x_1 - t)^2)$.
- For low noise amplitude (Cases 1 and 2), the experiment is long enough to obtain good estimates of the parameters. In Case 3, the experiment is not long enough to reach convergence.
- In all cases, we can clearly observe that the trajectories of the mobile sensors change as the estimated values of the parameters are getting closer to the real values.

5.2.4 A Second Illustrative Example

We use the same DPS as earlier, but we consider the mobile remote sensing problem from Chap. 4. The dynamics of the mobile sensors follow the same dynamical model

$$\dot{\boldsymbol{x}}_s^j(t) = \boldsymbol{u}_s^j(t), \qquad \boldsymbol{x}_s^j(0) = \boldsymbol{x}_{s0}^j, \tag{5.5}$$

for $\boldsymbol{x} = [x_1\ x_2\ x_3]^T \in \Omega_{\text{sens}} = (0, 1)^3$ and additional constraints

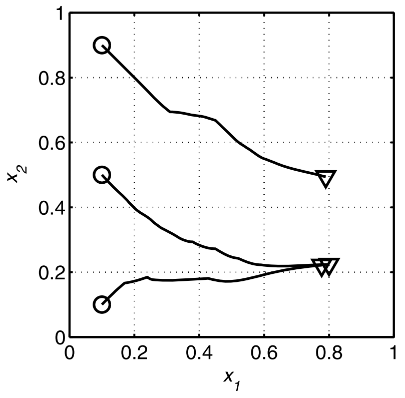
$$\left|u_i^j(t)\right| \le 0.6, \quad t \in T,\ j = 1, \dots, 2,\ i = 1, 2, \tag{5.6}$$

$$\left|u_i^j(t)\right| \le 0.2, \quad t \in T,\ j = 1, \dots, N,\ i = 3. \tag{5.7}$$

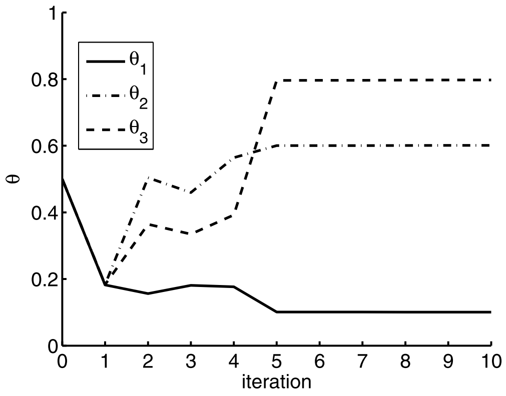
The remote sensor has a fixed initial position ($\mathbf{x}^1(0) = (0.1, 0.5, 0.1)$). The initial estimates for the parameter values are $\theta_1 = 0.3$, $\theta_2 = 0.5$, and $\theta_3 = 0.5$.

The resulting optimal trajectory of one mobile sensor can be observed in Fig. 5.5, and the evolution of the parameter estimates is given in Fig. 5.6.

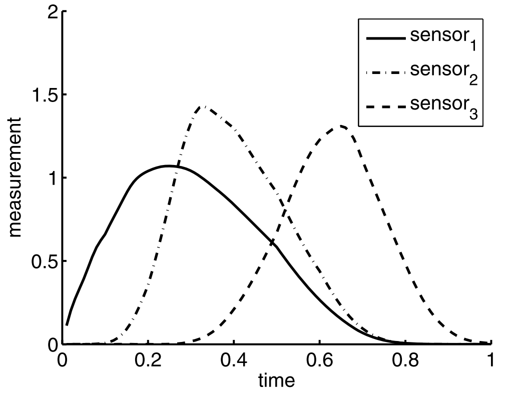
Fig. 5.2 Closed-loop D-optimum experiment for $\sigma = 0.0001$



(a) Sensor trajectories

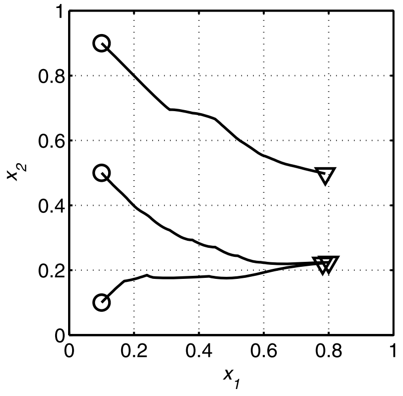


(b) Parameter estimates

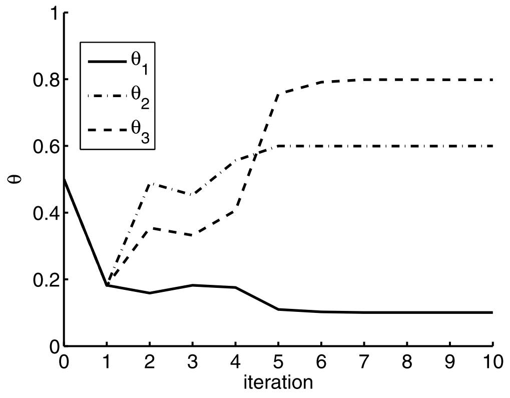


(c) Sensor measurements

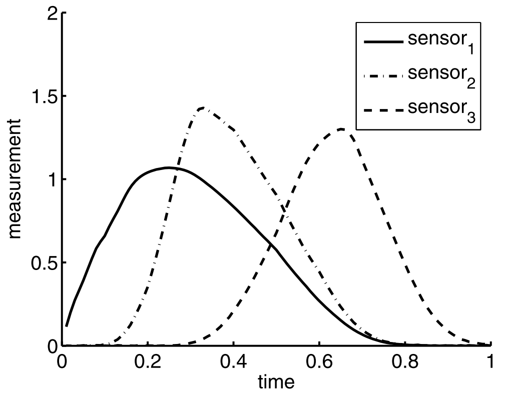
Fig. 5.3 Closed-loop D-optimum experiment for $\sigma = 0.001$



(a) Sensor trajectories

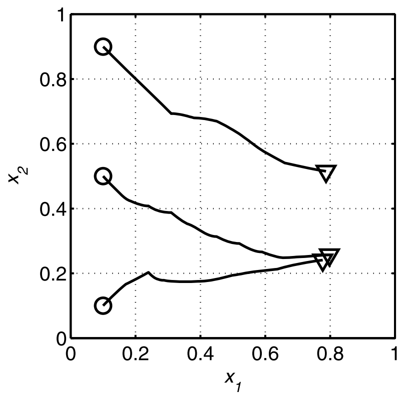


(b) Parameter estimates

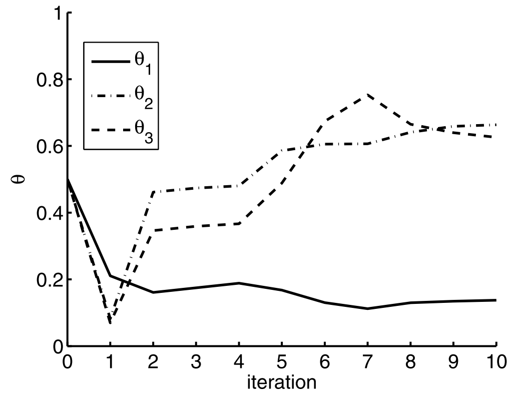


(c) Sensor measurements

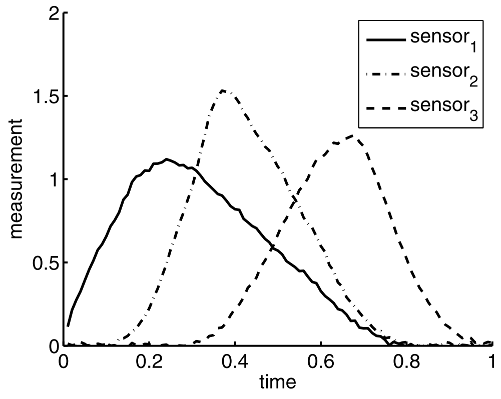
Fig. 5.4 Closed-loop D-optimum experiment for $\sigma = 0.01$



(a) Sensor trajectories



(b) Parameter estimates



(c) Sensor measurements

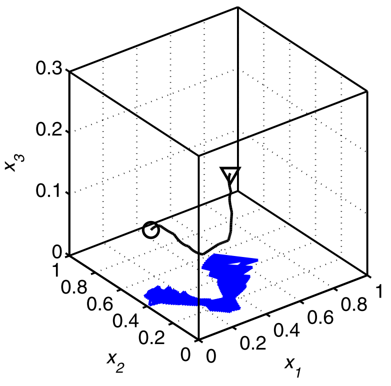


Fig. 5.5 Closed-loop D-optimal trajectory of one mobile remote sensor. The initial position is marked with an *open circle*, and the final position is designated by a *triangle*. The measured area is delineated by *blue* trapezoids

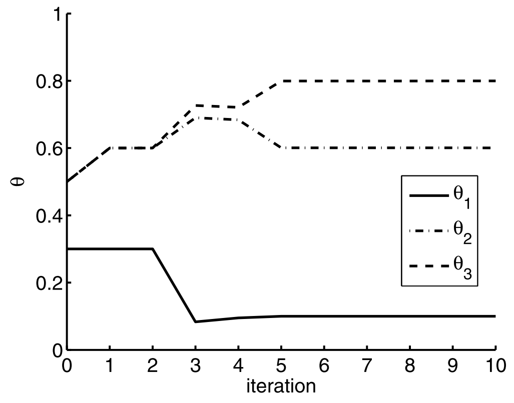


Fig. 5.6 Evolution of the online parameter estimates during the mobile remote sensing

5.3 Communication Topology in Online Optimal Sensing Policy for Parameter Estimation of Distributed Parameter Systems

5.3.1 The Interlaced Scheme with Communication Topology

The interlaced strategy when considering communication topology can be described as follows:

1. Given a set of parameters $\hat{\boldsymbol{\theta}}$ for the DPS, and the other sensors' current location, each sensor computes its optimal trajectory.
2. The sensors take measurements along the path of the obtained trajectory. The data gathered are then exchanged with other sensors according to a given communication topology.
3. Measurement data are used to refine the estimate of the parameters using an optimization routine, and a new set of system parameters is obtained.

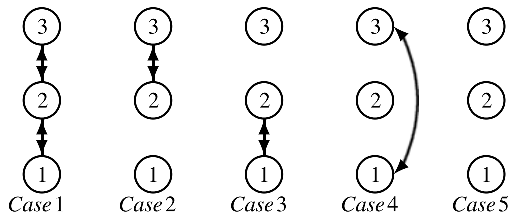
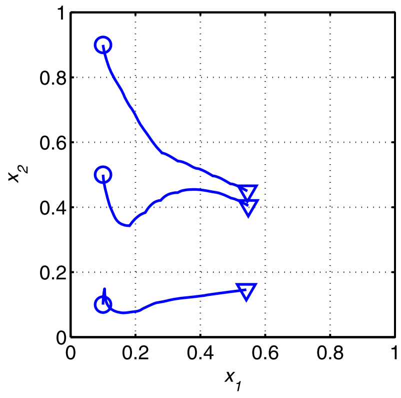
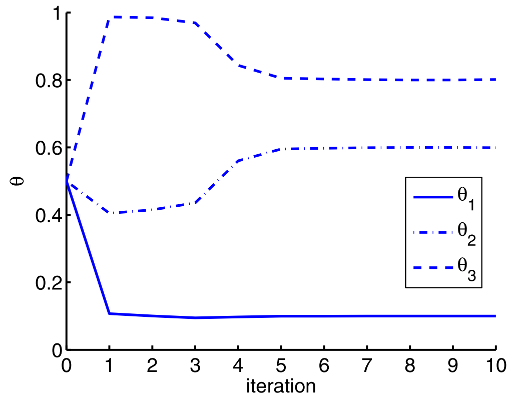


Fig. 5.7 Communication topologies considered for the illustrative example



(a) Sensor trajectories



(b) Parameter estimates

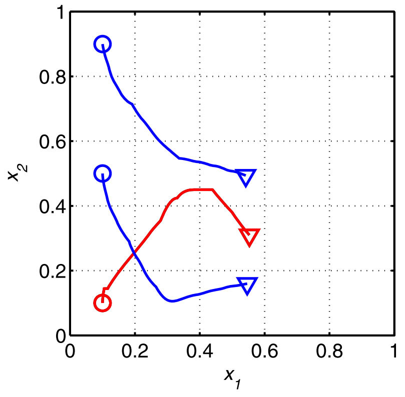
Fig. 5.8 Closed-loop D-optimum experiment for Case 1

5.3.2 An Illustrative Example

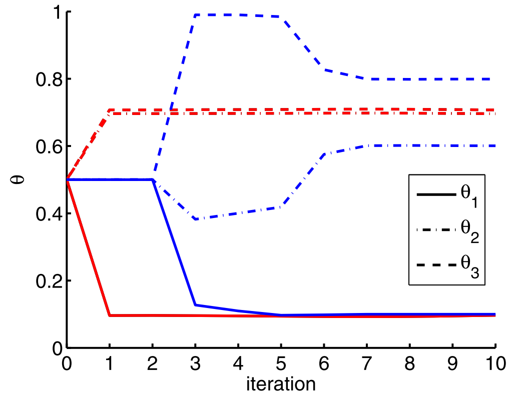
Here, we use a demonstrative example to illustrate our method. We consider the two-dimensional diffusion equation

$$\frac{\partial y}{\partial t} = \nabla \cdot (\kappa \nabla y) + F(\boldsymbol{x}, t) \tag{5.8}$$

Fig. 5.9 Closed-loop D-optimum experiment for Case 2



(a) Sensor trajectories



(b) Parameter estimates

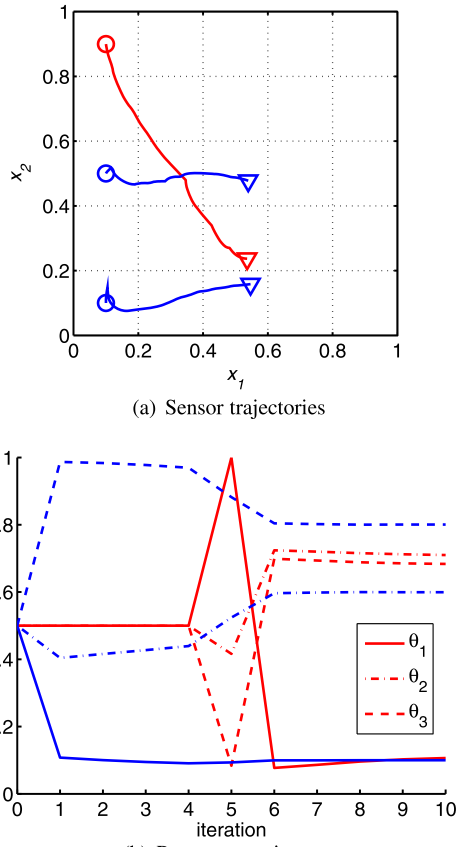
for $\boldsymbol{x} = [x_1\ x_2]^T \in \Omega = (0, 1)^2$ and $t \in [0, 1]$, subject to homogeneous zero initial and Dirichlet boundary conditions. The actuation function is given by $F(\boldsymbol{x}, t) = 20\exp(-50(2x_1 - t)^2)$. We can see that the excitation function F in (5.8) can be described as a source with a vertical line shape along the x_2-axis and moves like a wave with constant speed from the left to the right boundary of Ω between time $[0, 2]$. The spatial distribution of the diffusion coefficient is assumed to have the form

$$\kappa(x_1, x_2) = \theta_1 + \theta_2 x_1 + \theta_3 x_2. \tag{5.9}$$

In this example, the chosen values for the parameter are $\theta_1 = 0.1$, $\theta_2 = 0.6$, and $\theta_3 = 0.8$. Next, we are using RIOTS_95 to determine time-optimal sensor trajectories. The dynamics follow the simple model

$$\dot{\boldsymbol{x}}_s(t) = \boldsymbol{u}_s(t), \qquad \boldsymbol{x}(0) = \boldsymbol{x}_{s0}, \tag{5.10}$$

Fig. 5.10 Closed-loop D-optimum experiment for Case 3



and the constraints

$$\left|u_{si}(t)\right| \le 0.7, \quad t \in T, \; i = 1, \ldots, 6, \tag{5.11}$$

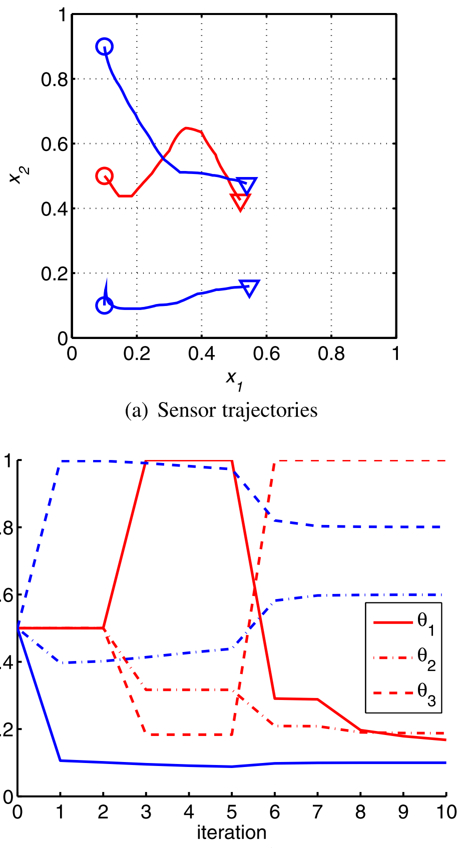
imposed on the controls; we are interested in designing their trajectories so as to obtain estimates of θ_1, θ_2, and θ_3. All three sensors have fixed initial positions ($\boldsymbol{x}_s^1(0) = (0.1, 0.1)$, $\boldsymbol{x}_s^2(0) = (0.1, 0.5)$, and $\boldsymbol{x}_s^3(0) = (0.1, 0.9)$). The initial estimates for the parameter values are $\theta_1 = 0.5$, $\theta_2 = 0.5$, and $\theta_3 = 0.5$.

We consider five different cases with different communication topologies. These topologies are detailed in Fig. 5.7.

The resulting experiments can be observed in Figs. 5.8 to 5.12. In each case, the initial positions are marked with open circles, and the final positions are designated by triangles. Sensors communicating with each other have the same color. Each figure contains both the resulting trajectories and the evolution of the parameters' estimates.

We can observe that for all cases, the sensors' trajectories follow the trend of the actuation function F. As expected, the communication topology has a great influ-

Fig. 5.11 Closed-loop D-optimum experiment for Case 4



(a) Sensor trajectories

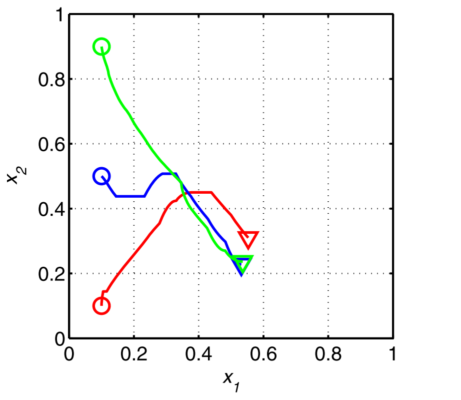
(b) Parameter estimates

ence on the experiment outcome. In Case 1, where all three sensors communicate with each other, the estimates become accurate starting from iteration 5. In Cases 2, 3, and 4, the two sensors communicating obtain a good estimate from iteration 7 (6 for Case 3), whereas the isolated sensor is not able to obtain accurate parameter values. In Case 5, it is surprising that the second sensor is able to estimate the system's parameters accurately from iteration 6. However, the two other sensors do not converge to the real parameter values.

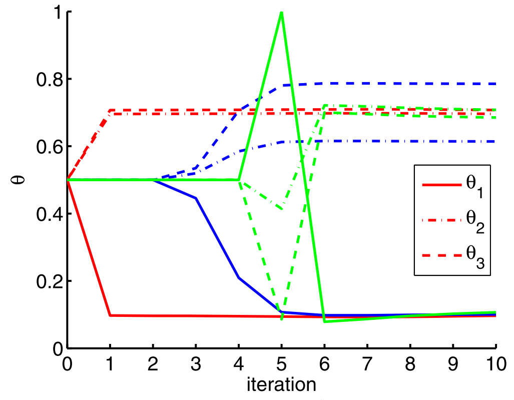
5.4 Convergence of the Interlaced Scheme

The proof of the convergence of the parameter estimation in the interlaced scheme is still under investigation. We have identified two directions to follow in the liter-

Fig. 5.12 Closed-loop D-optimum experiment for Case 5



(a) Sensor trajectories



(b) Parameter estimates

ature that could provide leads to demonstrate the proof. The first one is linked with the stability in the model predictive control (MPC) framework [54]. The second comes from the framework of sequential designs for parameter estimation for linear systems [163]. The first step in the proof will consist of finding the proper assumptions. The first assumption that has been identified is the weak persistent excitation condition.

Once the proof of convergence is obtained, we will focus on determining the convergence speed of the interlaced scheme based on the system's parameters. Then, we will study the effects of communication topologies on the convergence and its speed. Finally, we will be able to consider directed communication topologies. The directed communication topologies are fascinating in the online optimal sensing policy framework because the sensors not only can share their location, but they can also share their measurements, their parameter estimates, and their trajectories.

5.5 Chapter Summary

We introduced a numerical procedure for optimal sensor-motion scheduling of diffusion systems for parameter estimation. With the knowledge of the PDE governing a given DPS, mobile sensors find an initial trajectory to follow and refine the trajectory as their measurements allow finding a better estimate of the system's parameters. Using the MATLAB PDE toolbox for the system's simulations, RIOTS MATLAB toolbox for solving the optimal path-planning problem, and MATLAB Optimization toolbox for the estimation of the system's parameters, we were able to solve this parameter identification problem in an interlaced manner successfully. Simulation results are presented to show both the advantages of the strategy and the convergence of the estimation.

We were able to introduce the concept of communication topology into the framework of optimal sensor-motion scheduling of diffusion systems for parameter estimation. The method was successfully applied to an example. Our results show that when the sensors are not communicating, the lack of information greatly decreases the performance of the strategy.

Chapter 6
Optimal Mobile Actuation/Sensing Policies for Parameter Estimation of Distributed Parameter Systems

6.1 Introduction

So far in this monograph, our interest has been focused on optimal sensing policies. But as with any system, the actuation can also provide useful information for the estimation of parameters when combined with sensors. The main contribution of this chapter is the introduction of the actuation policy as a design variable in the framework, rather than a given input.

Determining a rich excitation to increase the relevance of observations and measurements of the states of a distributed parameter system is not a straightforward task. One needs to consider the actuation capabilities and location of the actuators so that the gathered information best helps the parameter estimation. Therefore, it is a necessity to develop systematic approaches in order to increase the efficiency of PDE parameter estimators. The problem of sensor location is not new as in, for example, review papers [92] and [147]. However, the investigation on how to best excite the PDE system for optimal parameter estimation has not been attempted so far. This chapter presents a framework for such optimal mobile actuation policy aiming at optimal parameter estimation of a class of distributed parameter systems.

In the field of mobile sensor trajectory planning, few approaches have been developed so far, but numerous scenarios have been considered. Rafajłowicz [119] investigated the problem using the determinant of the Fisher information matrix associated with the parameters he wanted to estimate. However, his results are more of an optimal time-dependent measure than a trajectory. In [147] and [146], Uciński reformulated the problem of time-optimal path planning into a state-constrained optimal-control one that allows the addition of different constraints on the dynamics of the mobile sensor. In [153], Uciński tried to properly formulate and solve the time-optimal problem for moving sensors that observe the state of a DPS in order to estimate its parameters. In [154], Turing's Measure of Conditioning was used to obtain optimal sensor trajectories. The problem was solved for heterogeneous sensors (i.e., with different measurement accuracies) in [144]. Limited power resource was considered in [112]. In [130], Song and colleagues added realistic constraints to the dynamics of the mobile sensor by considering a differential-drive mobile robot in the framework of the MAS-net Project.

The system is considered to have a known sensor setup, and mobile actuators are used to stimulate the system so that measurements from the sensors, possibly mobile, provide the best information for parameter estimation.

6.1.1 Problem Formulation for PDE Parameter Estimation

Consider a distributed parameter system described by the partial differential equation

$$\frac{\partial y}{\partial t} = \mathcal{F}(\boldsymbol{x}, t, y, \boldsymbol{\theta}) \quad \text{in } \Omega \times T, \tag{6.1}$$

with initial and boundary conditions

$$\mathcal{B}(\boldsymbol{x}, t, y, \boldsymbol{\theta}) = 0 \qquad \text{on } \Gamma \times T, \tag{6.2}$$

$$y = y_0 \quad \text{in } \Omega \times \{t = 0\}, \tag{6.3}$$

where $y(\boldsymbol{x}, t)$ stands for the scalar state at a spatial point $\boldsymbol{x} \in \bar{\Omega} \subset \mathbf{R}^n$ and time instant $t \in T$. $\Omega \subset \mathbf{R}^n$ is a bounded spatial domain with sufficiently smooth boundary $\Gamma = \partial\Omega$, and $T = (0, t_f]$ is a bounded time interval. $\mathcal{F}$ is assumed to be a known well-posed, possibly nonlinear, differential operator that includes first- and second-order spatial derivatives and includes terms for forcing inputs. $\mathcal{B}$ is a known operator acting on the boundary Γ, and $y_0 = y_0(\boldsymbol{x})$ is a given function.

We assume that the state y depends on the unknown parameter vector $\boldsymbol{\theta} \in \mathbf{R}^m$ to be determined from measurements made by N static or moving pointwise sensors over the observation horizon T. We call $\boldsymbol{x}_s^j : T \to \Omega_{\text{ad}}$ the position/trajectory of the jth sensor, where $\Omega_{\text{ad}} \subset \Omega \cup \Gamma$ is a compact set representing the domain where measurements are possible. The observations for the jth sensor are assumed to be of the form

$$z^j(t) = y\big(\boldsymbol{x}_s^j(t), t\big) + \varepsilon\big(\boldsymbol{x}_s^j(t), t\big), \quad t \in T,\ j = 1, \ldots, N, \tag{6.4}$$

where ϵ represents the measurement noise assumed to be white, zero-mean, Gaussian, and spatial uncorrelated with the following statistics

$$\mathrm{e}\big\{\varepsilon\big(\boldsymbol{x}_s^j(t), t\big)\varepsilon\big(\boldsymbol{x}_s^i\big(t'\big), t'\big)\big\} = \sigma^2 \delta_{ji} \delta(t - t'), \tag{6.5}$$

where σ^2 stands for the standard deviation of the measurement noise, and δ_{ij} and $\delta(\cdot)$ are the Kronecker and Dirac delta functions, respectively.

With the above settings, similar to [147], the optimal parameter estimation problem is formulated as follows: Given the model (6.1)–(6.3) and the measurements z^j from the sensors $\boldsymbol{x}_s^j$, $j = 1, \ldots, N$, determine an estimate $\hat{\boldsymbol{\theta}} \in \Theta_{\text{ad}}$ (Θ_{ad} being the set of admissible parameters) of the parameter vector that minimizes the generalized output least-squares fit-to-data functional given by

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\vartheta}\in\Theta_{\mathrm{ad}}}{\arg\min} \sum_{j=1}^{N} \int_T \big[z^j(t) - y\big(\boldsymbol{x}_s^j(t), t; \boldsymbol{\vartheta}\big)\big]^2 \,\mathrm{d}t, \tag{6.6}$$

where y is the solution of (6.1)–(6.3) with $\boldsymbol{\theta}$ replaced by $\boldsymbol{\vartheta}$.

By observing (6.6), it is possible to foresee that the parameter estimate $\hat{\boldsymbol{\theta}}$ depends on the number of sensors N and the mobile sensor trajectories $\boldsymbol{x}_s^j$. This fact triggered the research on the topic and explains why the literature so far focused on optimizing both the number of sensors and their trajectories. The intent was to select these design variables so as to produce best estimates of the system parameters after performing the actual experiment.

Note that, besides these explicit design variables, there exists an implicit one that is the forcing input in (6.1). Therefore, for given sensor trajectories, our interest here focuses on designing the optimal forcing input so as to get the most accurate parameter estimates.

6.2 Optimal Actuation Problem

The optimal actuation problem is very close to the optimal measurement problem in the sense that both use the sensitivity coefficients as a measure of the quality of the parameter estimation. However, both problems differ in the following ways:

- The optimal measurement problem assumes that the forcing input in (6.1) is known whereas the optimal actuation problem attempts to optimize trajectories of mobile actuators constituting part of the entirety of the forcing input.
- In the optimal actuation problem, the sensors' positions/trajectories are known beforehand and are not optimized.

6.2.1 Mobile Actuator Model

We assume that the actuators are mounted on vehicles whose dynamics are described by the following equation:

$$\dot{\boldsymbol{x}}_a^j(t) = \boldsymbol{f}\big(\boldsymbol{x}_a^j(t), \boldsymbol{u}^j(t)\big) \quad \text{a.e. on } T, \quad \boldsymbol{x}_a^j(0) = \boldsymbol{x}_{a0}^j, \tag{6.7}$$

where the function $f : \mathbb{R}^M \times \mathbb{R}^r \to \mathbb{R}^M$ has to be continuously differentiable, $\boldsymbol{x}_{a0}^j \in \mathbb{R}^M$ represents the initial disposition of the actuators, and $\boldsymbol{u} : T \to \mathbb{R}^r$ is a measurable control function satisfying the following inequality:

$$\boldsymbol{u}_{al} \le \boldsymbol{u}_a(t) \le \boldsymbol{u}_{au} \quad \text{a.e. on } T \tag{6.8}$$

for some constant vectors $\boldsymbol{u}_{al}$ and $\boldsymbol{u}_{au}$. Let us introduce

$$\boldsymbol{s}(t) = \big(\boldsymbol{x}_a^1(t), \boldsymbol{x}_a^2(t), \dots, \boldsymbol{x}_a^M(t)\big), \tag{6.9}$$

where $\boldsymbol{x}_a^k : T \to \Omega_{\text{ad}}$ is the trajectory of the kth actuator. We assume that all the vehicles are confined within an admissible region Ω_{ad} (a given compact set) where the actuation is possible. Ω_{ad} can be conveniently defined as

$$\Omega_{\text{ad}} = \big\{\boldsymbol{x} \in \Omega : b_i(\boldsymbol{x}) = 0, i = 1, \dots, I\big\}, \tag{6.10}$$

where the b_i functions are known continuously differentiable functions. That is to say, that the following constraints have to be satisfied:

$$h_{ij}\big(\boldsymbol{s}(t)\big) = b_i\big(\boldsymbol{x}_a^j(t)\big) \le 0 \quad \forall t \in T, \tag{6.11}$$

where $1 \le i \le I$ and $1 \le j \le N$. For simpler notation, we reformulate the conditions described in (6.11) in the following way:

$$\gamma_l\big(\boldsymbol{s}(t)\big) \le 0 \quad \forall t \in T, \tag{6.12}$$

where γ_l, $l = 1, \dots, \nu$, tally with (6.11), $\nu = I \times N$.

The actuation function for the ith mobile actuator is assumed to have the following form:

$$\mathcal{F}_i(\boldsymbol{x}, t) = \mathcal{G}_i\big(\boldsymbol{x}, \boldsymbol{x}_a^i, t\big). \tag{6.13}$$

6.2.2 Problem Definition

To define the considered problem, we reformulate (6.1):

$$\frac{\partial y}{\partial t} = \mathcal{F}(\boldsymbol{x}, t, y, \boldsymbol{\theta}) + \sum_{k=1}^{M} \mathcal{F}_k(\boldsymbol{x}, t) \quad \text{in } \Omega \times T, \tag{6.14}$$

and initial and boundary conditions remain unchanged. $\mathcal{F}$ may still include forcing inputs terms.

For the framework of optimal actuation, the FIM is given by the following new representation:

$$\boldsymbol{M}(s) = \sum_{k=1}^{M} \int_T \boldsymbol{h}\big(\boldsymbol{x}_a^k(t), t\big)\,\mathrm{d}t, \tag{6.15}$$

where for the kth actuator,

$$\boldsymbol{h}\big(\boldsymbol{x}_a^k(t), t\big) = \sum_{j=1}^{N} \boldsymbol{g}\big(\boldsymbol{x}_a^k(t), \boldsymbol{x}_s^j(t), t\big)\boldsymbol{g}^\mathsf{T}\big(\boldsymbol{x}_a^k(t), \boldsymbol{x}_s^j(t), t\big), \tag{6.16}$$

and

$$\boldsymbol{g}\big(\boldsymbol{x}_a^k(t), \boldsymbol{x}(t), t\big) = \int_T \nabla_{\boldsymbol{\vartheta}} y\big(\boldsymbol{x}(\tau), \tau; \boldsymbol{\vartheta}\big)\big|_{\boldsymbol{\vartheta}=\boldsymbol{\theta}^0} \,\mathrm{d}\tau. \tag{6.17}$$

In (6.17), y is the solution of (6.14) for $\mathcal{F}_k(\boldsymbol{x}, \tau) = \mathcal{G}_i(\boldsymbol{x}, \boldsymbol{x}_a^i, \tau)\delta(t-\tau)$ for all $k \in [1,\ M]$.

The purpose of the optimal actuation problem is to determine the forces (controls) applied to each vehicle conveying an actuator, which minimize the design criterion $\Psi(\cdot)$ defined on the FIMs of the form (6.15), which are determined by the corresponding trajectories. Our approach considers $\boldsymbol{s}_0$ as a control parameter vector to be optimized in addition to the control function $\boldsymbol{u}_a$.

Given the above formulation, we can cast the optimal actuation policy problem as the following optimization problem: Find the pair $(\boldsymbol{s}_0, \boldsymbol{u}_a)$ that minimizes

$$J(\boldsymbol{s}_0, \boldsymbol{u}_a) = \Phi\big[\boldsymbol{M}(\boldsymbol{s})\big] \tag{6.18}$$

over the set of feasible pairs

$$\begin{aligned}\mathcal{P} = \big\{(\boldsymbol{s}_0, \boldsymbol{u}_a) | \boldsymbol{u}_a : T \to \mathbb{R}^r \text{ is measurable,}\\ \boldsymbol{u}_{al} \le \boldsymbol{u}_a(t) \le \boldsymbol{u}_{au} \text{ a.e. on } T, \boldsymbol{s}_0 \in \Omega_{\text{ad}}^M\big\},\end{aligned} \tag{6.19}$$

subject to the constraint (6.12).

This problem does not have an analytical solution. It is therefore necessary to rely on numerical techniques to solve the problem. However, the problem can be reformulated as a classic Mayer problem where the performance index is defined only via terminal values of state variables. The reformulation is achieved by using the reformulation described in Sect. 4.3.

6.2.3 An Illustrative Example

In this section, we use a demonstrative example to illustrate our method. We consider the two-dimensional diffusion equation

$$\frac{\partial y}{\partial t} = \nabla \cdot (\kappa \nabla y) + \sum_{i=1}^{M} F_i \tag{6.20}$$

for $\boldsymbol{x} = [x_1\ x_2]^T \in \Omega = (0, 1)^2$ and $t \in [0, 1]$, subject to homogeneous zero initial and Dirichlet boundary conditions. The spatial distribution of the diffusion coefficient is assumed to have the form

$$\kappa(x_1, x_2) = \theta_1 + \theta_2 x_1 + \theta_3 x_2. \tag{6.21}$$

In our example, we select the initial estimates of the parameter values as $\theta_1^0 = 0.1$, $\theta_2^0 = -0.05$, and $\theta_3^0 = 0.2$, which are assumed to be nominal and known prior to the experiment. The actuation function is

$$F_i\big(\boldsymbol{x}, \boldsymbol{x}_a^i, t\big) = 1000 \exp\big(-50\big(\big(x_{a1}^i - x_1\big)^2 + \big(x_{a2}^i - x_2\big)^2\big)\big), \tag{6.22}$$

where $\boldsymbol{x}_a^i = [x_{a1}^i \; x_{a2}^i]^T$. The dynamics of the mobile actuators follow the simple model

$$\dot{\boldsymbol{x}}_a^j(t) = \boldsymbol{u}_a^j(t), \qquad \boldsymbol{x}_a^j(0) = \boldsymbol{x}_{a0}^j, \tag{6.23}$$

and additional constraints

$$\big|u_{ai}^j(t)\big| \le 0.7, \quad t \in T,\ i = 1, \ldots, M. \tag{6.24}$$

Our goal is to design their trajectories so as to obtain possibly the best estimates of θ_1, θ_2, and θ_3.

The determination of the Fisher information matrix for a given experiment requires the knowledge of the vector of the sensitivity coefficients $\boldsymbol{g} = [g_1, g_2, g_3]^T$ along sensor trajectories. The FIM can be obtained using the direct differentiation method [147] by solving the following set of PDEs:

$$\begin{aligned} \frac{\partial y}{\partial t} &= \nabla \cdot (\kappa \nabla y) + \sum F_k, \\ \frac{\partial g_1}{\partial t} &= \nabla \cdot \nabla y + \nabla \cdot (\kappa \nabla g_1), \\ \frac{\partial g_2}{\partial t} &= \nabla \cdot (x_1 \nabla y) + \nabla \cdot (\kappa \nabla g_2), \\ \frac{\partial g_3}{\partial t} &= \nabla \cdot (x_2 \nabla y) + \nabla \cdot (\kappa \nabla g_3), \end{aligned} \tag{6.25}$$

in which the first equation represents the original state equation, and the next three equations are obtained from the differentiation of the first equation with respect to the parameters θ_1, θ_2, and θ_3, respectively. The initial and Dirichlet boundary conditions for all four equations are homogeneous.

Five different given sensor setups are considered, and for each setup, optimal actuation trajectories of different numbers of actuators (1, 2, and 3) are compared:

- One static sensor located in the center of the domain $(0.5, 0.5)$.
- One static sensor located near one of the corners of the domain $(0.2, 0.8)$.
- Three static sensors located throughout the domain $((0.1, 0.7), (0.5, 0.2), (0.6, 0.4))$.
- One moving sensor with a linear motion $(0.1, 0.2) \to (0.6, 0.7)$.
- Two moving sensors. One moving sensor with a linear motion $(0.1, 0.2) \to (0.6, 0.7)$, and the other one moves along an arc.

Results for the different cases are summarized in Table 6.1, and the resulting trajectories can be observed in Figs. 6.1–6.5. In the figures, static sensor locations are represented by a red $\times$, mobile sensor trajectories are in red, and actuator trajectories are in blue ($\bigcirc$ locates the starting point, and $\triangledown$ the ending point).

Table 6.1 Values of the D-optimality criterion $\Psi(M)$ for different test cases

	Case 1	Case 2	Case 3	Case 4	Case 5
1 actuator	15.991	18.051	10.904	14.465	12.547
2 actuators	12.582	14.273	7.36	11.095	7.4806
3 actuators	11.28	13.022	5.8136	9.8976	6.4512

As expected, for all cases, the performance criterion value decreases as the number of actuators increases. We can also notice that the mobility, population, and location of the sensors have a direct impact on the performance of the strategy. Therefore, we can suppose the existence of an optimal combination of sensor and actuator trajectories.

6.3 Optimal Measurement/Actuation Problem

6.3.1 Mobile Sensor/Actuator Model

We assume that both sensors and actuators are equipped on vehicles whose dynamics can be described by the following differential equation:

$$\dot{\boldsymbol{x}}_x^j(t) = \boldsymbol{f}_x\big(\boldsymbol{x}_x^j(t), \boldsymbol{u}_x^j(t)\big) \quad \text{a.e. on } T, \quad \boldsymbol{x}_x^j(0) = \boldsymbol{x}_{x0}^j, \tag{6.26}$$

where $\boldsymbol{x}$ can stand for two different categories, the first being s for sensors, and the second being a for actuators.

With this nomenclature, the function $\boldsymbol{f}_x$ ($\boldsymbol{f}_s : \mathbb{R}^N \times \mathbb{R}^{r_s} \to \mathbb{R}^N$ for sensors, $\boldsymbol{f}_a : \mathbb{R}^M \times \mathbb{R}^{r_a} \to \mathbb{R}^M$ for actuators) has to be continuously differentiable, the vector $\boldsymbol{x}_{x0}^j$ ($\boldsymbol{x}_{s0}^j \in \mathbb{R}^N$ for sensors, $\boldsymbol{x}_{a0}^j \in \mathbb{R}^M$ for actuators) represents the initial disposition of the jth sensor/actuator, and $\boldsymbol{u}_x$ ($\boldsymbol{u}_s : T \to \mathbb{R}^{r_s}$ for sensors, $\boldsymbol{u}_a : T \to \mathbb{R}^{r_a}$ for actuators) is a measurable control function satisfying the following inequality

$$\boldsymbol{u}_{xl} \le \boldsymbol{u}_x(t) \le \boldsymbol{u}_{xu} \quad \text{a.e. on } T \tag{6.27}$$

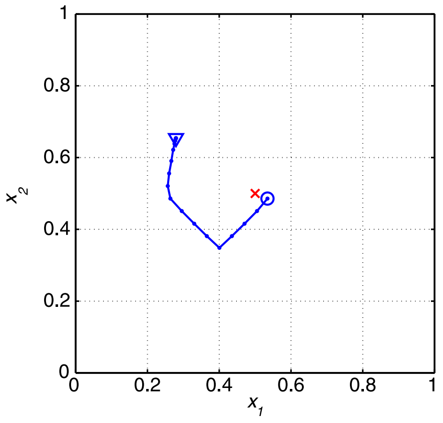
for some known constant vectors $\boldsymbol{u}_{xl}$ and $\boldsymbol{u}_{xu}$. Let us introduce

$$\boldsymbol{s}(t) = \big(\boldsymbol{x}_s^1(t), \boldsymbol{x}_s^2(t), \dots, \boldsymbol{x}_s^N(t), \boldsymbol{x}_a^1(t), \dots, \boldsymbol{x}_a^M(t)\big)^T, \tag{6.28}$$

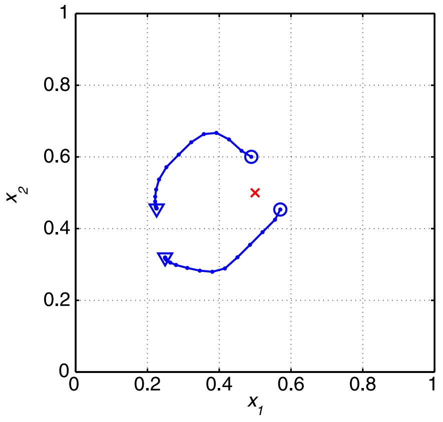
where $\boldsymbol{x}_s^j : T \to \Omega_{s\text{ad}}$ is the trajectory of the jth sensor, and $\boldsymbol{x}_a^k : T \to \Omega_{a\text{ad}}$ is the trajectory of the kth actuator. We assume that all the mobile nodes equipped with sensors are confined within an admissible region $\Omega_{s\text{ad}}$ (a given compact set) where the measurements are possible and reciprocally that all mobile nodes equipped with actuators are restrained in a domain $\Omega_{a\text{ad}}$ where actuation can be achieved. Considering the general index x defined earlier, $\Omega_{x\text{ad}}$ can be conveniently defined as

$$\Omega_{x\text{ad}} = \big\{\boldsymbol{x}_x \in \Omega : b_{xi}(\boldsymbol{x}_x) = 0,\ i = 1, \dots, I\big\}, \tag{6.29}$$

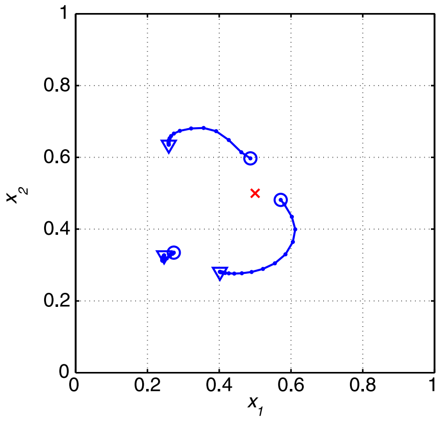
Fig. 6.1 D-optimum trajectories of mobile actuators for one stationary sensor



(a) One actuator

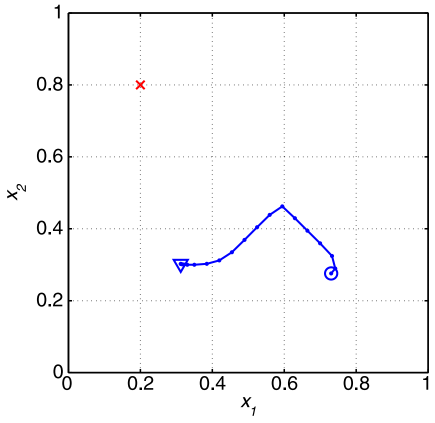


(b) Two actuators

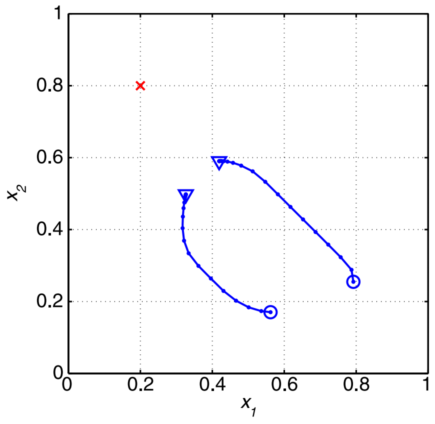


(c) Three actuators

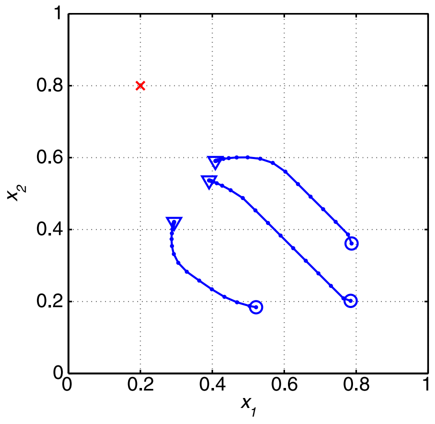
Fig. 6.2 D-optimum trajectories of mobile actuators for one stationary sensor



(a) One actuator

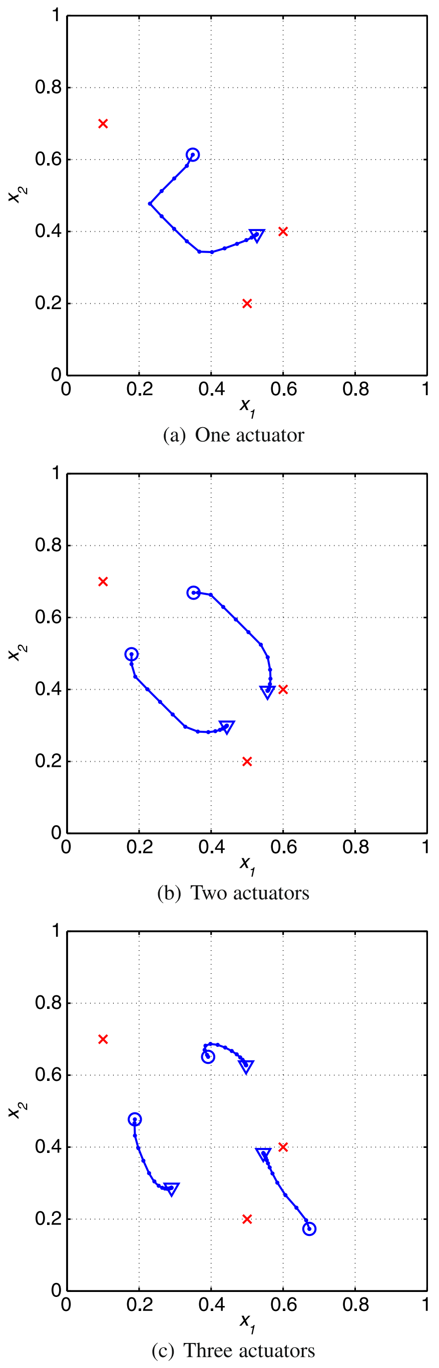


(b) Two actuators



(c) Three actuators

Fig. 6.3 D-optimum trajectories of mobile actuators for three stationary sensors

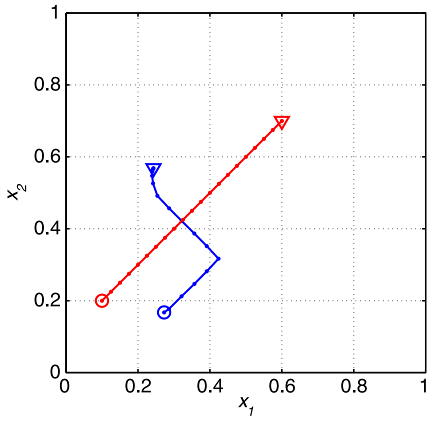


(a) One actuator

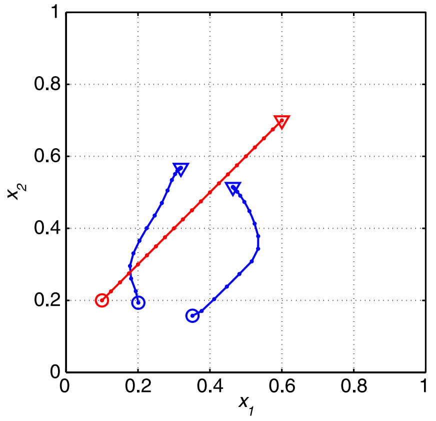
(b) Two actuators

(c) Three actuators

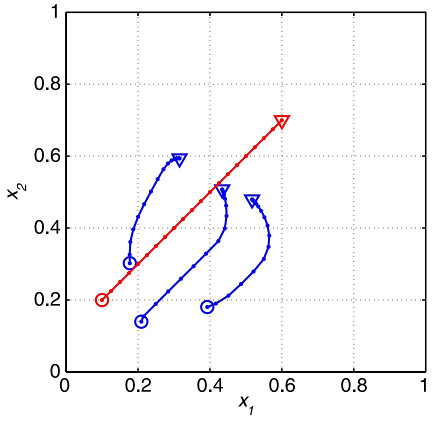
Fig. 6.4 D-optimum trajectories of mobile actuators for one mobile sensor



(a) One actuator



(b) Two actuators



(c) Three actuators

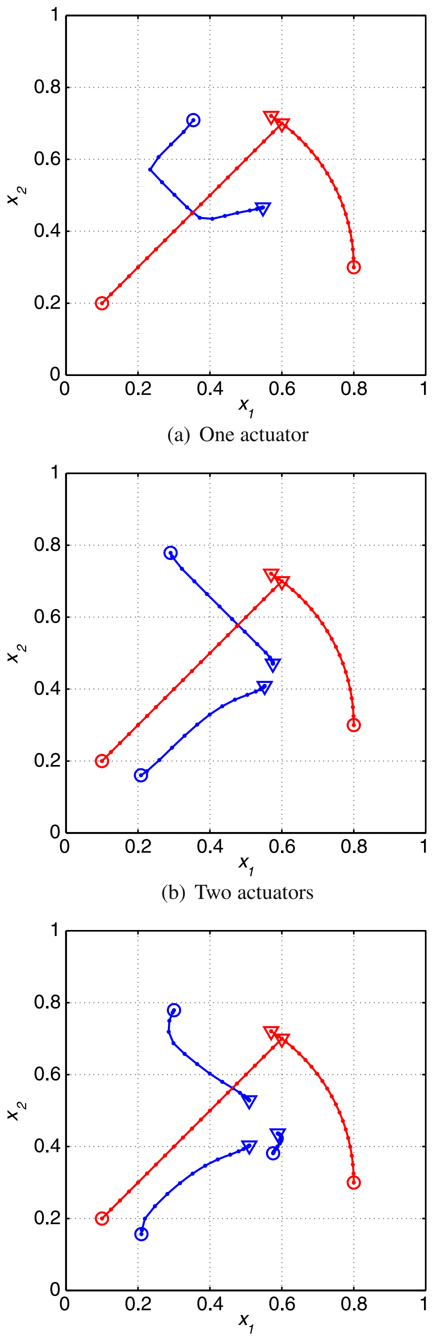


Fig. 6.5 D-optimum trajectories of mobile actuators for two mobile sensors

(a) One actuator

(b) Two actuators

(c) Three actuators

where b_{xi} are known continuously differentiable functions. That is to say, the following constraints have to be satisfied:

$$h_{ij}\big(s(t)\big) = b_x i\big(x_x^j(t)\big) \leq 0, \quad t \in T, \tag{6.30}$$

where $1 \leq i \leq I$ and $1 \leq j \leq (N + M)$. For simpler notation, we reformulate the conditions described in (6.30) in the following way:

$$\gamma_l\big(s(t)\big) \leq 0, \quad t \in T, \tag{6.31}$$

where $\gamma_l, l = 1, \dots, \nu$, tally with (6.30), $\nu = I \times (N + M)$. It would be possible to consider additional constraints on the path of the vehicles such as specific dynamics, collision avoidance, communication range maintenance, and any other conceivable constraints.

The actuation function for the kth mobile actuator is assumed to depend on the actuator's position as reflected by the following definition:

$$\mathcal{F}_k(x, t) = \mathcal{G}_k\big(x, x_a^k, t\big). \tag{6.32}$$

6.3.2 Problem Definition

The purpose of the optimal measurement/actuation problem is to determine the forces (controls) applied to each vehicle (conveying either a sensor or an actuator), which minimize the design criterion $\Psi(\cdot)$ defined on the FIMs of the form (6.15), which are determined by the corresponding sensor and actuator trajectories, subject to constraints on the magnitude of the controls and state constraints. To increase the degree of optimality, our approach considers s_0 as a control parameter vector to be optimized in addition to the control function $u = [u_s, u_a]^T$.

Given the above formulation, we can cast the optimal measurement/actuation policy problem as the following optimization problem: Find the pair (s_0, u) that minimizes

$$J(s_0, u) = \Phi\big[M(s)\big] \tag{6.33}$$

over the set of feasible pairs

$$\begin{aligned}\mathcal{P} = \big\{(s_0, u) | u : T \to \mathbb{R}^{r_s + r_a} \quad & \text{is measurable,} \\ u_l \leq u(t) \leq u_u \text{ a.e. on } T,\ & s_0 \in \Omega_{s\text{ad}} \times \Omega_{a\text{ad}}\big\},\end{aligned} \tag{6.34}$$

subject to the constraint (6.31).

6.3.3 An Illustrative Example

In this section, we use a demonstrative example to illustrate our method. We consider the two-dimensional diffusion equation

$$\frac{\partial y}{\partial t} = \nabla \cdot (\kappa \nabla y) + \sum_{k=1}^{M} F_k \tag{6.35}$$

for $\boldsymbol{x} = [x_1\ x_2]^T \in \Omega = (0, 1)^2$ and $t \in [0, 1]$, subject to homogeneous zero initial and Dirichlet boundary conditions. The spatial distribution of the diffusion coefficient is assumed to have the form

$$\kappa(x_1, x_2) = \theta_1 + \theta_2 x_1 + \theta_3 x_2. \tag{6.36}$$

In our example, we select the initial estimates of the parameter values as $\theta_1^0 = 0.1$, $\theta_2^0 = -0.05$, and $\theta_3^0 = 0.2$, which are assumed to be nominal and known prior to the experiment. The actuation function is

$$F_k\big(\boldsymbol{x}, \boldsymbol{x}_a^k, t\big) = 10 e^{-50((x_{a1}^k - x_1)^2 + (x_{a2}^k - x_2)^2)}, \tag{6.37}$$

where $\boldsymbol{x}_a^i = [x_{a1}^i\ x_{a2}^i]^T$. The dynamics of the mobile actuators follow the simple model

$$\dot{\boldsymbol{x}}_a^k(t) = \boldsymbol{u}_a^k(t), \qquad \boldsymbol{x}_a^k(0) = \boldsymbol{x}_{a0}^k, \tag{6.38}$$

and additional constraints

$$\big|u_{ai}^k(t)\big| \le 0.7, \quad t \in T,\ k = 1, \dots, M,\ i = 1, \dots, 2. \tag{6.39}$$

The dynamics of the mobile sensors follow the same model

$$\dot{\boldsymbol{x}}_s^j(t) = \boldsymbol{u}_s^j(t), \qquad \boldsymbol{x}_s^j(0) = \boldsymbol{x}_{s0}^j, \tag{6.40}$$

and additional constraints

$$\big|u_{si}^j(t)\big| \le 0.7, \quad t \in T,\ j = 1, \dots, N,\ i = 1, \dots, 2. \tag{6.41}$$

Our goal is to design their trajectories so as to obtain possibly the best estimates of θ_1, θ_2, and θ_3.

The strategy is tested on a simple team of one sensor and one actuator. In order to avoid getting stuck in a local minimum, computations were repeated several times from different initial solutions. Figure 6.6 presents the resulting trajectories for the run where the initial solutions lead to the best results (minimal value of the D-optimality criteria). Steering signals for both sensor and actuator are displayed in Figs. 6.7 and 6.8. Resulting trajectories for two sensors and one actuator are given in Fig. 6.9, and three sensors and one actuator in Fig. 6.10. Sensor trajectories are displayed in blue, while actuator trajectories are red.

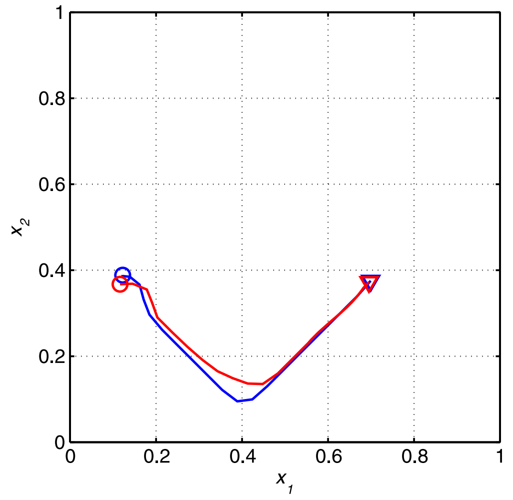


Fig. 6.6 D-optimal trajectories of a team of one mobile sensor and one mobile actuator. The initial positions are marked with *open circles*, and the final positions are designated by *triangles*

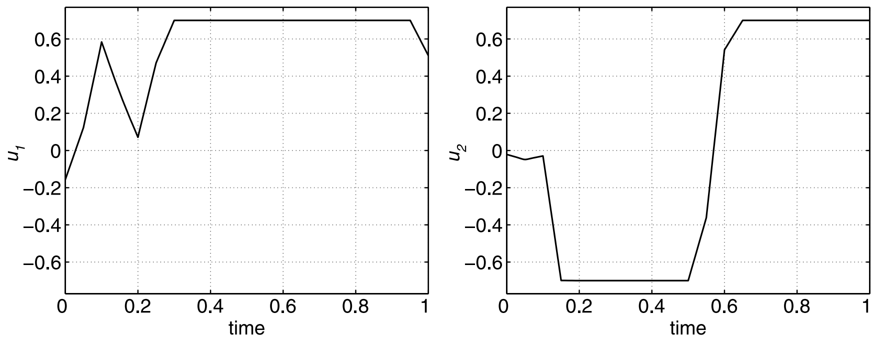


Fig. 6.7 Optimal control signal of the mobile sensor

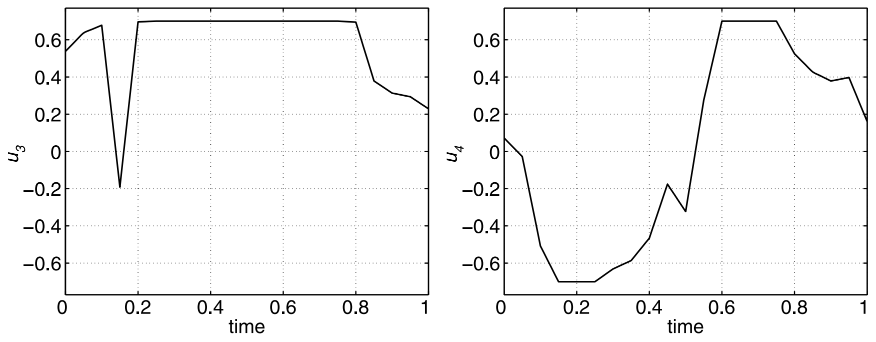


Fig. 6.8 Optimal control signal of the mobile actuator

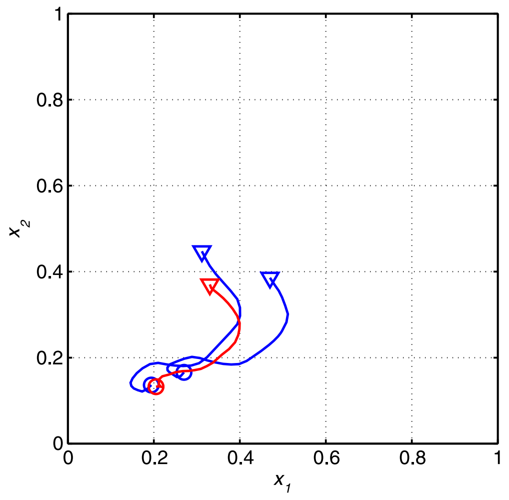


Fig. 6.9 D-optimal trajectories of a team of two mobile sensors and one mobile actuator

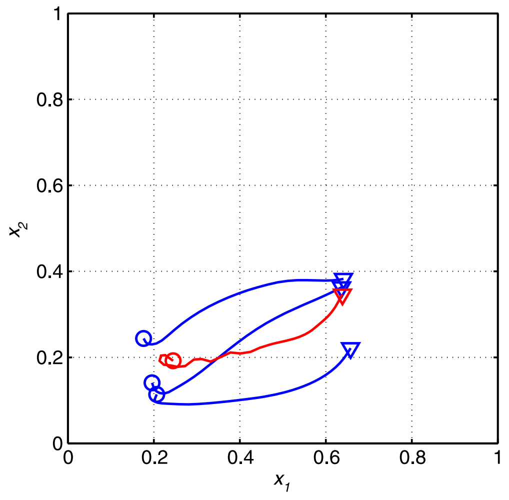


Fig. 6.10 D-optimal trajectories of a team of three mobile sensors and one mobile actuator

6.4 Chapter Summary

We introduced the optimal actuation framework for parameter identification in distributed parameter systems. The problem was formulated as an optimization problem using the concept of the Fisher information matrix. The problem was then reformulated into an optimal control one. With the help of the MATLAB PDE toolbox for the system simulations and RIOTS_95 MATLAB toolbox for solving the optimal control problem, we successfully obtained the optimal solutions for an illustrative example.

We introduced the optimal measurement/actuation framework for parameter identification in a cyber-physical system constituted of mobile sensors and actuators behaving in a distributed parameter system. The problem was formulated as an optimization problem using the concept of the Fisher information matrix. The problem was then reformulated into an optimal control one. We successfully obtained the optimal solutions for an illustrative example. Combined with the online scheme introduced in [141], this research represents a realistic example of a CPS. Mobile sensors and actuators are communicating to achieve the parameter estimation of the physical system that they are monitoring/stimulating. An exciting application consists of center-pivot operations, where our research center has a project of using camera-equipped unmanned aerial vehicles for soil moisture measurement combined with irrigators to stimulate the farming field. Thanks to this framework, an accurate model of the soil dynamics can be derived, and water savings can be obtained via optimal operations of the center pivot.

Chapter 7
Optimal Mobile Sensing with Fractional Sensor Dynamics

7.1 Introduction

The idea of fractional derivative dates back to a conversation between two mathematicians: Leibniz and L'Hôpital. In 1695, they exchanged about the meaning of a derivative of order $1/2$. Their correspondence has been well documented and is stated as the foundation of fractional calculus [108].

Many real-world physical systems display fractional-order dynamics, that is, their behavior is governed by fractional-order differential equations [110]. For example, it has been illustrated that materials with memory and hereditary effects, and dynamical processes, including gas diffusion and heat conduction, in fractal porous media can be more adequately modeled by fractional-order models than integer-order models [173].

The general definition of an optimal control problem requires minimization of a criterion function of the states and control inputs of the system over a set of admissible control functions. The system is subject to constrained dynamics and control variables. Additional constraints such as final time constraints can be considered. We introduce an original formulation and a general numerical scheme for a potentially almost unlimited class of FOCPs. An FOCP is an optimal control problem in which the criterion and/or the differential equations governing the dynamics of the system contain at least one fractional derivative operator.

Integer-order optimal controls (IOOCs) have been discussed for a long time, and a large collection of numerical techniques has been developed to solve IOOC problems. However, the number of publications on FOCPs is limited. A general formulation and a solution scheme for FOCPs were first introduced in [10], where fractional derivatives were introduced in the Riemann–Liouville sense, and FOCP formulation was expressed using the fractional variational principle and the Lagrange multiplier technique. The state and the control variables were given as a linear combination of test functions, and a virtual work type approach was used to obtain solutions. In [11, 12], the FOCPs were formulated using the definition of fractional derivatives in the sense of Caputo, the FDEs were substituted into Volterra-type integral equations, and a direct linear solver helped in calculating the solution of the obtained algebraic equations. In [13], the fractional dynamics of the FOCPs were defined in terms of the Riemann–Liouville fractional derivatives. The Grunwald–Letnikov formula was used as an approximation, and the resulting equations were solved using a direct scheme. Frederico and Torres [63–65], using similar definitions of the FOCPs, formulated a Noether-type theorem in the general context of the fractional optimal control in the sense of Caputo and studied fractional conservation laws in FOCPs. However, none of this work has taken advantage of the colossal research achieved in the numerical solutions of IOOCs.

In this chapter, we present a formulation and a numerical scheme for FOCP based on IOOC problem formulation. Therefore, the class of FOCP solvable by the proposed methodology is closely related to the considered IOOC solver RIOTS_95 [40, 126]. The fractional derivative operator is approximated in a frequency domain by using Oustaloup's Recursive Approximation, which results in a state space realization. The fractional differential equation governing the dynamics of the system is expressed as an integer-order state space realization. The FOCP can then be reformulated into an IOOC problem, solvable by a wide variety of algorithms from the literature. Three examples are solved to demonstrate the performance of the method. The work described here was first introduced in [142].

7.2 Fractional Optimal Control Problem Formulation

In this section, we briefly give some definitions regarding fractional derivatives, allowing us to formulate a general definition of an FOCP.

There are different definitions of the fractional derivative operator. The left Riemann–Liouville fractional derivative (LRLFD) of a function $f(t)$ is defined as

$$ {}_aD_t^{\alpha} f(t) = \frac{1}{\Gamma(n-\alpha)} \left(\frac{\mathrm{d}}{\mathrm{d}t}\right)^n \int_a^t (t-\tau)^{n-\alpha-1} f(\tau)\,\mathrm{d}\tau, \tag{7.1}$$

where the order of the derivative α satisfies $n-1 \le \alpha < n$. The right Riemann–Liouville fractional derivative (RRLFD) is defined as

$$ {}_tD_b^{\alpha} f(t) = \frac{1}{\Gamma(n-\alpha)} \left(-\frac{\mathrm{d}}{\mathrm{d}t}\right)^n \int_t^b (t-\tau)^{n-\alpha-1} f(\tau)\,\mathrm{d}\tau. \tag{7.2}$$

Another fractional derivative is the left Caputo fractional derivative (LCFD) defined as

$$ {}_a^CD_t^{\alpha} f(t) = \frac{1}{\Gamma(n-\alpha)} \int_a^t (t-\tau)^{n-\alpha-1} \left(\frac{\mathrm{d}}{\mathrm{d}t}\right)^n f(\tau)\,\mathrm{d}\tau. \tag{7.3}$$

The right Caputo fractional derivative (RCFD) is defined as

$$ {}_t^CD_b^{\alpha} f(t) = \frac{1}{\Gamma(n-\alpha)} \int_t^b (t-\tau)^{n-\alpha-1} \left(\frac{\mathrm{d}}{\mathrm{d}t}\right)^n f(\tau)\,\mathrm{d}\tau. \tag{7.4}$$

From any of these definitions, we can specify a general FOCP: Find the optimal control $u(t)$ for a fractional dynamical system that minimizes the following performance criterion:

$$J(u) = G\big(x(a), x(b)\big) + \int_a^b L(x, u, t)\,\mathrm{d}t, \tag{7.5}$$

subject to the system dynamics

$${}_aD_t^{\alpha} x(t) = H(x, u, t) \tag{7.6}$$

with initial condition

$$x(a) = x_a \tag{7.7}$$

and constraints

$$u_{\min}(t) \le u(t) \le u_{\max}(t), \tag{7.8}$$

$$x_{\min}(a) \le x(a) \le x_{\max}(a), \tag{7.9}$$

$$L_{ti}^{\nu}\big(t, x(t), u(t)\big) \le 0, \tag{7.10}$$

$$G_{ei}^{\nu}\big(x(a), x(b)\big) \le 0, \tag{7.11}$$

$$G_{ee}^{\nu}\big(x(a), x(b)\big) = 0, \tag{7.12}$$

where x is the state variable, $t \in [a, b]$ stands for the time, and F, G, and H are arbitrary given nonlinear functions. The subscripts ti, ei, and ee on the functions $G(\cdot, \cdot)$ and $L(\cdot, \cdot, \cdot)$ stand for, respectively, trajectory constraint, endpoint inequality constraint, and endpoint equality constraint.

7.3 Oustaloup Recursive Approximation of the Fractional Derivative Operator

Oustaloup recursive approximation (ORA) was introduced and is now utilized to approximate fractional-order transfer functions using a rational transfer function [36, 171]. The approximation is given by

$$s^{\alpha} = \prod_{n=1}^{N} \frac{1 + s/\omega_{z,n}}{1 + s/\omega_{p,n}}. \tag{7.13}$$

The resulting approximation is valid only within a frequency range $[\omega_l \ \omega_h]$. The number of poles and zeros N has to be decided beforehand, and the performance of the approximation strongly depends on its approximation parameter choice: small values of N cause low-order, simpler approximations. Consequently, the Bode diagram exhibits undulations in both phase and gain responses around the real response. Such undulations can easily be removed by increasing the value of N, at the cost of higher order and increased amount of calculations. Frequencies of poles and zeros in (7.13) are obtained, given α, N, ω_l, and ω_h, by [173]:

$$\omega_{z,1} = \omega_l \sqrt{\eta}, \tag{7.14}$$

$$\omega_{p,n} = \omega_{z,n}\varepsilon; \quad n \in [1\ N], \tag{7.15}$$

$$\omega_{z,n+1} = \omega_{p,n}\eta; \quad n \in [1\ N-1], \tag{7.16}$$

$$\varepsilon = (\omega_h/\omega_l)^{\alpha/N}, \tag{7.17}$$

$$\eta = (\omega_l/\omega_h)^{(1-\alpha)/N}. \tag{7.18}$$

When $\alpha < 0$, inverting (7.13) helps in obtaining the approximation. For $|\alpha| > 1$, our definition does not hold anymore. A practical solution is to separate the fractional orders of s in the following way:

$$s^\alpha = s^n s^\delta; \quad \alpha = n + \delta; \quad n \in \mathbb{Z}; \quad \delta \in [0, 1]. \tag{7.19}$$

Under such a condition, only s^δ needs to be approximated. Discrete approximation for the fractional differentiation operator can be found in [35].

For FOCP, such a definition of ORA as a zero-pole transfer function is not helpful. Instead, a state space realization of the approximation is required. The first step toward a state space realization is to expand the transfer function given in (7.13):

$$s^\alpha = \frac{\sum_{i=0}^{N} a_i s^i}{\sum_{j=0}^{N} b_j s^j}, \tag{7.20}$$

where

$$a_i = \sum_{k=i}^{N} \prod_{l=0}^{k} \frac{1}{\omega(z,l)} \tag{7.21}$$

and

$$b_j = \sum_{k=j}^{N} \prod_{l=0}^{k} \frac{1}{\omega(p,l)}. \tag{7.22}$$

Equation (7.20) can further be modified to match the following definition:

$$s^\alpha = \frac{\sum_{i=0}^{N-1} c_i s^i}{\sum_{j=0}^{N} b_j s^j} + d \tag{7.23}$$

with $b_N = 1$. It is finally possible to approximate the operator s^α using a state space definition

$$ {}_aD_t^\alpha x(t) \approx \left\{ \begin{array}{l} \dot{z} = Az + Bu \\ x = Cz + Du \end{array} \right\} \tag{7.24}$$

with

$$A = \begin{bmatrix} -b_{N-1} & -b_{N-2} & \cdots & -b_1 & -b_0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}, \tag{7.25}$$

$$B = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \tag{7.26}$$

$$C = \begin{bmatrix} c_{N-1} & c_{N-2} & \cdots & c_1 & c_0 \end{bmatrix}, \tag{7.27}$$

$$D = d. \tag{7.28}$$

7.4 Fractional Optimal Control Problem Reformulation-I

With our state space approximation of the fractional derivative operator, it is now possible to reformulate the FOCP described in (7.5)–(7.12). Find the optimal control $u(t)$ for a dynamical system that minimizes the performance criterion

$$J(u) = G\big(Cz(a) + Du(a), Cz(b) + Du(b)\big) + \int_a^b L(Cz + Du, u, t)\,\mathrm{d}t, \tag{7.29}$$

subject to the dynamics

$$\dot{z}(t) = Az + B\big(H(Cz + Du, u, t)\big) \tag{7.30}$$

with initial condition

$$z(a) = x_a w/(Cw) \tag{7.31}$$

and constraints

$$u_{\min}(t) \le u(t) \le u_{\max}(t), \tag{7.32}$$

$$x_{\min}(a) \le Cz(a) + Du(a) \le x_{\max}(a), \tag{7.33}$$

$$L_{ti}^{\nu}\big(t, Cz(t) + Du(t), u(t)\big) \le 0, \tag{7.34}$$

$$G_{ei}^{\nu}\big(Cz(a)+Du(a), Cz(b)+Du(b)\big) \le 0, \tag{7.35}$$

$$G_{ee}^{\nu}\big(Cz(a)+Du(a), Cz(b)+Du(b)\big) = 0, \tag{7.36}$$

where z is the state vector, w is a vector of size N, $t \in [a,b]$ stands for the time, and F, G, and H are arbitrary nonlinear functions. The subscripts ti, ei, and ee on the functions $G(\cdot,\cdot)$ and $L(\cdot,\cdot,\cdot)$ stand for, respectively, trajectory constraint, endpoint inequality constraint, and endpoint equality constraint.

The choice for the vector w is indeed important as it can improve the convergence of the optimization. Since B has the form given in (7.26), our method here is to choose w as

$$w = [1 \quad 0 \quad \cdots \quad 0]^T. \tag{7.37}$$

The state $x(t)$ of the initial FOCP can be retrieved from

$$x(t) = Cz(t) + Du(t). \tag{7.38}$$

The choice of $[\omega_l, \omega_h]$ needs to be carefully taken into consideration as a narrow bandwidth may lead to inaccurate results because of possible missing dynamics, and a large bandwidth would create a large computational burden as N would increase. The choice of N is not considered here as we use the rule of thumb $N = \log(\omega_h) - \log(\omega_l)$.

This framework allows us to approximately solve a large variety of FOCPs thanks to the link we created with the traditional optimal control problems. In fact, the proposed conversion allows us to apply any readily available IOOC solver to find an approximate solution of almost any given FOCP problem. We decide to use the RIOTS_95 MATLAB toolbox.

7.5 Impulse-Response-Based Linear Approximation of Fractional Transfer Functions

7.5.1 *Approximation Method*

This methodology was derived from [76]. Consider the analytical impulse response $h(t)$ of a given fractional system. The approximation problem occurs in obtaining a linear system of order n whose impulse response $h_a(t)$ coincides with $h(t)$ well. The linear system is modeled by the following state space realization:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + bu(t), \tag{7.39}$$

$$y(t) = c\mathbf{x}(t), \tag{7.40}$$

where the state $\mathbf{x}(t)$ is of size n, and the system matrix A is n by n. The impulse response $h_a(t)$ can be expressed in terms of A, b, and c by [165]

$$h(t) = ce^{At}b, \tag{7.41}$$

where the state-transition matrix e^{At} denotes the exponential of the matrix At. Let us describe the methodology for solving the approximation problem. We consider a set of sampled data $h(kT)$ from the analytical impulse response $h(t)$, with T standing for the sampling period. An approximate linear system would have the following property:

$$h(kT) \approx ce^{AkT}b = c\left(e^{AT}\right)^k b, \tag{7.42}$$

which can be reformulated in the following way:

$$h(kT) \approx c\left(e^{A_d}\right)^k b \tag{7.43}$$

with

$$A_d = e^{AT}. \tag{7.44}$$

We then take $2p$ data points from the sampled impulse response to form a Hankel data matrix H defined as

$$H = \begin{pmatrix} h(0) & h(1) & \dots & h(p-1) \\ h(1) & h(2) & \dots & h(p) \\ \vdots & \vdots & \dots & \vdots \\ h(p) & h(p-1) & \dots & h(2p-1) \end{pmatrix}_{p+1,p}, \tag{7.45}$$

that is,

$$H = \begin{pmatrix} cb & cA_d b & \dots & cA_d^{p-1}b \\ cA_d b & cA_d^2 b & \dots & cA_d^p b \\ \vdots & \vdots & \dots & \vdots \\ cA_d^p b & cA_d^{p+1} b & \dots & cA_d^{2p-1} b \end{pmatrix}. \tag{7.46}$$

H is further reformulated by the factorization

$$H = \begin{pmatrix} c \\ cA_d \\ \vdots \\ cA_d^{p-1} \end{pmatrix}_{p+1,n} \begin{pmatrix} b & A_d b & \dots & A_d^{p-1} b \end{pmatrix} = OC, \tag{7.47}$$

where n is the approximated numerical rank of the Hankel data matrix H and is determined by its singular values (square roots of eigenvalues of HH^T). By examining singular values of H, we are able to choose a proper integer n to be the dimension of the approximating linear system. In other words, n is the number of state variables of the linear system that are adequate in describing the distributed system specified by $h(t)$. Since the matrix H is given, factorization of H into a product of two matrices is always possible using the singular value decomposition. After O and C

are generated from the Hankel data matrix, matrices A, b, and c can be obtained as follows:

$$c = \text{1st row of } O, \tag{7.48}$$

$$b = \text{1st column of } C. \tag{7.49}$$

Define

$$O_1 = O \text{ without the last row}, \tag{7.50}$$

$$O_2 = O \text{ without the first row}. \tag{7.51}$$

Then

$$O_2 = O_1 A_d. \tag{7.52}$$

Solving the above equation yields

$$A_d = \left(O_1^T O_1\right)^{-1} O_1^T O_2. \tag{7.53}$$

Finally, we recall the relationship $A = e^{AT}$ and obtain A from A_d by

$$A = \ln(A_d)/T, \tag{7.54}$$

where ln denotes the natural log of a matrix.

7.5.2 Suboptimal Approximation of the Fractional Integrator

We try to approximate the fractional transfer function

$$H(s) = \frac{1}{s^{\alpha}} \tag{7.55}$$

with $\alpha \in [0, 1]$. The analytical impulse response of such a system is given by

$$h(t) = \frac{t^{-\alpha-1}}{\Gamma(-\alpha)}, \tag{7.56}$$

where $\Gamma(\cdot)$ represents the Gamma function. For a given transfer function, an infinite number of approximations can be performed. Therefore, for a given order n of the state space realization of the approximation, we wish to find the values of T and p that give the best approximation. In addition, the impulse response of a fractional integrator displays a singularity at the origin ($t = 0$) as observed in (7.56). Therefore, to avoid this infinite term, $h(0)$ has to be approximated by a finite value. This finite initial value giving the best approximation is also sought. The best approximation is obtained via an exhaustive search. The performance criterion used to

Table 7.1 Parameter values used for the exhaustive search of the best approximation

T	10^{-3}	5×10^{-4}	10^{-4}	5×10^{-5}	10^{-5}
	5×10^{-6}	10^{-6}	5×10^{-7}	10^{-7}	
p	25	50	75	100	250
	500	750	1000		
$h(0)$	$10\cdot h(1)$	$10^2\cdot h(1)$	$10^3\cdot h(1)$	$10^4\cdot h(1)$	

assess the quality of an approximation is the ITSE of the step response because of the absence of singularity and improved results. The analytical step response of the system described by (7.55) is

$$s(t)=\frac{t^{-\alpha}}{\Gamma(-\alpha+1)}. \tag{7.57}$$

The search is performed for approximation orders n ranging from 1 to 10. Table 7.1 summarizes the different values used in the search for the best parameters set, and Table 7.2 gives the obtained ITSE for each order of approximation and for each order of derivation. These values were upper bounded by the computer's memory.

7.6 Fractional Optimal Control Problem Reformulation-II

With our state space approximation of the fractional derivative operator, it is now possible to reformulate the FOCP described in (7.5)–(7.12). Find the optimal control $u(t)$ for a dynamical system that minimizes the performance criterion

$$J(u)=G\big(cz(a),cz(b)\big)+\int_a^b L(cz,u,t)\,\mathrm{d}t, \tag{7.58}$$

subject to the dynamics

$$\dot z(t)=Az+b\big(H(cz,u,t)\big) \tag{7.59}$$

with initial condition

$$z(a)=x_a w/(cw). \tag{7.60}$$

Equation (7.60) ensures that the initial condition $cz(a)=x_a$ is maintained with the following constraints:

$$u_{\min}(t)\le u(t)\le u_{\max}(t), \tag{7.61}$$

$$x_{\min}(a)\le Ccz(a)\le x_{\max}(a), \tag{7.62}$$

$$L_{ti}^{\nu}\big(t,cz(t),u(t)\big)\le 0, \tag{7.63}$$

Table 7.2 ITSE of the best model for different approximation orders and fractional orders

ITSE	$\alpha=0.1$	$\alpha=0.2$	$\alpha=0.3$	$\alpha=0.4$	$\alpha=0.5$
$n=1$	$1.05e{-}2$	$2.62e{-}3$	$6.26e{-}2$	$1.77e{-}2$	$3.12e{-}3$
$n=2$	$3.61e{-}4$	$1.39e{-}3$	$1.56e{-}3$	$6.47e{-}4$	$8.50e{-}4$
$n=3$	$2.81e{-}4$	$1.34e{-}3$	$1.40e{-}4$	$5.40e{-}5$	$1.49e{-}4$
$n=4$	$3.45e{-}5$	$1.30e{-}3$	$1.01e{-}4$[a]	$2.01e{-}6$	$4.22e{-}5$
$n=5$	$3.25e{-}6$[a]	$1.25e{-}3$	$2.27e{-}4$	$1.49e{-}6$	$1.86e{-}5$
$n=6$	$8.40e{-}6$	$1.25e{-}3$	$3.13e{-}4$	$1.51e{-}7$[a]	$1.15e{-}5$
$n=7$	$2.80e{-}5$	$1.31e{-}3$	$3.61e{-}4$	$1.41e{-}6$	$6.09e{-}6$
$n=8$	$2.00e{-}4$	$1.14e{-}3$	$3.92e{-}4$	$3.26e{-}6$	$2.94e{-}6$
$n=9$	$4.33e{-}4$	$8.96e{-}4$	$4.14e{-}4$	$5.10e{-}6$	$2.53e{-}6$
$n=10$	$6.80e{-}4$	$7.55e{-}4$[a]	$4.32e{-}4$	$6.84e{-}6$	$2.46e{-}6$[a]

ITSE	$\alpha=0.6$	$\alpha=0.7$	$\alpha=0.8$	$\alpha=0.9$
$n=1$	$2.32e{-}3$	$8.98e{-}4$	$4.72e{-}4$	$1.85e{-}4$
$n=2$	$7.07e{-}4$	$3.96e{-}4$	$2.10e{-}4$	$6.01e{-}5$
$n=3$	$9.39e{-}5$	$4.47e{-}5$	$1.56e{-}5$	$3.42e{-}6$
$n=4$	$2.89e{-}5$	$1.26e{-}5$	$4.12e{-}6$	$1.02e{-}6$
$n=5$	$1.51e{-}5$	$6.93e{-}6$	$2.40e{-}6$	$6.77e{-}7$
$n=6$	$1.01e{-}5$	$5.35e{-}6$	$1.93e{-}6$	$4.00e{-}7$
$n=7$	$4.50e{-}6$	$2.31e{-}6$	$8.51e{-}7$	$2.38e{-}7$[a]
$n=8$	$3.85e{-}6$[a]	$2.06e{-}6$[a]	$7.80e{-}7$[a]	$2.53e{-}7$
$n=9$	$4.17e{-}6$	$2.28e{-}6$	$8.71e{-}7$	$2.78e{-}7$
$n=10$	$4.39e{-}6$	$2.44e{-}6$	$9.29e{-}7$	$2.89e{-}7$

[a]Indicates the best approximate

$$G_{ei}^{\nu}\big(cz(a), cz(b)\big) \le 0, \tag{7.64}$$

$$G_{ee}^{\nu}\big(cz(a), cz(b)\big) = 0, \tag{7.65}$$

where z is the state vector, w is a vector of size N, $t \in [a,b]$ stands for the time, and F, G, and H are arbitrary nonlinear functions. The subscripts ti, ei, and ee on the functions $G(\cdot,\cdot)$, and $L(\cdot,\cdot,\cdot)$ stand for, respectively, “trajectory constraint”, “endpoint inequality constraint”, and “endpoint equality constraint”.

The choice for the vector w is indeed important as it can improve the convergence of the optimization. To make computation faster, our experiments have shown that

$$w = [1 \quad 0 \quad \cdots \quad 0]^T \tag{7.66}$$

represents the best choice. The state $x(t)$ of the initial FOCP can be retrieved from

$$x(t) = cz(t). \tag{7.67}$$

7.7 Illustrative Examples

In this section, we demonstrate the capability of the introduced approach. First, we solve two widely used examples from the literature, and then we introduce a new problem that none of the previously introduced methodologies attempted to solve. For each problem, we examine the solution for different values of α. For this purpose, α was taken between 0.1 and 1. Problems are first stated in the traditional FOCP framework and then reformulated via our introduced methodology. Results of these studies are given at the end of each subsection.

7.7.1 A Linear Time-Invariant Problem

Our first example can be found in [10, 12, 13, 142]. It is a linear time-invariant (LTI) fractional-order optimal control problem stated as follows. Find the control $u(t)$ which minimizes the quadratic performance index

$$J(u) = \frac{1}{2}\int_0^1 \left[x^2(t) + u^2(t)\right] \mathrm{d}t, \tag{7.68}$$

subject to the dynamics

$${}_0D_t^\alpha x = -x + u \tag{7.69}$$

with free terminal condition and the initial condition

$$x(0) = 1. \tag{7.70}$$

According to [9], the analytical solution of the problem defined above for $\alpha = 1$ is

$$x(t) = \cosh(\sqrt{2}t) + \beta \sinh(\sqrt{2}t), \tag{7.71}$$

$$u(t) = (1 + \sqrt{2}\beta)\cosh(\sqrt{2}t) + (\sqrt{2} + \beta)\sinh(\sqrt{2}t), \tag{7.72}$$

where

$$\beta = -\frac{\cosh(\sqrt{2}) + \sqrt{2}\sinh(\sqrt{2}t)}{\sqrt{2}\cosh(\sqrt{2}) + \sinh(\sqrt{2}t)} \approx -0.98.$$

Using the methodology we just introduced, we reformulate the problem defined by (7.68)–(7.70). Find the control $u(t)$ which minimizes the quadratic performance index

$$J(u) = \frac{1}{2}\int_0^1 \left(cz(t)\right)^2 + u^2(t)\, \mathrm{d}t, \tag{7.73}$$

subject to the dynamics

$$\dot{z} = Az + B\left(-(cz) + u\right) \tag{7.74}$$

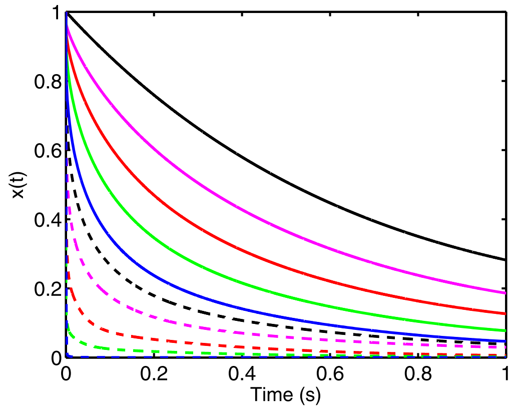


Fig. 7.1 State $x(t)$ as a function of t for the LTI problem for different α (*dashed blue*: $\alpha = 0.1$, *dashed green*: $\alpha = 0.2$, *dashed red*: $\alpha = 0.3$, *dashed magenta*: $\alpha = 0.4$, *dashed black*: $\alpha = 0.5$, *solid blue*: $\alpha = 0.6$, *solid green*: $\alpha = 0.7$, *solid red*: $\alpha = 0.8$, *solid magenta*: $\alpha = 0.9$, *solid black*: $\alpha = 1$)

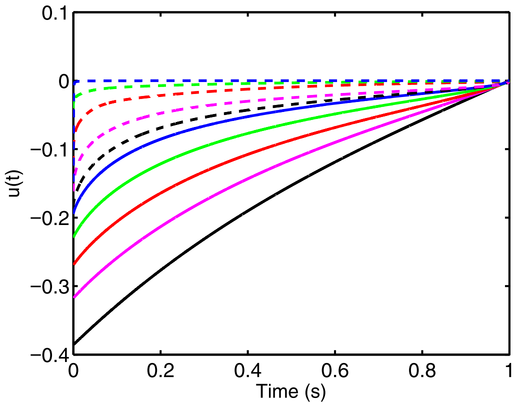


Fig. 7.2 Control $u(t)$ as a function of t for the LTI problem for different α (*dashed-blue*: $\alpha = 0.1$, *dashed green*: $\alpha = 0.2$, *dashed red*: $\alpha = 0.3$, *dashed magenta*: $\alpha = 0.4$, *dashed black*: $\alpha = 0.5$, *solid blue*: $\alpha = 0.6$, *solid green*: $\alpha = 0.7$, *solid red*: $\alpha = 0.8$, *solid magenta*: $\alpha = 0.9$, *solid black*: $\alpha = 1$)

and the initial condition

$$z(0) = [1 \quad 0 \quad \cdots \quad 0]^T. \tag{7.75}$$

Figures 7.1 and 7.2 show the state $x(t)$ and the control input $u(t)$ as functions of time t for different values of α. For $\alpha = 1$, the results match those of the analytical solution. Results are comparable to those obtained in [10, 13, 142].

7.7.2 A Linear Time-Variant Problem

The second example studied here is also studied in [10, 12, 13, 142]. It is a linear time-variant (LTV) problem stated as follows. Find the control $u(t)$ which minimizes the quadratic performance index

$$J(u) = \frac{1}{2}\int_0^1 \left[x^2(t) + u^2(t)\right] \mathrm{d}t, \tag{7.76}$$

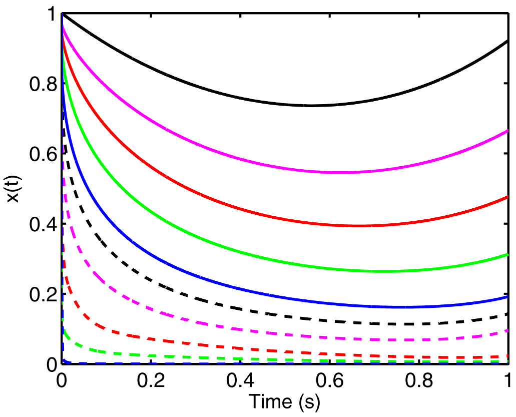


Fig. 7.3 State $x(t)$ as a function of t for the LTV problem for different α (*dashed blue*: $\alpha = 0.1$, *dashed green*: $\alpha = 0.2$, *dashed red*: $\alpha = 0.3$, *dashed magenta*: $\alpha = 0.4$, *dashed black*: $\alpha = 0.5$, *solid blue*: $\alpha = 0.6$, *solid green*: $\alpha = 0.7$, *solid red*: $\alpha = 0.8$, *solid magenta*: $\alpha = 0.9$, *solid black*: $\alpha = 1$)

subject to the dynamics

$$ {}_0D_t^{\alpha} x = tx + u \tag{7.77}$$

with free terminal condition and the initial condition

$$x(0) = 1. \tag{7.78}$$

Using the proposed methodology, we reformulate the problem defined by (7.76)–(7.78). Find the control $u(t)$ which minimizes the quadratic performance index

$$J(u) = \frac{1}{2}\int_0^1 \big(cz(t)\big)^2 + u^2(t)\,\mathrm{d}t, \tag{7.79}$$

subject to the dynamics

$$\dot{z} = Az + b\big((cz)t + u\big) \tag{7.80}$$

and the initial condition

$$z(0) = [1 \quad 0 \quad \cdots \quad 0]^T. \tag{7.81}$$

Figures 7.3 and 7.4 show the state $x(t)$ and the control $u(t)$ as functions of t for different values of α $(0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1)$. For $\alpha = 1$, the optimal control problem has been solved in [9]. In that paper, the author used a scheme specific to integer-order optimal control problems. The numerical solution obtained with the proposed methodology for $\alpha = 1$ is accurate, and results for fractional orders of α match those found in the literature [10, 13, 142].

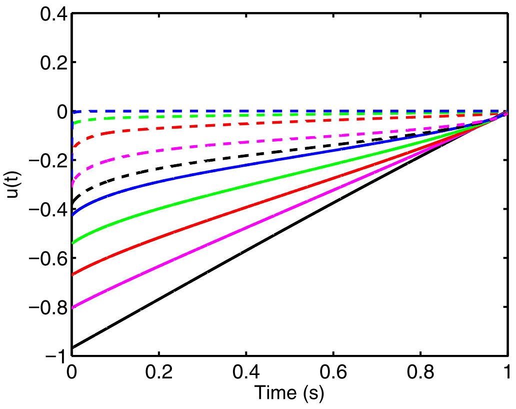


Fig. 7.4 Control $u(t)$ as a function of t for the LTV problem for different α (*dashed blue*: $\alpha = 0.1$, *dashed green*: $\alpha = 0.2$, *dashed red*: $\alpha = 0.3$, *dashed magenta*: $\alpha = 0.4$, *dashed black*: $\alpha = 0.5$, *solid blue*: $\alpha = 0.6$, *solid green*: $\alpha = 0.7$, *solid red*: $\alpha = 0.8$, *solid magenta*: $\alpha = 0.9$, *solid black*: $\alpha = 1$)

7.8 Optimal Mobile Sensing Policies with Fractional Sensor Dynamics

7.8.1 *Sensor Dynamics*

We assume that both sensors and actuators are equipped on vehicles whose dynamics can be described by the following differential equation:

$$ {}_aD_t^{\alpha} \boldsymbol{x}^j(t) = \boldsymbol{f}\big(\boldsymbol{x}^j(t), \boldsymbol{u}^j(t)\big) \quad \text{a.e. on } T \quad \boldsymbol{x}^j(0) = \boldsymbol{x}_0^j. \tag{7.82}$$

With this nomenclature, the function $\boldsymbol{f}$ has to be continuously differentiable, the vector $\boldsymbol{x}_0^j$ represents the initial disposition of the jth sensor, and $\boldsymbol{u}$ is a measurable control function satisfying the following inequality:

$$\boldsymbol{u}_l \le \boldsymbol{u}(t) \le \boldsymbol{u}_u \quad \text{a.e. on } T \tag{7.83}$$

for some known constant vectors $\boldsymbol{u}_l$ and $\boldsymbol{u}_u$. Let us introduce

$$\boldsymbol{s}(t) = \big(\boldsymbol{x}^1(t), \boldsymbol{x}^2(t), \dots, \boldsymbol{x}^N(t)\big)^T, \tag{7.84}$$

where $\boldsymbol{x}^j : T \to \Omega_{\text{ad}}$ is the trajectory of the jth sensor. We define $\boldsymbol{s}_0 = \boldsymbol{s}(0)$ as the initial location of the mobile sensors. We assume that all the mobile nodes equipped with sensors are confined within an admissible region Ω_{ad} (a given compact set) where the measurements are possible. Considering the general index defined earlier, Ω_{ad} can be conveniently defined as

$$\Omega_{\text{ad}} = \big\{\boldsymbol{x} \in \Omega : b_i(\boldsymbol{x}) = 0,\ i = 1, \dots, I\big\}, \tag{7.85}$$

where b_i are known continuously differentiable functions. That is to say, the following constraints have to be satisfied:

$$h_{ij}\big(\boldsymbol{s}(t)\big) = b_i\big(\boldsymbol{x}^j(t)\big) \le 0, \quad t \in T, \tag{7.86}$$

where $1 \le i \le I$ and $1 \le j \le N$. For simpler notation, we reformulate the conditions described in (7.86) in the following way:

$$\gamma_l\big(\boldsymbol{s}(t)\big) \le 0, \quad t \in T, \tag{7.87}$$

where $\gamma_l,\ l = 1, \ldots, \nu$, tally with (7.86), $\nu = I \times N$. It would be possible to consider additional constraints on the path of the vehicles such as specific dynamics, collision avoidance, communication range maintenance, and any other conceivable constraints.

7.8.2 Optimal Measurement Problem

The measurement problem for bounded parameter values can be defined by reformulating the FIM associated with the problem in the following way:

$$\boldsymbol{M} = \sum_{j=1}^{N} \int_T \boldsymbol{g}\big(\boldsymbol{x}_s^j(t), t\big) \boldsymbol{g}^\mathsf{T}\big(\boldsymbol{x}_s^j(t), t\big)\,\mathrm{d}t, \tag{7.88}$$

where

$$\boldsymbol{g}(\boldsymbol{x}, t) = \nabla_{\boldsymbol{\theta}} y(\boldsymbol{x}, t; \boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}^0} \tag{7.89}$$

denotes the vector of the so-called sensitivity coefficients, $\boldsymbol{\theta}^0$ being a prior estimate to the unknown parameter vector $\boldsymbol{\theta}$.

The purpose of the optimal measurement problem is to determine the forces (controls) applied to each vehicle that minimize the design criterion $\Psi(\cdot)$ defined on the FIMs of the form (7.88), which are determined unequivocally by the corresponding trajectories, subject to constraints on the magnitude of the controls and induced state constraints. To increase the degree of optimality, our approach considers s_0 as a control parameter vector to be optimized in addition to the control function $\boldsymbol{u}$.

Given the above formulation, we can cast the optimal measurement policy problem as the following optimization problem: Find the pair $(\boldsymbol{s}_0, \boldsymbol{u})$ that minimizes

$$J(\boldsymbol{s}_0, \boldsymbol{u}) = \Phi\big[\boldsymbol{M}(\boldsymbol{s})\big] \tag{7.90}$$

over the set of feasible pairs

$$\begin{aligned} \mathcal{P} = \big\{(\boldsymbol{s}_0, \boldsymbol{u}) | \boldsymbol{u} : T \to \mathbb{R}^r \quad &\text{is measurable,} \\ &\boldsymbol{u}_l \le \boldsymbol{u}(t) \le \boldsymbol{u}_u \text{ a.e. on } T, \boldsymbol{s}_0 \in \Omega_{\mathrm{ad}}\big\}, \end{aligned} \tag{7.91}$$

subject to the constraint (7.87).

7.8.3 Optimal Control Problem Reformulation

The problem is converted into a canonical optimal control one, making possible the use of existing optimal control problem solvers. The first step consists of approximating the fractional operator using a rational approximation. It is possible to approximate the operator ${}_aD_t^\alpha$ using a state space definition:

$$ {}_aD_t^\alpha x = f(x,u,t) \Leftrightarrow \begin{Bmatrix} \dot{z} = Az + bf(cz,u,t) \\ x = cz \end{Bmatrix}. \tag{7.92}$$

The dynamics of the mobile sensors can hence be written as

$$\begin{cases} \dot{\boldsymbol{z}}^j(t) = A\boldsymbol{z}^j(t) + b\boldsymbol{f}(c\boldsymbol{z}^j(t), \boldsymbol{u}^j(t)), \\ \boldsymbol{x}^j(t) = c\boldsymbol{z}^j(t). \end{cases} \tag{7.93}$$

Accordingly, a new experiment $\boldsymbol{s}_z(t)$ can be defined as

$$\boldsymbol{s}_z(t) = \big(\boldsymbol{z}^1(t), \boldsymbol{z}^2(t), \dots, \boldsymbol{z}^N(t)\big)^T, \tag{7.94}$$

and $\boldsymbol{s}(t)$ can be recovered by

$$\boldsymbol{s}(t) = \big(c\boldsymbol{z}^1(t), c\boldsymbol{z}^2(t), \dots, c\boldsymbol{z}^N(t)\big)^T \tag{7.95}$$

and

$$\boldsymbol{s}_0 = c\boldsymbol{s}_z(0). \tag{7.96}$$

We also define the function $\boldsymbol{f}_z$ given as

$$\dot{\boldsymbol{s}}_z(t) = \boldsymbol{f}_z\big(\boldsymbol{s}_z(t), \boldsymbol{u}(t), t\big), \tag{7.97}$$

so that the experiment $\boldsymbol{s}_z(t)$ can be recovered from the control input $\boldsymbol{u}(t)$.

Consider the matrix-valued function

$$\Pi\big(\boldsymbol{s}_z(t), t\big) = \sum_{j=1}^{N} \boldsymbol{g}\big(c\boldsymbol{z}^j(t), t\big)\boldsymbol{g}^T\big(c\boldsymbol{z}^j(t), t\big). \tag{7.98}$$

Setting $r : T \to \mathbb{R}^{m(m+1)/2}$ as the solution of the differential equations

$$\dot{\boldsymbol{r}}(t) = \operatorname{svec}\Big(\Pi\big(\boldsymbol{s}_z(t), t\big)\Big), \quad \boldsymbol{r}(0) = 0, \tag{7.99}$$

we obtain

$$M(\boldsymbol{s}_z) = \operatorname{Smat}\big(\boldsymbol{r}(t_f)\big), \tag{7.100}$$

i.e., minimization of $\Phi[M(\boldsymbol{s})]$ thus reduces to minimization of a function of the terminal value of the solution to (7.99). Introducing an augmented state vector

$$\boldsymbol{q}(t) = \begin{bmatrix} \boldsymbol{s}_z(t) \\ \boldsymbol{r}(t) \end{bmatrix}, \tag{7.101}$$

we obtain

$$\boldsymbol{q}_0 = \boldsymbol{q}(0) = \begin{bmatrix} \boldsymbol{s}_{z0} \\ \boldsymbol{0} \end{bmatrix}. \tag{7.102}$$

Then, the equivalent canonical optimal control problem occurs in finding a pair $(\boldsymbol{q}_0, \boldsymbol{u}) \in \bar{\mathcal{P}}$ that minimizes the performance index

$$\bar{J}(\boldsymbol{q}_0, \boldsymbol{u}) = \phi\big(\boldsymbol{q}(t_f)\big), \tag{7.103}$$

subject to

$$\begin{cases} \dot{\boldsymbol{q}}(t) = \phi(\boldsymbol{q}(t), \boldsymbol{u}(t), t), \\ \boldsymbol{q}(0) = \boldsymbol{q}_0, \\ \bar{\gamma}_l(\boldsymbol{q}(t)) \le 0, \end{cases} \tag{7.104}$$

where

$$\begin{aligned} \bar{\mathcal{P}} = \big\{(\boldsymbol{q}_0, \boldsymbol{u}) | \boldsymbol{u} : T \to \mathbb{R}^r \quad & \text{is measurable,} \\ & \boldsymbol{u}_l \le \boldsymbol{u}(t) \le \boldsymbol{u}_u \text{ a.e. on } T, c\boldsymbol{s}_{z0} \in \Omega_{\text{ad}}^N\big\}, \end{aligned} \tag{7.105}$$

and

$$\phi(\boldsymbol{q}, \boldsymbol{u}, t) = \begin{bmatrix} \boldsymbol{f}_z(\boldsymbol{s}_z(t), \boldsymbol{u}(t), t) \\ \operatorname{svec}(\Pi(\boldsymbol{s}_z(t), t)) \end{bmatrix}, \tag{7.106}$$

$$\bar{\gamma}_l\big(\boldsymbol{q}(t)\big) = \gamma_l\big(c\boldsymbol{s}_z(t)\big). \tag{7.107}$$

7.8.4 An Illustrative Example

We consider again the following two-dimensional diffusion equation:

$$\frac{\partial y}{\partial t} = \nabla \cdot (\kappa \nabla y) + 20 \exp\big(-50(x_1 - t)^2\big) \tag{7.108}$$

for $\boldsymbol{x} = [x_1\ x_2]^T \in \Omega = (0, 1)^2$ and $t \in [0, 1]$, subject to homogeneous zero initial and Dirichlet boundary conditions. The spatial distribution of the diffusion coefficient is assumed to have the form

$$\kappa(x_1, x_2) = \theta_1 + \theta_2 x_1 + \theta_3 x_2. \tag{7.109}$$

In this example, the chosen values for the parameter are $\theta_1 = 0.1$, $\theta_2 = -0.05$, and $\theta_3^0 = 0.2$, which are assumed to be known prior to the experiment. The dynamics of the mobile sensors follow the following model:

$${}_aD_t^\alpha \boldsymbol{x}^j(t) = \boldsymbol{u}^j(t), \quad \boldsymbol{x}^j(0) = \boldsymbol{x}_0^j, \tag{7.110}$$

and additional constraints

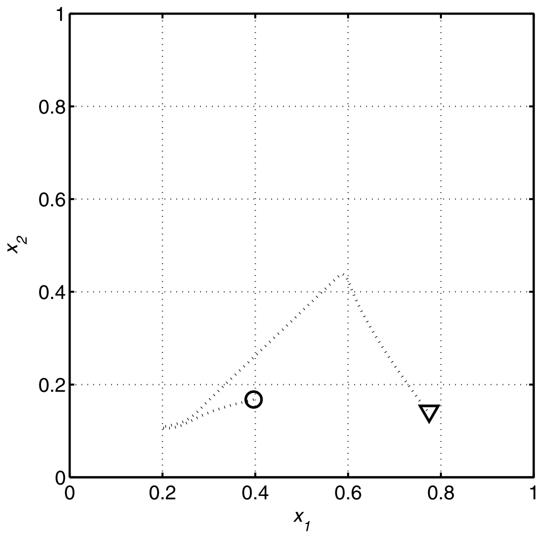


Fig. 7.5 D-optimal trajectory of one mobile sensor for $\alpha = 0.8$

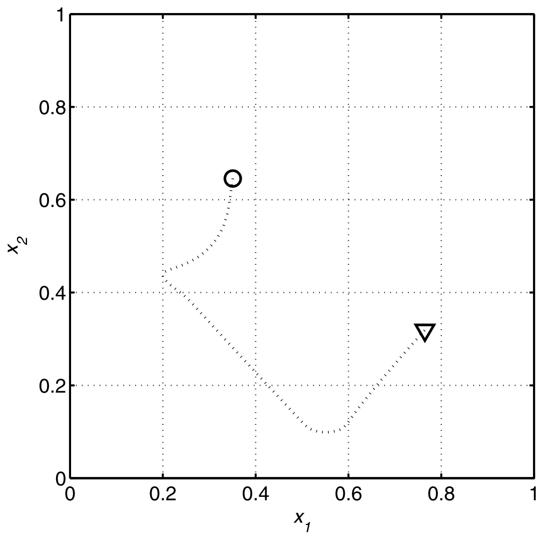


Fig. 7.6 D-optimal trajectory of one sensor mobile for $\alpha = 0.9$.

$$\left|u_i^j(t)\right| \le 0.7, \quad t \in T, \; j = 1, \dots, N, \; i = 1, \dots, 2. \tag{7.111}$$

Our goal is to design their trajectories so as to obtain possibly the best estimates of θ_1, θ_2, and θ_3. In order to avoid getting stuck in a local minimum, computations were repeated several times from different initial solutions. Figure 7.5 presents the resulting trajectories for the best run for one sensor with fractional dynamics of order $\alpha = 0.8$. The trajectory for $\alpha = 0.9$ is given in Fig. 7.6. The trajectories for two sensors are displayed in Fig. 7.7 ($\alpha = 0.8$) and Fig. 7.8 ($\alpha = 0.9$). Finally, the trajectories of a team of three sensors are given in Fig. 7.9 ($\alpha = 0.9$).

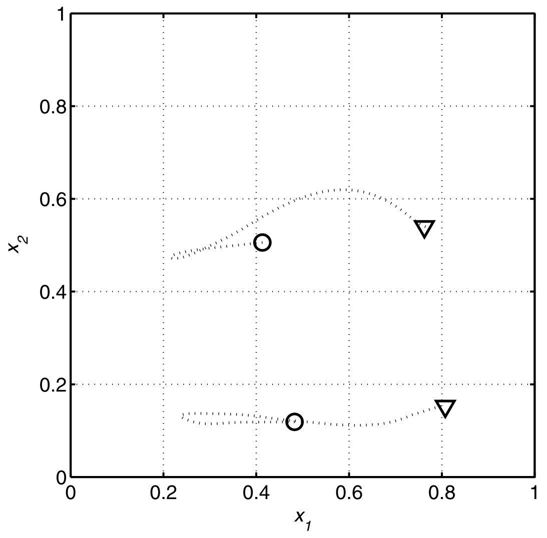


Fig. 7.7 D-optimal trajectories of two mobile sensors for $\alpha = 0.8$

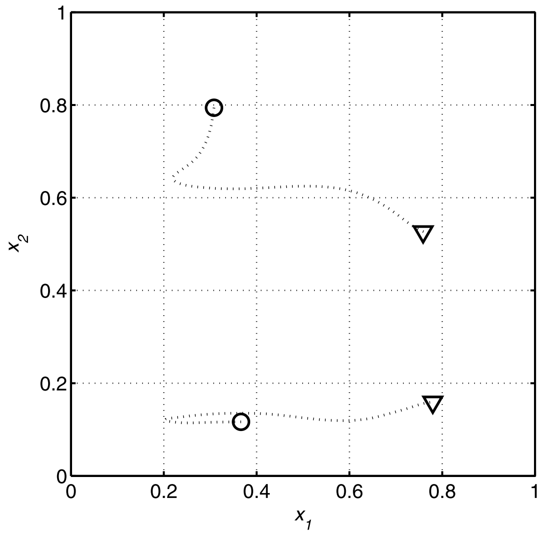


Fig. 7.8 D-optimal trajectories of two mobile sensors for $\alpha = 0.9$

7.9 Chapter Summary

A new formulation toward solving a wide class of fractional optimal control problems has been introduced. The formulation made use of an analytical impulse-response-based approximation to model the fractional dynamics of the system in terms of a state space realization. This approximation created a bridge with a classic optimal control problem, and a readily available optimal control solver was used to solve the fractional optimal control problem. The methodology allowed us to reproduce results from the literature and solve a more complex problem of a fractional

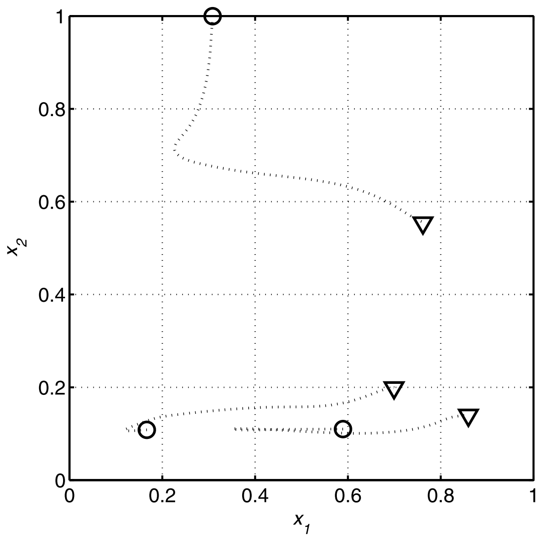


Fig. 7.9 D-optimal trajectories of three mobile sensors for $\alpha = 0.9$

free final time problem. Numerical results show that the methodology, though simple, achieves good results. For all examples, the solution for the integer-order case of the problem is also obtained for comparison purposes.

For the first time, fractional dynamics of the mobile sensors were considered. It is important to note the fact that the introduced formulation has proven to be transcribable into an optimal control problem that can then be solved by readily available optimal control software, in our case, the MATLAB toolbox RIOTS_95. We successfully solved the example of a diffusion system for several teams of sensors and different dynamics. We were also able to use the approximation to obtain the optimal trajectories of a team of sensors with fractional dynamics.

Chapter 8
Optimal Mobile Remote Sensing Policy for Downscaling and Assimilation Problems

8.1 Background on Downscaling and Data Assimilation

In this chapter, our efforts focus on the downscaling problem in the framework of surface soil moisture. Our purpose is to introduce a new methodology to transform low-resolution remote sensing data (for example, from a satellite) about soil moisture to higher-resolution information that contains better information for use in hydrologic studies or water management decision making. Our goal is to obtain a high-resolution data set with the help of a combination of ground measurements and low-altitude remote sensing (typically images obtained from a UAV). In the following, we first describe the methodology developed using only low-resolution information and ground truth. Then, we introduce in two different ways the optimal trajectories of remote sensors, first to solve the problem of maximum coverage knowing the location of ground measurements, and then to solve the problem of optimal data assimilation to optimally improve the assimilation problem using remote sensors.

Because the reader most likely has an electrical engineering background, we give a short introduction of the principles used as a base for the piece of work. These principles come from geoscience and require some definitions and motivation.

8.1.1 Downscaling

The earliest piece of literature that can be linked to downscaling was written by Klein in 1948 [90]. At the beginning, statistical downscaling was used in the field of weather forecasting where global models were not able to provide local information of climate. At that time, downscaling was referred to as *specification*. Later, in the 1980s, similar methodologies were called a "statistical problem of climate inversion" [17, 89]. Another term used is "model output statistics" [167].

The interest and emergence of downscaling are linked to the tools it is based upon, namely, global climate models. Such models appeared only in the 1980s, which explains the young age of this topic.

Readers interested in comprehensive reviews on downscaling should read [22, 159, 166] and references therein.

8.1.1.1 Definition of Downscaling

Even if it is a popular topic in geoscience, as an electrical engineer, one of the first questions one might ask is "What is downscaling?" one basic definition of downscaling is "the process of making the link between the state of some variable representing a large space (henceforth referred to as the 'large scale') and the state of some variable representing a much smaller space (henceforth referred to as the 'small scale')" [22].

Let us take this definition as a starting point and give insight about what is meant by *link*, *large scale*, and *small scale*. As an example, in the downscaling framework for weather modeling, the large-scale variable may, for instance, represent the circulation pattern over a large region, whereas the small scale may be the local temperature as measured at one given point (station measurement).

One of the critical conditions generally assumed for the large-scale variable is the fact that its variations should be slow and smooth in space. The small-scale variable may be a reading from a thermometer or barometer or measurement from a soil moisture probe. It is also important that the large scale and the small scale are physically linked and not just related by a statistical fluctuation or a coincidence. The theory of downscaling requires an implicit and fundamental link between both scales. It is important to distinguish the two concepts of large scale and large volume/area. The two are not necessarily the same, as a large volume may contain many noisy and incoherent small-scale processes. The term small scale could be a little misleading, but what is meant is that the small scale should be local to the domain of interest instead of defined using a small scale. In fact, the local process must be associated with large spatial scales for downscaling to be possible. The main purpose of downscaling is to identify synchronized time behavior on large and small scales. Therefore, practical downscaling focuses on the time dimension.

8.1.1.2 Motivation

The second question we need to ask ourselves is "Why downscaling?" the answer to this question is usually linked to a specific purpose, for example, using global climate models to make an inference about the local climate in a specific area. The global mean value of the temperature is usually not directly relevant for practical use, and more details are required to perform a study.

Global circulation models are a very important tool when studying the earth's climate. However, using them for the study of local climate would provide very

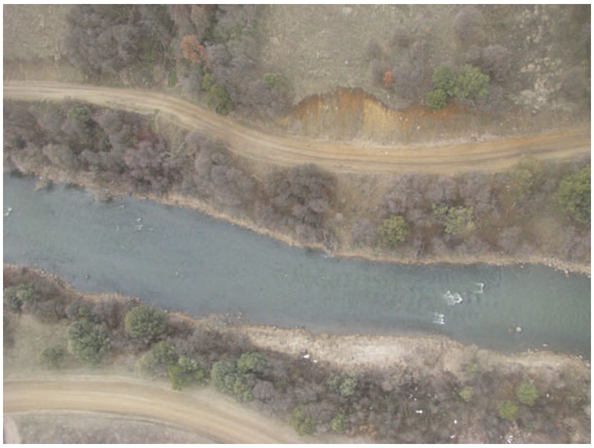


Fig. 8.1 Unmanned aerial image

poor results. It is therefore common to downscale the results from the GCMs either through a local, high-resolution regional climate model (RCM) [42–44] or through empirical/statistical downscaling (ESD) [160]. The GCMs usually do not provide an exact depiction of the real climate system. They frequently involve simple statistical models giving an approximate representation of subgrid processes. It is important to mention one of the limitations of downscaling that the statistical models are based on historical data. It means that there is no guarantee that the past identified statistical relationships between the different data fields will still be true in the future.

8.1.1.3 The Future of Downscaling

It is of importance to note that the use of downscaling may dim in the future. There are two trends in technology that would let us believe so. The first one is the increase in resolution in remote sensors (Fig. 8.1). Nowadays, camera resolution usually doubles every few years. We can imagine that in the near future, even satellite images will be provided at a small-scale resolution, and the frequency of observation can be increased by using unmanned aerial imagery. The second trend is the increase in computing power. Every year, several supercomputers are built that allow more and more detailed GCMs. We provide an illustration (Fig. 8.2) of the increase in detail of GCMs over recent years. During the 1990s, high-resolution GCMs were simulated on the T42 resolution scheme (upper left). For the T42 resolution, the variables (temperature, moisture) were given a single average value over an area of about 200 by 300 km. In 2007, increased computing power allowed scientists to run GCMs at T85 resolution (upper right); variables were averaged over an area of 100 by 150 km. In the future, better resolution will give an enhanced depiction of atmospheric processes and allow for a more realistic topography, increasing the accuracy on regional climate.

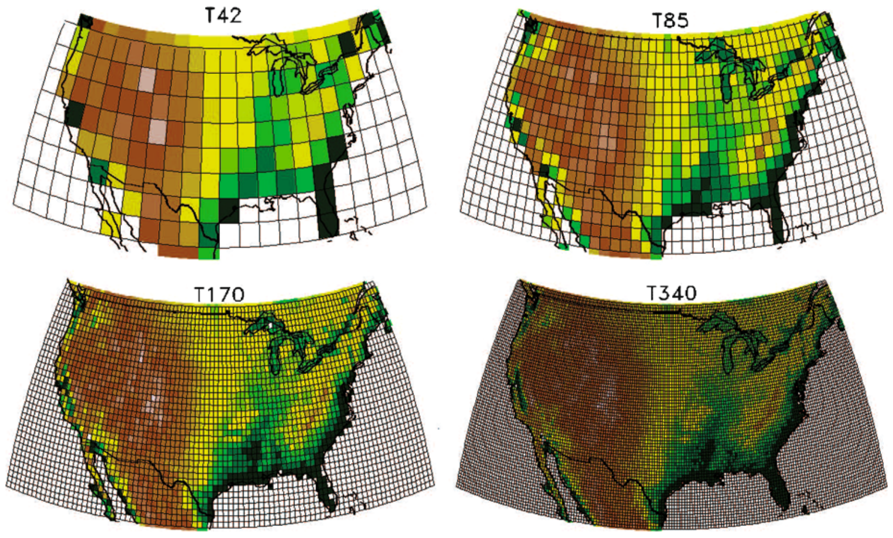


Fig. 8.2 Illustrations of several resolution models for global circulation models (© UCAR, illustration courtesy Warren Washington, NCAR)

8.1.2 Data Assimilation

In geophysics, the process of approximating the true state of a physical system at a given time is called analysis. The information on which the analysis is based includes observational data and a model of the physical system, together with some background information on initial and boundary conditions and possibly additional constraints on the analysis. The analysis is useful in itself as a description of the physical system, but it can also be used, for example, as an initial state for studying the further time evolution of the system.

An analysis can be very simple, for example, a spatial interpolation of observations. However, much better results can be obtained by including the dynamic evolution of the physical system in the analysis. An analysis that combines time-distributed observations and a dynamic model is called assimilation or data assimilation.

Data assimilation methods are designed to combine any type of measurements with estimates from geophysical models. Here are some general reasons to use data assimilation [123]:

1. When comparing the quantity of in situ measurements in the environment and the quantity of satellite remote sensing observations, the latter is much larger. However, their spatial and temporal coverage is still not sufficient for many applications. Data assimilation methods are required to interpolate and extrapolate the remote sensing data.
2. Remote sensing instruments typically observe electromagnetic properties of the Earth system. This implies that most satellite observations are limited to the parts

of the Earth system that can be penetrated by electromagnetic radiation at microwave, infrared, or visible frequencies. Data assimilation systems can spread information from remote sensing observations to all model variables that are in some way connected to the observations.

3. The temporal or spatial resolution of remote sensing data is often too coarse or too fine for a given application. By merging the satellite data with models that resolve the scale of interest, data assimilation methods are capable of aggregating or downscaling the remote sensing data.
4. Some types of remote sensing data are plentiful to the point of overwhelming processing capabilities. Typically, data assimilation systems for numerical weather prediction include sophisticated thinning algorithms for satellite observations, with the consequence that only a small fraction of the available satellite data is actually used in the preparation of a weather forecast. Moreover, there is a great deal of redundancy in satellite observations from different platforms. Data assimilation systems can organize and merge potentially redundant or conflicting satellite data and conventional observations into a single best estimate.
5. In an assimilation system, the physical constraints imposed by models offer additional valuable information. Moreover, models are often forced with boundary conditions that are based on observations. Such boundary conditions may offer indirect and independent observational information about the remotely sensed fields—information that can be captured through data assimilation.

The basic tenet of data assimilation is to combine the complementary information from measurements and models of the Earth system into an optimal estimate of the geophysical fields of interest. In doing so, data assimilation systems interpolate and extrapolate the remote sensing observations and provide complete estimates at the scales required by the application, both in time and spatial dimensions. Data assimilation systems thereby organize the useful and redundant observational information into physically consistent estimates of the variables of relevance to data users. The optimal combination of the measurements with the model information rests on the consideration of the respective uncertainties (or error bars) that come with the observations and the model estimates. Whenever and wherever highly accurate remote sensing data are available, the assimilation estimates will be close to these observations. At times and locations that are not observed by any instrument, the assimilation estimates will draw close to the model solution but will nonetheless be subject to the influence of satellite data in spatial or temporal proximity to the location of interest.

The basic concept of data assimilation is easily understood by considering a scalar model variable m with uncertainty (or error variance) σ_m^2 and a corresponding scalar observation o with uncertainty σ_o^2. The model estimate m represents *prior* or *background* information and may, for example, come from an earlier model forecast that is valid at the time of the newly arrived observation o. The goal is to find the least-squares estimate $\hat{x}$ of the true state x based on the available information. To this end, an objective function J (also known as a cost function, penalty function, or misfit) is defined to quantify the misfit between the true state x and the model estimate and the observation, respectively. In our simple case, the objective function J

is

$$J = \frac{(x-m)^2}{\sigma_m^2} + \frac{(x-o)^2}{\sigma_o^2}. \tag{8.1}$$

Minimization of J with respect to x (by solving $\mathrm{d}J/\mathrm{d}x = 0$) yields

$$\hat{x} = \frac{m\sigma_m^2 + o\sigma_o^2}{\sigma_m^2 + \sigma_o^2}, \tag{8.2}$$

which is typically rewritten as

$$\hat{x} = (1-K)m + Ko, \quad \text{where } K = \frac{\sigma_m^2}{\sigma_m^2 + \sigma_o^2}. \tag{8.3}$$

This best estimate (or analysis) $\hat{x}$ is a weighted sum of the model background m and the observation o. The weights are determined by the relative uncertainties of the model and the observation and are expressed in the (Kalman) gain K (note that $0 \leq K \leq 1$). If the measurement error variance σ_o^2 is small compared to the model uncertainty σ_m^2, the gain will be large, and the resulting estimate will draw closely to the observation, and vice versa. Equal model and measurement error variances $\sigma_o^2 = \sigma_m^2$ produce equal weights ($K = 0.5$), reflecting our equal trust in the model and the observation. Rewriting (8.3) as

$$\hat{x} - m = K(o - m) \tag{8.4}$$

shows that the assimilation increment (difference between the assimilation estimate $\hat{x}$ and the model estimate m) is proportional to the innovation or background departure (difference between the observation o and the model estimate m). The Kalman gain serves as the constant of proportionality. Equation (8.4) is sometimes called the update equation, because the prior model estimate m is updated with information from the observation o. If the errors in the model forecast and the observation are uncorrelated, the error variance of the assimilation estimate is

$$\sigma_{\hat{x}}^2 = (1-K)\sigma_m^2 = K\sigma_o^2 \tag{8.5}$$

and is smaller than the error variances of either the model estimate or the observation alone (recall that $0 \leq K \leq 1$), reflecting the increased knowledge about the true state x after data assimilation.

The assimilation problem can be discussed from many angles, depending on the background and preferences (control theory, estimation theory, probability theory, variational analysis, etc.). A few excellent introductions to data assimilation from different points of view are given by [23–25, 59, 77, 98, 100, 137, 138].

There are numerous data assimilation techniques. We restrict our discussion to advanced data assimilation methods that are based on some measure of model and observation error characteristics.

8.1.2.1 Variational Data Assimilation

In a realistic application, the first right-hand-side term of (8.1) consists of a large sum of model states. The error variance σ_m^2 then becomes the error covariance matrix of these model states. Similarly, the second right-hand-side term of (8.1) becomes a large sum over the individual conventional and satellite observations weighted by the inverse measurement error covariance. Because of the immense size of the vectors and matrices and because of nonlinearities, analytic solutions such as (8.3) are impossible. Instead, variational data assimilation algorithms employ advanced numerical methods to minimize J directly. The two terms of the simple objective function (8.1) are representative of the main ingredients of most current, large-scale atmospheric data assimilation systems. If both terms correspond to a single instant in time, the resulting static data assimilation methods include common techniques such as Optimal Interpolation, Physical-Space Statistical Analysis System (PSAS), 1DVAR, and 3DVAR (where 1D and 3D refer to one and three spatial dimensions, respectively). If the objective function J contains measurements at several different times within an assimilation interval and if the minimum of J is sought for this interval (by varying the model initial condition), the assimilation method is known as 4DVAR (where 4D refers to three spatial dimensions plus the time dimension). In 4DVAR, the error covariance evolution is sometimes referred to as implicit because the assimilation estimates can be obtained without ever explicitly computing their full error covariance matrix. The 4DVAR data assimilation step is thus more flow-dependent than in 3DVAR, and the quality of the estimates improves.

8.1.2.2 The Kalman Filter

Data assimilation algorithms known as Kalman filters share the static update (8.2) with some of the variational techniques, but Kalman filter algorithms also explicitly compute the error covariances through an additional matrix equation (not shown) that propagates error information from one update time to the next, subject to possibly uncertain model dynamics. The error covariance propagation in the traditional Kalman filter and its nonlinear variant, the extended Kalman filter (EKF), however, is prohibitively expensive for large-scale applications. Like variational methods, the Kalman filter can be derived from an objective function, given a number of additional assumptions about the error structure, including model and observation errors that are uncorrelated in time and mutually uncorrelated. The EKF has been demonstrated successfully for soil moisture data assimilation [122, 127]. Reduced-rank approximations such as the ensemble Kalman filter (EnKF) [58, 74, 140] are designed to reduce the number of degrees of freedom to a manageable level. The idea behind the EnKF—a Monte Carlo variant of the Kalman filter—is that a comparably small ensemble of model trajectories captures the relevant parts of the error structure. The EnKF is flexible in its treatment of errors in model dynamics and parameters. It is also very suitable for modestly nonlinear problems.

8.2 Downscaling and Assimilation Problems for Surface Soil Moisture

The work described in this section is based on [86]. We would like to orient the reader looking for more details to this article.

8.2.1 Introduction

In hydrology, when trying to simulate a system using PDEs or trying to identify the dynamics, it is important that the data used as initial conditions or used to know the state of the system have a similar scale to the model. Because most of the time it is not the case, we need to be able to modify the scale of a given measurement and fit it to match our model scale. Among the potential candidates of scale modification techniques, the one we discuss here uses data assimilation such that the best use of the collected information at different scales can be achieved. Such a problem is also called *scale reconciliation* and is defined as the process of data assimilation done to merge data at different scales.

Most of the time, collecting information is not an issue as data can be gathered in many different ways—from satellites, UAV imagery, or ground measurements. However, combining the data from all those different sources in the most efficient way for a given application becomes a research problem. Indeed, each different platform provides data at different temporal and spatial resolutions, most of the time not matching the model scale. Therefore, the problem is in extracting the information of interest within all this heterogeneous sensor data, in both temporal and spatial scales.

In the work under consideration, the purpose of the introduced methodology is to combine information from a coarse-resolution image and point measurements. More precisely, we are interested only in the merging of the spatial scale. Under such consideration, we need to assume that the continuity of soil moisture and the correlation distance of soil moisture have to be larger than the spacing between ground measurements.

The sources for soil moisture measurement are generally twofold. First, the traditional soil moisture measurement methods provide pointwise data and are based on gravimetric, nuclear and electromagnetic, and tensiometric and hygrometric methods. On the other hand, remote sensing measurements in the microwave region can give useful information about soil moisture due to the strong contrast between the dielectric constant of dry soil and water and its effect on microwave emission. They provide coarse-resolution images of the distribution of surface soil moisture.

8.2.2 Kaheil and McKee's Algorithm

We consider the downscaling of an image G_0 at resolution 0 down to a fine image G_n at resolution n. The structure considered for downscaling is described in

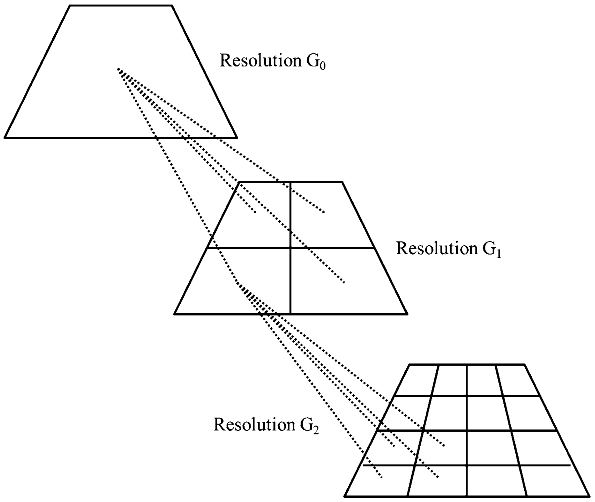


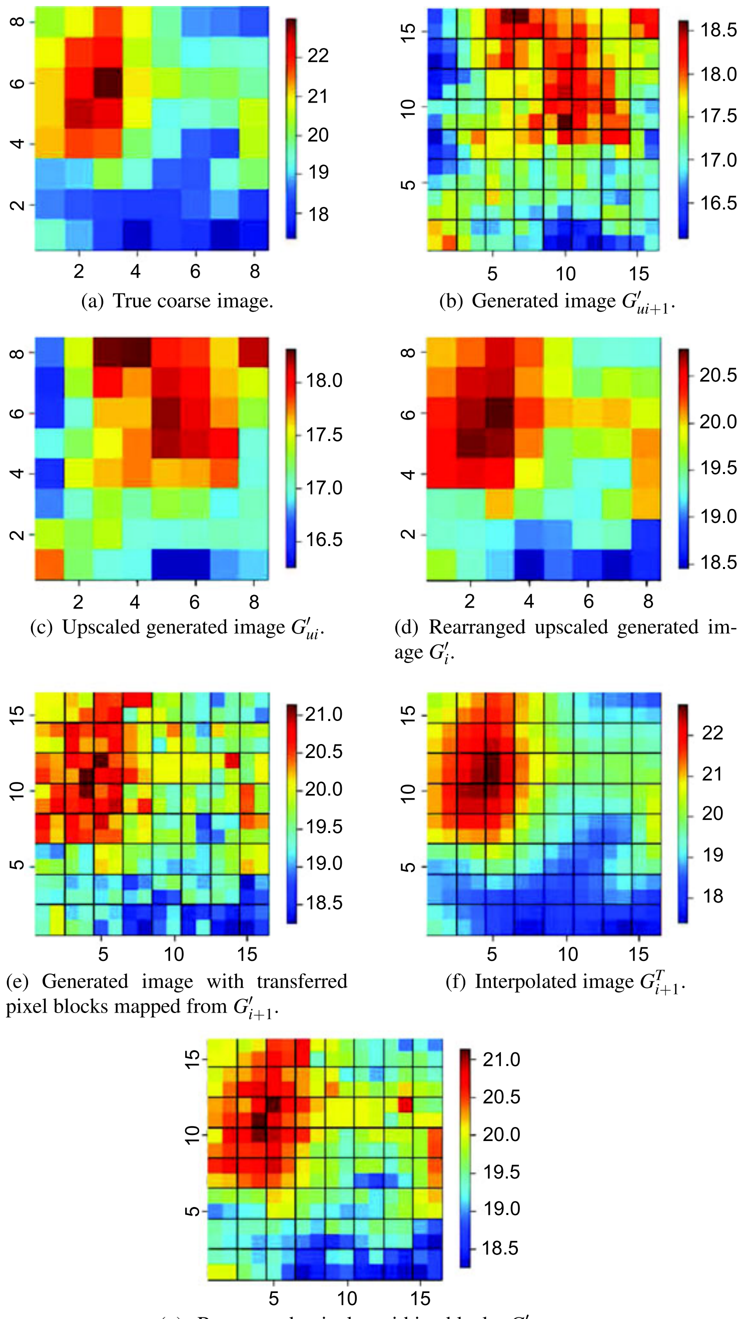
Fig. 8.3 Description of the downscaling structure

Fig. 8.3. As can be observed, for the lowest resolution 0, a given pixel possesses only one value, and at the next resolution, there are four values. At the final nth resolution, the original area containing the pixel will contain 4^n values.

The purpose of the introduced algorithm is to create an image at a high resolution satisfying two conditions. The first condition to be met is that the generated image should be close to the original image when upscaled back to the lower resolution. The second condition requires the final image to incorporate the point measurement information. The proposed approach can be portrayed as the repetition (n times) of two steps (initialization and spatial pattern search) and a final assimilation using pointwise ground measurements. The initialization and the spatial pattern search are here to generate an image at resolution n based solely on the underlying dynamics of the system and the original satellite coarse image. The assimilation step is here to combine the new generated image and the ground measurements to produce the final image. To illustrate each step, we provide in Fig. 8.4a graphical illustration of the inner workings of the method.

8.2.2.1 Initialization

The initialization step is composed of three tasks. The first task is a maximum-likelihood parameter estimation (MLE) analysis executed on the low-resolution image G_0, assumed to be the true image at resolution 0. The end result of the MLE is a list of parameters linked with the variogram of the coarse image G_0. The parameters estimated are then used to create an image at the next finer resolution called G'_{ui+1}. In G'_{ui+1}, i stands for the current iteration number, u refers to the "unsorted" nature of the image, and the apostrophe stands for an unassimilated image. The unsorted nature of the generated image comes from a given variogram and not the image itself. Therefore, the spatial properties of the two images are the same, but the spatial patterns are different. The second part of the initialization consists of upscaling



(a) True coarse image.

(b) Generated image G'_{ui+1}.

(c) Upscaled generated image G'_{ui}.

(d) Rearranged upscaled generated image G'_{i}.

(e) Generated image with transferred pixel blocks mapped from G'_{i+1}.

(f) Interpolated image G^{T}_{i+1}.

(g) Rearranged pixels within blocks G'_{i+1}.

Fig. 8.4 Hierarchical algorithm Step 1 [86]

image G'_{ui+1} back to the former resolution (resolution i) to obtain an image G'_{ui}. Because of the unsorted nature of the image generated by the MLE, its upscaled version will be different from the original image in both value and spatial pattern. The third task of the initialization consists of rearranging the pixels within image G'_{ui} so that the order of these pixels (minimum to maximum) will follow the same order as those of the true image. The resulting image is called G'_i. In this task, only the pixels at a coarse resolution are rearranged, which means that the arrangement of pixels within a coarse pixel is untouched. The resulting image is called G'_{i+1}. The spatial pattern issue being solved, the value deviation issue is addressed using a ratio bias remover to correct for the values. The values in each of image G'_{i+1} are multiplied by the ratio $R = \sum(G_i)/\sum(G'_i)$ to correct for the value bias. After this third task, the resulting image G'_{i+1} is similar to its coarse image when upscaled. However, there is still a discontinuity coming from the transfer of pixel blocks; each pixel within the coarse pixels have not been rearranged. These discontinuities are polished in the next step using a spatial pattern search technique.

8.2.2.2 Spatial Pattern Search

The image resulting from the initialization step is at a finer resolution and is very close to the coarse image when upscaled. However, there are still some discontinuities between two coarse neighboring pixels. To address this problem, the pixels within the "initialized" image are rearranged and sorted. The reference for this rearrangement of pixels is an interpolated version G_i at resolution $i+1$ called G^T_{i+1}. There are several interpolation techniques that can be used to generate G^T_{i+1} such as a linear interpolation or cubic splines. Let us consider the four pixels from G'_{i+1} belonging to the same original pixel in G_i. These four pixels are rearranged according to the distribution of pixels within the reference interpolated image. This task still guarantees that the resulting image provides little error when upscaled to the true coarser image but improves the smoothness between two neighboring pixel blocks. The resulting image is still called G'_{i+1}. The initialization and the spatial pattern search are repeated, increasing the value of the subscript with each iteration until the final resolution n is reached. The final image G'_n is generated after n iterations. However, G'_n still does not account for the ground measurements. Therefore, another bias remover is required and is described next.

8.2.2.3 Assimilation

The third step, called *assimilation*, consists of fine-tuning the final image based on the original image and point measurements. The method introduced here makes use of support vector machines (SVMs). An SVM is a machine learning paradigm based on statistical learning theory. The theory and algorithms for SVMs can be found in [47] and references therein. The main idea behind using SVMs for the assimilation step is to approximate a one-to-one function between the approximated coarse image and the true coarse image. The resulting function at the coarse resolution is applied at the fine resolution to obtain a new approximation of the fine image. Therefore, the relation at the coarsest resolution between the observed image and model-generated image is learned through the application of the SVM algorithm. The advantage of using SVMs for our application is the fact that the point measurement data can be added to the training set. The training data set of the SVM consists of random pixels of image G'_n at resolution 0 as input, corresponding pixels of G_0 as output, and readings of fine pixels from G'_n at point measurement locations versus corresponding point measurement values P_z. Once the SVM has finished the training process, it is applied to G'_n to get the final fine-scaled image G_n.

8.3 Introduction of UAV-Based Remote Sensors

The framework introduced in Sect. 8.2 considers only a single coarse image (from either a satellite or an aerial vehicle) and ground measurements. However, it would be possible to extend the framework for multiscale downscaling and assimilation where the observations could come from both satellite, UAV(s), and ground measurements. The introduction of UAVs into the framework can be challenging as it was developed from static measurements. UAVs are by nature mobile platforms, which means that their locations can be variable. Based on this observation, we can see that there is an infinite number of possible trajectories for these UAVs, and we need to decide on one based on some criterion. This criterion could be used in an optimization to generate the trajectory of the UAVs.

There are several potential candidates for the optimality criterion of the UAV trajectories. For example, we could optimize the trajectory of the UAVs so that they maximize the coverage of the area defined by G_0 but avoid the location of ground sensors to reduce redundancy.

When comparing the original image G_0 and the final image upscaled back to its initial coarse resolution $G_{n\to 0}$, an error may still exist at certain locations. We call such an image $G_{e0} = |G_0 - G_{n\to 0}|$ (Fig. 8.5). The image G_{e0} can be seen as the residual error from the downscaling and assimilation procedure. In the following, we develop a methodology that optimizes the trajectory of UAVs so as to maximize the coverage of G_{e0}, providing the best information for another downscaling and assimilation procedure, enhanced with remote aerial measurements.

8.4 Optimal Trajectories for Data Assimilation

8.4.1 Description of the Problem

The purpose of the optimal measurement problem is to determine the steering of the UAV which minimizes a design criterion $J(\cdot)$ defined by the area covered by the

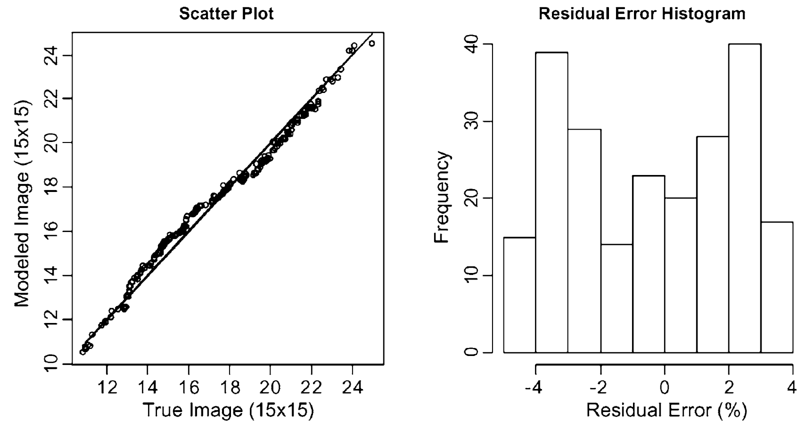


Fig. 8.5 Illustration of the downscaling residual error

UAV and G_{e0}. The value of the design criterion is determined by the trajectories resulting from that steering, subject to constraints on the magnitude of the controls and induced state constraints.

The mobile remote sensors are assumed to ambulate in a spatial domain $\Omega_{\text{sens}} \in \mathbb{R}^3$. The sensors are able to remotely take measurements in $\Omega_{\text{meas}} \in \mathbb{R}^2$ over a given observation horizon $T = [t_0,\ t_f]$. We call $\boldsymbol{x}^j = [x_1^j(t), x_2^j(t), x_3^j(t)]^T$: $T \to \Omega_{\text{sens}}$ the trajectory of the jth remote sensor. We call $z^j : T \to \Omega$ the collection of measurements in Ω_{meas} where the jth sensor is observing. We assume that a function $\boldsymbol{f} : \Omega_{\text{sens}} \to \Omega_{\text{meas}}$ linking the position of the sensor and measurements exists. The observations for the jth sensor are assumed to be of the form

$$z^j(t) = \boldsymbol{y}\big(\boldsymbol{f}\big(\boldsymbol{x}^j(t)\big), t\big) + \boldsymbol{\varepsilon}\big(\boldsymbol{f}\big(\boldsymbol{x}^j(t)\big), t\big), \tag{8.6}$$

where ε stands for the measurement noise. We assume that the UAV's dynamics can be described by the following differential equation:

$$\dot{\boldsymbol{x}}^j(t) = \boldsymbol{g}\big(\boldsymbol{x}^j(t), \boldsymbol{u}^j(t)\big) \quad \text{a.e. on } T, \quad \boldsymbol{x}^j(0) = \boldsymbol{x}_0^j, \tag{8.7}$$

where the vector $\boldsymbol{x}_0^j \in \mathbb{R}^3$ represents the initial location of the jth sensor, and $\boldsymbol{u}$: $T \to \mathbb{R}^{r_s}$ is a measurable control function satisfying the inequality

$$\boldsymbol{u}_l \le \boldsymbol{u}(t) \le \boldsymbol{u}_u \quad \text{a.e. on } T, \tag{8.8}$$

for some known constant vectors $\boldsymbol{u}_l$ and $\boldsymbol{u}_u$. We define the remote sensing function as follows:

$$\boldsymbol{y}_i(\boldsymbol{x}, \boldsymbol{u}, t) = G_{e0}\big(\boldsymbol{f}\big(\boldsymbol{x}^i(t)\big)\big), \tag{8.9}$$

where $\boldsymbol{f}$ is the geographical remote sensing function, giving the location of the measurements on the ground. We define the weighting function linked with the altitude of a sensor as follows:

$$G(\boldsymbol{x}) = \begin{cases} 0 & \text{if } x_3 > z_0, \\ i/n & \text{if } z_{i+1} > x_3 > z_i, \\ 1 & \text{if } z_n > x_3. \end{cases} \tag{8.10}$$

Let us introduce

$$\boldsymbol{s}(t) = \left(\boldsymbol{x}^1(t), \boldsymbol{x}^2(t), \ldots, \boldsymbol{x}^N(t)\right)^T, \tag{8.11}$$

where $\boldsymbol{x}^j : T \to \Omega_{\text{sens}}$ is the trajectory of the jth sensor. We define the set $\boldsymbol{s}_0$ of initial locations as

$$\boldsymbol{s}_0 = \left(\boldsymbol{x}^1(0), \boldsymbol{x}^2(0), \ldots, \boldsymbol{x}^N(0)\right)^T. \tag{8.12}$$

8.4.2 Problem Formulation

Given the above formulation, we can cast the optimal measurement policy problem as the following optimization problem: Find the pair $(\boldsymbol{s}_0, \boldsymbol{u})$ that minimizes

$$J(\boldsymbol{s}_0, \boldsymbol{u}) = \sum_{i=1}^{N} \left(\int_\Omega \left(G_{e0}(\boldsymbol{x})\right) \mathrm{d}x - \int_\Omega \left(\int_{t_0}^{t_f} y_i\left(\boldsymbol{x}^i, \boldsymbol{u}, t\right) G\left(\boldsymbol{x}^i\right) \mathrm{d}t \right) \mathrm{d}x \right)^2 \tag{8.13}$$

over the set of feasible pairs

$$\begin{aligned} \mathcal{P} = \big\{(\boldsymbol{s}_0, \boldsymbol{u}) | \boldsymbol{u} : T \to \mathbb{R}^r \quad & \text{is measurable}, \\ & \boldsymbol{u}_l \le \boldsymbol{u}(t) \le \boldsymbol{u}_u \text{ a.e. on } T, \boldsymbol{s}_0 \in \Omega_{\text{sens}}\big\}, \end{aligned} \tag{8.14}$$

subject to constraints on the control input.

8.4.3 Numerical Method to Find the Solution

This problem can hardly be solved using analytical methods. It is therefore necessary to use a numerical method to solve the problem. We use the MATLAB toolbox called RIOTS_95 [126]. It is a powerful tool for solving a large class of optimal control problems. The considered problem can be described with M-files. The theory behind the toolbox can be found in [115] and uses the approach of consistent approximations.

The performance criterion $J(\cdot)$ is calculated using the following steps. First, the left-hand side of the criterion $\int_\Omega G_{e0}(\boldsymbol{x})\,\mathrm{d}x$ is calculated based on the given G_{e0}.

Then, based on the trajectories of the UAVs and the defined remote sensing function, the resulting measurement footprint is evaluated. The convex hull of the measurement footprint is then calculated in order to discard the redundancies of measurements on the ground. Each point of the convex hull is then assigned a weight based on the distance of the UAV from the ground. Then, the convex hull is transformed into a Delaunay triangulation, and the integral of $G_{e0}(\boldsymbol{x})$ of each triangle is computed. The sum of the integrals of all triangles is then added to compute the right-hand side of the criterion $\int_\Omega (\int_{t_0}^{t_f} y_i(\boldsymbol{x}^i, \boldsymbol{u}, t) G(\boldsymbol{x}^i)\,\mathrm{d}t)\,\mathrm{d}x$.

8.5 An Illustrative Example

8.5.1 System's Description

We use a demonstrative example to illustrate the method developed earlier. We consider the mapping G_{e0} of the residual error of a downscaling problem as

$$G_{e0}(\boldsymbol{x}) = \theta_1 + \theta_2 x_1 + \theta_3 x_2 + \varepsilon(\boldsymbol{x}) \tag{8.15}$$

for $\boldsymbol{x} = [x_1\ x_2]^T \in \Omega_{sys} = (0,1)^2$ and $t \in [0,1]$. $\varepsilon(\boldsymbol{x})$ refers to a random field of amplitude σ^2. The dynamics of the mobile sensors follow the given dynamical model

$$\dot{\boldsymbol{x}}^j(t) = \boldsymbol{u}^j(t), \qquad \boldsymbol{x}^j(0) = \boldsymbol{x}_0^j, \tag{8.16}$$

for $\boldsymbol{x} = [x_1\ x_2\ x_3]^T \in \Omega_{\text{sens}} = (0,1)^3$ and additional constraints

$$\left|u_i^j(t)\right| \le 0.7, \quad t \in T,\ j = 1, \ldots, 2,\ i = 1, 2, \tag{8.17}$$

$$\left|u_i^j(t)\right| \le 0.2, \quad t \in T,\ j = 1, \ldots, N,\ i = 3. \tag{8.18}$$

We consider three different sets of values for θ_1, θ_2, θ_3, and σ.

8.5.2 Results

8.5.2.1 $\theta_1 = 1, \theta_2 = 0.1, \theta_3 = 0.2, \sigma = 0$

These parameter values are considered to test the methodology when G_{e0} is linear in space. This allows us to test the numerical method under smooth conditions and make sure that the implementation allows convergence of the optimization. The resulting trajectory for one UAV is given in Fig. 8.6, where both the initial location of the UAV and the trajectory are optimized. The optimal trajectories of two UAVs are given in Fig. 8.7, the initial locations are set as $\boldsymbol{x}_0^1 = [0.9, 0.9, 0.4]^T$ and $\boldsymbol{x}_0^2 = [0.9, 0.8, 0.4]^T$, and only the trajectory is optimized. We can observe that the covering of G_{e0} is mostly located in the higher values. This can be expected as we are trying to maximize the coverage of G_{e0}.

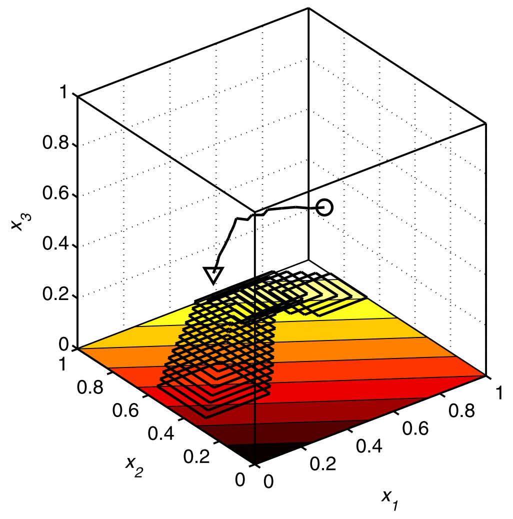


Fig. 8.6 Optimal trajectory of one sensor for $\theta_1 = 1$, $\theta_2 = 0.1$, $\theta_3 = 0.2$, and $\sigma = 0$

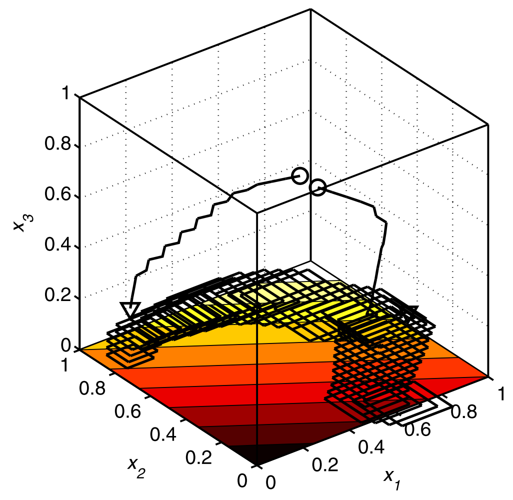


Fig. 8.7 Optimal trajectory of two sensors for $\theta_1 = 1$, $\theta_2 = 0.1$, $\theta_3 = 0.2$, and $\sigma = 0$

8.5.2.2 $\theta_1 = 0, \theta_2 = 0, \theta_3 = 0, \sigma = 1$

We consider this example to see how the methodology would perform under a realistic scenario. G_{e0} is a pseudo-random field that would be likely to happen from the outcome of a downscaling and assimilation procedure as described in Sect. 8.2. The optimization can hardly converge because of the randomness of the field. We provide the resulting trajectories for the best attempt in Fig. 8.8.

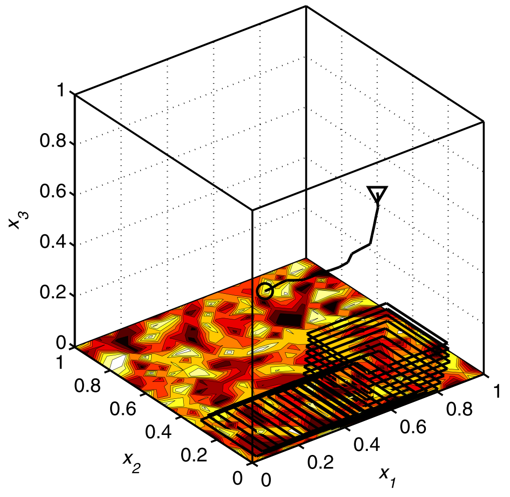


Fig. 8.8 Optimal trajectory of one sensor for $\theta_1 = 0$, $\theta_2 = 0$, $\theta_3 = 0$, and $\sigma = 1$

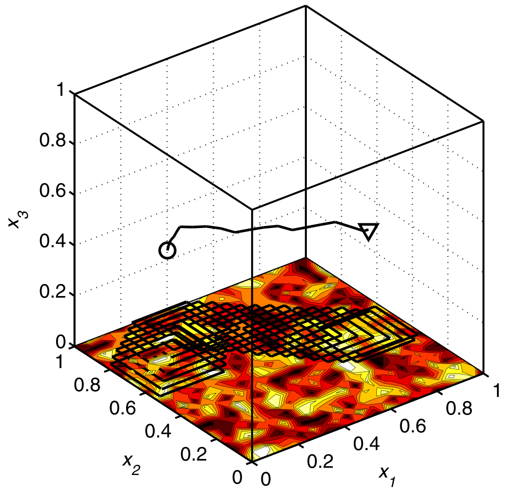


Fig. 8.9 Optimal trajectory of 1 sensor for $\theta_1 = 1$, $\theta_2 = 0$, $\theta_3 = 0$, and $\sigma = 0.1$

8.5.2.3 $\theta_1 = 1, \theta_2 = 0, \theta_3 = 0, \sigma = 0.1$

Because of the poor results obtained when the field is random, we introduce an offset to encourage the optimization to increase the coverage. The result is given in Fig. 8.9. In most cases, the optimization is able to converge. Several attempts are necessary to obtain a good trajectory.

8.6 Chapter Summary

In this chapter, we were able to address the problem of downscaling soil moisture data. Based on an existing methodology to downscale, we introduced the problem of optimal remote sensor trajectory so as to maximize the coverage of the areas where the downscaling was inaccurate. The problem was formulated as an optimal control one, which allowed us to use optimal control solvers. A numerical method to solve the problem was introduced and successfully applied to a numerical example.

Chapter 9
Conclusions and Future Work

9.1 Conclusions

CPSs constitute one of the next big challenges of the engineering community. They will require advances from a lot of different domains of engineering in order to be successful.

Among the challenges of CPSs, system identification has to be one of the first to be addressed, especially when the system under consideration is a DPS. Because of the complexity of DPSs, the identification procedure will itself be a CPS with actuators to ensure a good enough excitation and sensors to gather measurements about the dynamics of the system. However, the distributed nature of the DPS makes the location of such actuators and sensors a question to be addressed. The solution will depend not only on the dynamics of the system but also on the nature of those actuators and sensors. The nature of sensors and actuators can be their dynamics, their geometrical support (pointwise, zonal, whole domain, boundary), their communication topology, their autonomy, and their precision. Most of this monograph focuses on providing methodologies to obtain the optimal sensing and actuation policies in a CPS of a distributed nature.

In this monograph, we provide the following list of contributions to the state of the art:

- We propose an approach to optimize both the trajectories of mobile sensors and their measurement accuracy for parameter estimation of a distributed parameter system. Using sensors with different accuracies can lead to better parameter estimates than homogeneous sensors. This approach has the advantage of providing the maximum number of sensors necessary (a sensor with an accuracy of 0 can be discarded).
- We consider the case of remote sensing where the sensor is not located inside the considered DPS but in a different domain. We introduce a method to obtain the optimal trajectories of those mobile robots remotely monitoring a distributed parameter system with respect to parameter estimation.
- We provide a numerical solution for generating and refining a mobile sensor motion trajectory for the estimation of the parameters of a DPS in the "closed-loop"

sense. The basic idea is to use the finite-horizon control type of scheme. First, the optimal trajectories are computed in a finite time horizon based on the assumed parameter values. For the following time horizon, the parameters of the distributed parameter system are estimated using the measured data in the previous time horizon, and the optimal trajectories are updated accordingly based on these estimated parameters obtained.

- Under such a closed-loop scheme, we discuss the influence of the communication topology between the mobile sensors on the estimation of the parameters of a distributed parameter system. Of course, more communication leads to faster estimates, but acceptable results can be obtained with limited communication.
- We introduce the problem of determining the optimal sensors' trajectories so as to estimate a set of unknown parameters for a system of a distributed nature where the bounds on the parameters' values are known. This leads to average trajectories that can be fairly close to those obtained with the real parameter values.
- Besides the explicit design variables that are the sensor trajectories, there exists an implicit one that is the excitation of the system, that is to say, the actuation. Given a sensor configuration (static and/or mobile), we propose a numerical procedure to optimize the trajectory of mobile actuators to find parameter estimates of a distributed parameter system.
- Based on the newly introduced optimal actuation policy, we develop a framework to solve the problem of determining optimal sensors' and actuators' trajectories so as to estimate a set of unknown parameters in what constitutes a CPS.
- We discuss fractional-order optimal control problems (FOCPs) and their solution by means of rational approximation. The original problem is then reformulated to fit the definition used in general-purpose optimal control problem (OCP) solvers.
- A different direction to approximately solving FOCPs is introduced. The method uses a rational approximation of the fractional derivative operator obtained from the singular value decomposition of the Hankel data matrix of the impulse response and can potentially solve any type of FOCPs.
- We propose a methodology to optimize the trajectory of mobile sensors whose dynamics contain fractional derivatives to find parameter estimates of a distributed parameter system.
- We introduce a methodology to obtain the optimal trajectories of a group of mobile remote sensors for scale reconciliation for surface soil moisture.

9.2 Future Research Directions

Even though the framework has been greatly extended by the work described in this monograph, there are still plenty of research opportunities.

9.2.1 *Communication Topology Influence on Regional Controllability and Observability for DPS*

The framework of regional controllability and observability of DPSs was introduced a long time ago, before applications even existed. Since then, little progress has been achieved to bring the framework further. The reason is that sensors and actuators in DPSs were first introduced as mathematical concepts rather than based on real applications. Therefore, concepts such as communication topologies have never been considered. Nowadays, communication topology is a highly competitive research direction because of its direct impact on mobile robots' algorithm. A natural evolution of the framework is to introduce communication topology in regional controllability and observation.

9.2.2 *Directed Communication Topologies*

A good way to improve the estimation would be the use of directed communication topology where at least one sensor would be able to receive information from all other sensors and therefore have great information regarding the system. The use of such topologies will be part of our future research efforts.

9.2.3 *Regional Identifiability of a DPS*

Identifiability is a term mostly used in statistics. The concept of identifiability has proved useful when attempting to answer questions like "Is it theoretically possible to learn the true value of this model's underlying parameter after obtaining an infinite number of observations from it?" The problem of identifiability of a DPS has been studied in the past. However, the problem of regional identifiability has not yet been investigated.

Appendix A
Notation

A.1 General Notation

$\mathbb{R}$	Set of real numbers
$\mathbb{N}$	Set of natural numbers
$\mathbb{Z}$	Set of integer numbers
y	State variable
$\boldsymbol{u}$	Control variable
(z)	Measurements
t	Time
t_f	Final time of the experiment
T	$(0, t_f)$
Ω	Space domain, an open bounded regular subset of $\mathbb{R}^n$
Γ or $\partial\Omega$	Boundary of Ω
$\overline{\Omega}$	$\Omega \cup \partial\Omega$
Q	$\Omega \times]0, T[$
Σ	$\partial\Omega \times]0, T[$
$\mathcal{L}(X, Y)$	Space of linear maps from X to Y
$\mathcal{L}(X)$	$\mathcal{L}(X, X)$
$L^p(0, T; X)$	Integrable functions $f :]0, T[\mapsto X$ such that $t \mapsto \lVert f(t)\rVert^p$ is integrable on $]0, T[$
$L^2(\Omega)$	Space of square-integrable functions on Ω
$D(H)$	Domain of the operator H
$\boldsymbol{\theta}$	Parameter vector
$\hat{\boldsymbol{\theta}}$	Estimate of the parameter vector
$\det(A)$	Determinant of the matrix A
$\operatorname{trace}(A)$	Trace of the matrix A
$\lambda_{\max}(A)$	Largest eigenvalue of the matrix A

A.2 Special Notation in Chap. 2

A, B, C	Dynamics, control, and observation operators
Z	Observation space (Hilbert space)
U	Control space (Hilbert space)
Y	State space (Hilbert space)
$(\Phi(t))_{t\geq 0}$	Semigroup generated by A
ω	Subregion of Ω
$\mathrm{Im}(H)$	Image of H
$\mathrm{Ker}(H)$	Kernel of the operator H
$H^\star$	Adjoint operator of H
$P_A x$	Projection of x on A
$\langle \cdot, \cdot \rangle_H$	Inner product in H
p_ω or χ_ω	Restriction to the region ω
$p_\omega^\star$ or i_ω	Adjoint of p_ω
$\bar{A}$	Closure of A
$\mathrm{supp}(g)$	Support of a function g

A.3 Special Notation in Chap. 3

p_i	Measurement precision weight of the ith sensor
ξ_N	Design of an experiment
$\xi^\star$	Optimal design
$\mathfrak{M}$	Set of admissible information matrices

A.4 Special Notation in Chap. 4

Ω_{sys}	Space domain where the system is defined
Ω_{sens}	Space domain where sensors can ambulate
Ω_{meas}	Space domain where sensors can take measurements
res	Resolution of the remote sensors

A.5 Special Notation in Chap. 6

$\mathcal{G}_i(\boldsymbol{x}, \boldsymbol{x}_a^i, t)$	Actuation function for the kth actuator
$\boldsymbol{x}_a^k$	Trajectory of the kth actuator

A.6 Special Notation in Chap. 7

α	Order of derivation
${}_aD_t^\alpha f(.)$	Left Riemann–Liouville fractional derivative of a function $f(\cdot)$
${}_tD_b^\alpha f(\cdot)$	Right Riemann–Liouville fractional derivative of a function $f(\cdot)$
${}_a^CD_t^\alpha f(.)$	Left Caputo fractional derivative of a function $f(\cdot)$
${}_t^CD_b^\alpha f(.)$	Right Caputo fractional derivative of a function $f(\cdot)$
A	State matrix
B	Input matrix
C	Output matrix
D	Feedthrough matrix
Γ	Gamma function $\Gamma(x) = \int_\infty^0 t^{x-1}e^{-t}\,\mathrm{d}t$

Appendix B
RIOTS Tutorial

B.1 Introduction

RIOTS_95 is designed as a MATLAB toolbox written mostly in C, Fortran, and M-file scripts. It provides an interactive environment for solving a very broad class of optimal control problems (OPCs). RIOTS_95 comes precompiled for use with the Windows 95/98/2000 or Windows NT operating systems. The user-OCPs can be prepared purely in M-files, and no compiler is needed to solve the OCPs. To speed up the OCP solving process, there are two ways to go: by using the MATLAB Compiler or by providing the user-OCP in C, which is to be compiled by a C-compiler and then linked with some prebuilt linking libraries. This chapter describes the use and operation of RIOTS_95 together with two demonstrative examples in solving optimal control problems, one of which is an application in chemical engineering.

The numerical methods used by RIOTS_95 are supported by the theory in the PhD dissertations of Dr. Adam L. Schwartz [125], who uses the approach of consistent approximations as defined by Polak [114]. In this approach, a solution is obtained as an accumulation point of the solutions to a sequence of discrete-time optimal control problems that are, in a specific sense, consistent approximations to the original continuous-time, optimal control problem. The discrete-time optimal control problems are constructed by discretizing the system dynamics with one of four fixed step-size Runge–Kutta integration methods and by representing the controls as finite-dimensional B-splines. Note that RIOTS_95 also includes a variable step-size integration routine and a discrete-time solver. The integration proceeds on a (possibly nonuniform) mesh that specifies the spline breakpoints. The solution obtained for one such discretized problem can be used to select a new integration mesh upon which the optimal control problem can be rediscretized to produce a new discrete-time problem that more accurately approximates the original problem. In practice, only a few such rediscretizations need to be performed to achieve an acceptable solution.

RIOTS_95 provides three different programs that perform the discretization and solve the finite-dimensional discrete-time problem. The appropriate choice of optimization program depends on the type of problem being solved and on the number of points in the integration mesh. In addition to these optimization programs, RIOTS_95 also includes other utility programs that are used to refine the discretization mesh, to compute estimates of integration errors, to compute estimates for the error between the numerically obtained solution and the optimal control, and to deal with oscillations that arise in the numerical solution of singular optimal control problems.

B.2 Features of RIOTS_95

RIOTS_95 is a collection of programs that are callable from the mathematical simulation program MATLAB for Windows. Most of these programs are written in either C, Fortran (and linked into MATLAB using its MEX/DLL facility), or MATLAB M-script language. All of MATLAB functionality, including command line execution and data entry and data plotting, are available to the user. The following is a list of some of the main features of RIOTS_95:

- Solves a very large class of finite-time optimal control problems that includes trajectory and endpoint constraints, control bounds, variable initial conditions (free final time problems), and problems with integral and/or endpoint cost functions.
- System functions can be supplied by the user as either object code or M-files.
- System dynamics can be integrated with fixed step-size Runge–Kutta integration, a discrete-time solver, or a variable step-size method. The software automatically computes gradients for all functions with respect to the controls and any free initial conditions. These gradients are computed exactly for the fixed step-size routines.
- The controls are represented as splines. This allows for a high degree of function approximation accuracy without requiring a large number of control parameters.
- The optimization routines use a coordinate transformation that creates an orthonormal basis for the spline subspace of controls. The use of an orthogonal basis can result in a significant reduction in the number of iterations required to solve a problem and an increase in the solution accuracy. It also makes the termination tests independent of the discretization level.
- There are three main optimization routines, each suited for different levels of generality of the optimal control problem. The most general is based on sequential quadratic programming methods. The most restrictive, but most efficient for large discretization levels, is based on the projected descent method. A third algorithm uses the projected descent method in conjunction with an augmented Lagrangian formulation.
- There are programs that provide estimates of the integration error for the fixed step-size Runge–Kutta methods and estimates of the error of the numerically obtained optimal control.
- The main optimization routine includes a special feature for dealing with singular optimal control problems.
- The algorithms are all founded on rigorous convergence theory.

In addition to being able to accurately and efficiently solve a broad class of optimal control problems, RIOTS_95 is designed in a modular, toolbox fashion that allows the user to experiment with the optimal control algorithms and construct new algorithms. The programs `outer` and `aug_lagrng`, described in detail in [126], are examples of this toolbox approach to constructing algorithms.

B.3 Class of Optimal Control Problems Solvable by RIOTS_95

RIOTS_95 is designed to solve optimal control problems of the form

$$\textbf{OCP}: \quad \min_{(u,\xi)\in L_\infty^m[a,b]\times R^n} \left\{ f(u,\xi) \doteq g_o(\xi, x(b)) + \int_a^b l_o(t,x,u)\,\mathrm{d}t \right\},$$

$$\begin{aligned}
\text{subject to} \quad & \dot{x} = h(t,x,u), \qquad x(a) = \xi, \quad t \in [a,b],\\
& u_{\min}^j(t) \le u^j(t) \le u_{\max}^j(t), \quad j = 1,\dots,m,\\
& \xi_{\min}^j \le \xi^j \le \xi_{\max}^j, \quad j = 1,\dots,n,\\
& l_{ti}^\nu\big(t, x(t), u(t)\big) \le 0, \quad \nu \in \mathbf{q}_{ti},\ t \in [a,b],\\
& g_{ei}^\nu\big(\xi, x(b)\big) \le 0, \quad \nu \in \mathbf{q}_{ei},\\
& g_{ee}^\nu\big(\xi, x(b)\big) = 0, \quad \nu \in \mathbf{q}_{ee},
\end{aligned}$$

where $x(t) \in R^n$, $u(t) \in R^m$, $g : R^n \times R^n \to R$, $l : R \times R^n \times R^m \to R$, and $h : R \times R^n \times R^m \to R^n$, and we have used the notation $\mathbf{q} \doteq 1,\dots,q$, and $L_\infty^m[a,b]$ is the space of Lebesgue-measurable, essentially bounded functions $[a,b] \to R^m$. The functions in **OCP** can also depend upon parameters that are passed from MATLAB at execution time using `get_flags`. Refer to [126], Sect. 4, for details.

The subscripts o, ti, ei, and ee on the functions $g(\cdot,\cdot)$ and $l(\cdot,\cdot,\cdot)$ stand for, respectively, "objective function", "trajectory constraint", "endpoint inequality constraint", and "endpoint equality constraint". The subscripts for $g(\cdot,\cdot)$ and $l(\cdot,\cdot,\cdot)$ are omitted when all functions are being considered without regard to the subscript. The functions in the description of problem **OCP** and the derivatives of these functions,[1] must be supplied by the user as either object code or M-files. The bounds on the components of xi and u are specified on the MATLAB command line at runtime.

The optimal control problem **OCP** allows optimization over both the control u and one or more of the initial states ξ. To be concise, we will define the variable

$$\eta = (u,\xi) \in H_2 \doteq L_\infty^m[a,b] \times R^n.$$

[1]If the user does not supply derivatives, the problem can still be solved using RIOTS with finite-difference computation of the gradients.

With this notation, we can write, for example, $f(\eta)$ instead of $f(\xi, u)$. We define the inner product on H_2 as

$$\langle \eta_1, \eta_2 \rangle_{H_2} \doteq \langle u_1, u_2 \rangle_{L_2} + \langle \xi_1, \xi_2 \rangle.$$

The norm corresponding to this inner product is given by $\| \eta \|_{H_2} = \langle \eta, \eta \rangle_{H_2}^{1/2}$. Note that H_2 is a pre-Hilbert space.

Appendix C
Implementations

C.1 Remote Sensors Trajectory Optimization

In this section, we provide the file required to simulate the example given in Chap. 4. Table C.1 gives the main MATLAB program used to define the initial conditions of the problem and call the RIOTS function. Table C.2 gives the function `sys_init.m`, which provides information about the dimensions of the optimization problem. Table C.3 gives the function `sys_h.m` in which the dynamic model is defined. Table C.4 gives the function `sys_g.m`, which is used to compute the endpoint cost function. Table C.5 gives the function `sys_l.m`, which is used to compute values for the integrands of cost functions. Table C.6 gives the function `interp_sensitivities.m`, which is used to estimate the value of the sensitivity coefficients at a given location.

Table C.1 Main function to call RIOTS used in Chap. 4

```
load sensitivities
global WGHT_CTRL

n_sensors = 3; % number of sensors
n_ctrls = 3 * n_sensors; % number of control input
WGHT_CTRL = 2.0 / n_ctrls;
n_sensor_dynamics = n_ctrls; % number of sensor dynamics
s0 = [0.1; 0.1; 0.2 ; ...
     0.1; 0.5; 0.2 ; ...
     0.1; 0.9 ; 0.2]; % initial conditions for 3 sensors
s_lower = zeros(n_sensor_dynamics, 1);
s_upper = ones(n_sensor_dynamics, 1);
n_df_ctrl = length(TGRID) + 1;
u0 = [0.4*ones(1, n_df_ctrl); % 1 sensor
      0.0*ones(1, n_df_ctrl);
      0.0*ones(1, n_df_ctrl)];
  u0= [u0 ; 0.4*ones(1, n_df_ctrl); % 2 sensors
      0.0*ones(1, n_df_ctrl);
      0.0*ones(1, n_df_ctrl)];
  u0= [u0 ; 0.4*ones(1, n_df_ctrl); % 3 sensors
      0.0*ones(1, n_df_ctrl);
      0.0*ones(1, n_df_ctrl)];
u_min = [-0.7 ; -0.7 ; -0.2 ; ...
         -0.7 ; -0.7 ; -0.2; ...
         -0.7 ; -0.7 ; -0.2]; % minimum control input
u_max = [0.7 ; 0.7 ; 0.2 ; ...
         0.7 ; 0.7 ; 0.2; ...
         0.7 ; 0.7 ; 0.2]; % maximum control input

n_params = size(SENSVS, 4); % number of parameters
n_additional_state_vars = n_params * (n_params + 1) / 2;
x0 = [s0; zeros(n_additional_state_vars, 1)];
x0_lower = [s_lower; zeros(n_additional_state_vars, 1)];
x0_upper = [s_upper; zeros(n_additional_state_vars, 1)];
fixed    = [zeros(n_sensor_dynamics, 1); ...
             ones(n_additional_state_vars, 1)];
X0 = [x0, fixed, x0_lower, x0_upper];
[u, x, crit_val] = riots(X0, u0, TGRID, u_min, u_max, ...
                         [], [300, 0, 1], 4);
```

Table C.2 `sys_init.m` file for RIOTS used in Chap. 4

```
function neq = sys_init(params)

global SENSVS

if isempty(params)
   n_sensors = 3;
   n_controls = 3 * n_sensors;
   n_parameters = size(SENSVS, 4);
   n_states = 3 * n_sensors + n_parameters ...
                 * (n_parameters + 1) / 2;
   neq = [1 n_states; 2 n_controls];
else
   global sys_params
   sys_params = params;
end
```

Table C.3 `sys_h.m` file for RIOTS used in Chap. 4.

```
function xdot = sys_h(neq, t, x, u)

global sys_params IND_TRIANGLE

n_sensor_dynamics = neq(2);
n_sensors = round(neq(2) / 3);
n_parameters = round(sqrt(IND_TRIANGLE(end)));

x1 = x(1: 3: n_sensor_dynamics - 2);
x2 = x(2: 3: n_sensor_dynamics - 1);
x3 = x(3: 3: n_sensor_dynamics);
u1 = u(1: 3: n_sensor_dynamics - 2);
u2 = u(2: 3: n_sensor_dynamics - 1);
u3 = u(3: 3: n_sensor_dynamics);
g = interp_sensitivities(x1, x2, x3, u1, u2, u3, t, neq(4));
a = zeros(n_parameters, n_parameters);
for loop = 1: n_sensors
    a = a + g(loop, :)' * g(loop, :);
end

xdot = [u; a(IND_TRIANGLE)];
```

Table C.4 `sys_g.m` file for RIOTS used in Chap. 4

```
function J = sys_g(neq, t, x0, xf)

global sys_params IND_TRIANGLE

n_sensor_dynamics = neq(2);
n_parameters = round(sqrt(IND_TRIANGLE(end)));

F_NUM = neq(5);

if F_NUM == 1
  fim = zeros(n_parameters, n_parameters);
  fim(IND_TRIANGLE) = xf(n_sensor_dynamics + 1: end);
  fim = fim';
  fim(IND_TRIANGLE) = xf(n_sensor_dynamics + 1: end);
  J = -log(det(fim));
else
  error('Reference to a non-existing constraint ...
              on initial/final state')
end
```

Table C.5 `sys_l.m` file for RIOTS used in Chap. 4

```
function z = sys_l(neq,t,x,u)

global sys_params WGHT_CTRL

F_NUM = neq(5);
n_sensor_dynamics = neq(2);
n_sensors = round(neq(2) / 3);

x3 = x(3: 3: n_sensor_dynamics);

if F_NUM == 1
  z = 0.1 / sqrt(x3' * x3);
else
  error('Reference to a non-existing state constraint')
end
```

Table C.6 `interp_sensitivities.m` file for RIOTS used in Chap. 4

```
function g = interp_sensitivities(x,y,z,xdot,ydot,zdot,t,k)
global TGRID XGRID YGRID SENSVS
wr = (t - TGRID(k)) / (TGRID(k + 1) - TGRID(k));
wl = 1.0 - wr;
g = zeros(length(x), size(SENSVS, 4));

[thetadot,phidot,rdot] = cart2sph(xdot,ydot,zdot);
phidot = phidot - pi/2;

rTR = rdot;
thetaTR = thetadot - pi/4 ;
phiTR = phidot + pi/6;
[xTR,yTR,zTR] = sph2cart(thetaTR,phiTR,rTR);
xTR = x + xTR;
yTR = y + yTR;
zTR = z + zTR;
xGTR = x + z.*(xTR-x)./(z-zTR);
yGTR = y + z.*(yTR-y)./(z-zTR);

rTL = rdot;
thetaTL = thetadot + pi/4 ;
phiTL = phidot + pi/6;
[xTL,yTL,zTL] = sph2cart(thetaTL,phiTL,rTL);
xTL = x + xTL;
yTL = y + yTL;
zTL = z + zTL;
xGTL = x + z.*(xTL-x)./(z-zTL);
yGTL = y + z.*(yTL-y)./(z-zTL);

rBR = rdot;
thetaBR = thetadot + pi/4 ;
phiBR = phidot - pi/6;
[xBR,yBR,zBR] = sph2cart(thetaBR,phiBR,rBR);
xBR = x + xBR;
yBR = y + yBR;
zBR = z + zBR;
xGBR = x + z.*(xBR-x)./(z-zBR);
yGBR = y + z.*(yBR-y)./(z-zBR);

rBL = rdot;
thetaBL = thetadot - pi/4 ;
phiBL = phidot - pi/6;
[xBL,yBL,zBL] = sph2cart(thetaBL,phiBL,rBL);
xBL = x + xBL;
yBL = y + yBL;
zBL = z + zBL;
xGBL = x + z.*(xBL-x)./(z-zBL);
yGBL = y + z.*(yBL-y)./(z-zBL);
```

Table C.6 (Continued)

```
for j = 1 : length(x)
    res = 3;
    XI = zeros(res,res);
    YI = zeros(res,res);

    T = [linspace(xGTL(j),xGTR(j),res); ...
            linspace(yGTL(j),yGTR(j),res)];
    B = [linspace(xGBL(j),xGBR(j),res); ...
            linspace(yGBL(j),yGBR(j),res)];
    % L = [linspace(xGTL,xGBL,res);linspace(yGTL,yGBL,res)];
    % R = [linspace(xGTR,xGBR,res);linspace(yGTR,yGBR,res)];

    for i = 1:res
        XI(i,:) = linspace(T(1,i),B(1,i),res);
        YI(i,:) = linspace(T(2,i),B(2,i),res);
    end

    for loop = 1: size(SENSVS, 4)
        g(j, loop) = wl * sum(sum(interp2(XGRID, YGRID, ...
    SENSVS(:, :, k, loop), XI, YI, '*cubic', 0))) / res^2 ...
            + wr * sum(sum(interp2(XGRID, YGRID, ...
    SENSVS(:, :, k + 1, loop), XI, YI, '*cubic', 0))) / res^2;
    end
end

end
```

C.2 Online Scheme for Trajectory Optimization

In this section, we provide the file required to simulate the example given in Chap. 5. Table C.7 gives the main MATLAB program used to define the initial conditions of the problem and call the RIOTS function. Table C.8 gives the function `sys_init.m`, which provides information about the dimensions of the optimization problem. Table C.9 gives the function `sys_h.m` in which the dynamic model is defined. Table C.10 gives the function `sys_g.m`, which is used to compute the endpoint cost function. Table C.11 gives the function `sys_l.m`, which is used to compute values for the integrands of cost functions. Table C.12 gives the function `interp_sensitivities.m`, which is used to estimate the value of the sensitivity coefficients at a given location. Table C.13 gives the function `paramestimatenonlin.m`, which is used to estimate the parameters of the system based in a set of measurements.

Table C.7 Main function to call RIOTS used in Chap. 5

```
global WGHT_CTRL
load sensitivities
load simulations
startup;
a = 0.1;
b = 0.6;
c = 0.8;
reala = 0.1;
realb = 0.6;
realc = 0.8;
n_sensors = 3; %to change also in sys_init
n_params = size(SENSVS, 4);
n_ctrls = 2 * n_sensors;
WGHT_CTRL = 2.0 / n_ctrls;
n_sensor_dynamics = n_ctrls;
u_min = -0.6;
u_max =  0.6;
s0 = [0.1; 0.1; 0.1; 0.5; 0.1; 0.9];
s_lower = zeros(n_sensor_dynamics, 1);
s_upper = ones(n_sensor_dynamics, 1);
ob_int = 10;
ob_num = 10;

n_df_ctrl = ob_num + 1 ;
u0 = [0.4*ones(1, n_df_ctrl);    0.0*ones(1, n_df_ctrl);
    0.4*ones(1, n_df_ctrl);    0.0*ones(1, n_df_ctrl);
    0.4*ones(1, n_df_ctrl);    0.0*ones(1, n_df_ctrl)];
n_additional_state_vars = n_params * (n_params + 1) / 2;
x0 = [s0; zeros(n_additional_state_vars, 1)];
x0_lower = [s_lower; zeros(n_additional_state_vars, 1)];
x0_upper = [s_upper; zeros(n_additional_state_vars, 1)];
fixed    = [ones(n_sensor_dynamics, 1); ...
            ones(n_additional_state_vars, 1)];
X0 = [x0, fixed, x0_lower, x0_upper];

% Definition of the initial conditions
% for the first iteration of the sensitivity
n_xgrid_divs = 20;
n_ygrid_divs = n_xgrid_divs;
pdesize = (n_xgrid_divs + 1)*(n_ygrid_divs + 1);
w0 = [repmat(0, pdesize, 1); zeros(n_sensors * pdesize, 1)];
param=[]; traj=[]; timeest=[]; measest=[]; xest=[];
%
```

Table C.7 (Continued)

```
for i=1:ob_int
    timehor = linspace((i-1)/ob_int,(i-1)/ob_int+1,ob_num+1);
    timeopt = linspace((i-1)/ob_int,i/ob_int,ob_num+1);
    [SENSVS,unused] = sensitivity(a,b,c,timehor,w0);
    [unused,w] = sensitivity(a,b,c,timehor,w0);
    unused = [];
    w0 = [w(1:pdesize,end);zeros(n_sensors * pdesize, 1)];
    n_additional_state_vars = n_params * (n_params + 1) / 2;
    X0 = [x0, fixed, x0_lower, x0_upper];
    [u, x, crit_val] = riots(X0, u0, timehor, u_min ...
    * ones(n_ctrls, 1), u_max * ones(n_ctrls, 1), ...
     [], 200, 4, [], 10, 2);
    x = interp1(timehor',x',timeopt','cubic')';
    x0 = x(:,end);
    u0 = [u(1,end)*ones(1, n_df_ctrl);
    u(2,end)*ones(1, n_df_ctrl);
    u(3,end)*ones(1, n_df_ctrl);
    u(4,end)*ones(1, n_df_ctrl);
    u(5,end)*ones(1, n_df_ctrl);
    u(6,end)*ones(1, n_df_ctrl)];
    traj=[traj,x];
    SENSVS = [];
    for j=1:n_sensors
    meas(j,:) = interpn(XGRID,YGRID,TGRID,PGRID1,PGRID2, ...
                PGRID3,SIMS,x(2*j-1,:),x(2*j,:), ...
                timeopt, reala*ones(length(timeopt),1)', ...
                realb*ones(length(timeopt),1)', ...
                realc*ones(length(timeopt),1)','cubic');
    end
    meas = meas + 0.0001*randn(n_sensors,length(meas));

    timeest=[timeest timeopt(:,2:end)];
    measest=[measest meas(:,2:end)];
    xest=[xest x(:,2:end)];

    a = paramestimatenonlin(measest, xest, timeest, ...
             XGRID, YGRID, TGRID, PGRID1, PGRID2, PGRID3, ...
              SIMS, n_sensors, [a,b,c]);
    b = a(2);
    c = a(3);
    a = a(1);
    param=[param,[a;b;c]];
end
```

Table C.8 `sys_init.m` file for RIOTS used in Chap. 5

```
function neq = sys_init(params)

global SENSVS

if isempty(params)
   n_sensors = 3;
   n_controls = 2 * n_sensors;
   n_parameters = size(SENSVS, 4);
   n_states = 2 * n_sensors + n_parameters ...
                * (n_parameters + 1) / 2;
   neq = [1 n_states; 2 n_controls ];
else
   global sys_params
   sys_params = params;
end
end
```

Table C.9 `sys_h.m` file for RIOTS used in Chap. 5

```
function xdot = sys_h(neq, t, x, u)

global sys_params IND_TRIANGLE

n_sensor_dynamics = neq(2);
n_sensors = round(neq(2) / 2);
n_parameters = round(sqrt(IND_TRIANGLE(end)));

x1 = x(1: 2: n_sensor_dynamics - 1);
x2 = x(2: 2: n_sensor_dynamics);
g = interp_sensitivities(x1, x2, t, neq(4));
a = zeros(n_parameters, n_parameters);
for loop = 1: n_sensors
    a = a + g(loop, :)' * g(loop, :);
end

xdot = [u; a(IND_TRIANGLE)];
```

Table C.10 `sys_g.m` file for RIOTS used in Chap. 5

```
function J = sys_g(neq, t, x0, xf)

global sys_params IND_TRIANGLE

n_sensor_dynamics = neq(2);
n_parameters = round(sqrt(IND_TRIANGLE(end)));

F_NUM = neq(5);

if F_NUM == 1
  fim = zeros(n_parameters, n_parameters);
  fim(IND_TRIANGLE) = xf(n_sensor_dynamics + 1: end);
  fim = fim';
  fim(IND_TRIANGLE) = xf(n_sensor_dynamics + 1: end);
  J = -log(det(fim));
else
  error('Reference to a non-existing ...
     constraint on initial/final state')
end
```

Table C.11 `sys_l.m` file for RIOTS used in Chap. 5

```
function z = l(neq,t,x,u)

global sys_params WGHT_CTRL

F_NUM = neq(5);

if F_NUM == 1
  z = 0;
else
  error('Reference to a non-existing state constraint')
end
```

Table C.12 `interp_sensitivities.m` file for RIOTS used in Chap. 5

```
function g = interp_sensitivities(x, y, t, k)
global TGRID XGRID YGRID SENSVS
wr = (t - TGRID(k)) / (TGRID(k + 1) - TGRID(k));
wl = 1.0 - wr;
g = zeros(length(x), size(SENSVS, 4));
for loop = 1: size(SENSVS, 4)
   g(:, loop) = wl * interp2(XGRID, YGRID, ...
                     SENSVS(:, :, k, loop), x, y, '*cubic') ...
                  + wr * interp2(XGRID, YGRID, ...
                     SENSVS(:, :, k + 1, loop), x, y, '*cubic');
end
```

Table C.13 `paramestimatenonlin.m` file for RIOTS used in Chap. 5

```
function a = paramestimatenonlin(meas, x, time, XGRID, ...
   YGRID, TGRID, PGRID1, PGRID2, PGRID3, SIMS, n_sensors, a)
options = optimset('TolFun',1e-4,'MaxTime',500);
a = lsqnonlin(@myfun ,a ,0 ,1 ,options);

function F = myfun(a)

    for j = 1:n_sensors
    est = interpn(XGRID,YGRID,TGRID,PGRID1,PGRID2,PGRID3, ...
                    SIMS, x(2*j-1,:),x(2*j,:),time, ...
                    a(1)*ones(length(time),1)', ...
                    a(2)*ones(length(time),1)', ...
                    a(3)*ones(length(time),1)','cubic');
    F(j) =  sum(meas(j,:)-est);
    end

end

end
```

C.3 Fractional-Order Trajectory Optimization

In this section, we provide the file required to simulate the example given in Chap. 7. Table C.14 gives the main MATLAB program used to define the initial conditions of the problem and call the RIOTS function. Table C.15 gives the function `sys_init.m`, which provides information about the dimensions of the optimization problem. Table C.16 gives the function `sys_h.m` in which the dynamic model is defined. Table C.17 gives the function `sys_g.m`, which is used to compute the endpoint cost function. Table C.18 gives the function `sys_l.m`, which is used to compute values for the integrands of cost functions. Table C.19 gives the function `interp_sensitivities.m`, which is used to estimate the value of the sensitivity coefficients at a given location.

Table C.14 Main function to call RIOTS used in Chap. 7.

```
clear all
load sensitivities
load res09.mat

global WGHT_CTRL A b c

n=5;
A = res{n}.A;
b = res{n}.b;
c = res{n}.c;
sys=ss(A,b,c,0);

n_sensors = 3;
n_ctrls = 2 * n_sensors * length(c');
WGHT_CTRL = 2.0 / (2 * n_sensors);
n_sensor_dynamics = n_ctrls;
u_min = -0.7;
u_max =  0.7;
s0 = 0.1*[1;zeros(length(c')-1,1)] ...
            / (c*[1;zeros(length(c')-1,1)]);
s0 = [s0 ; 0.2*c'/(c*c')];
s0 = [s0 ; 0.1*c'/(c*c')];
s0 = [s0 ; 0.5*c'/(c*c')];
s0 = [s0 ; 0.1*c'/(c*c')];
s0 = [s0 ; 0.8*c'/(c*c')];
s_lower = zeros(n_sensor_dynamics, 1);
s_upper = ones(n_sensor_dynamics, 1);
n_df_ctrl = length(TGRID) + 1;
u0 = [0.4*ones(1, n_df_ctrl);      0.0*ones(1, n_df_ctrl);
      0.4*ones(1, n_df_ctrl);      0.0*ones(1, n_df_ctrl);
      0.4*ones(1, n_df_ctrl);      0.0*ones(1, n_df_ctrl)];

n_params = size(SENSVS, 4);
n_additional_state_vars = n_params * (n_params + 1) / 2;
x0 = [s0; zeros(n_additional_state_vars, 1)];
x0_lower = [s_lower; zeros(n_additional_state_vars, 1)];
x0_upper = [s_upper; zeros(n_additional_state_vars, 1)];
fixed    = [zeros(n_sensor_dynamics, 1); ...
            ones(n_additional_state_vars, 1)];
X0 = [x0, fixed, x0_lower, x0_upper];
[u, x, crit_val] = riots(X0, u0, TGRID, ...
                 u_min * ones(2 * n_sensors, 1), ...
                 u_max * ones(2 * n_sensors, 1), ...
                 [], [100, 0, 1], 4);
```

Table C.15 `sys_init.m` file for RIOTS used in Chap. 7.

```
function neq = sys_init(params)

global SENSVS c

if isempty(params)
   n_sensors = 3;
   n_controls = 2 * n_sensors;
   n_parameters = size(SENSVS, 4);
   n_states = 2 * n_sensors * length(c') ...
    + n_parameters * (n_parameters + 1) / 2;
   neq = [1 n_states; 2 n_controls ];
else
   global sys_params
   sys_params = params;
end
```

Table C.16 `sys_h.m` file for RIOTS used in Chap. 7.

```
function xdot = sys_h(neq, t, x, u)

global sys_params IND_TRIANGLE A b c

n_sensor_dynamics = neq(2);
n_sensors = round(neq(2) / 2);
n_parameters = round(sqrt(IND_TRIANGLE(end)));

x1 = x(1: length(c'));
x2 = x(length(c') + 1 : 2 * length(c'));
x3 = x(2 * length(c') + 1 : 3 * length(c'));
x4 = x(3 * length(c') + 1 : 4 * length(c'));
x5 = x(4 * length(c') + 1 : 5 * length(c'));
x6 = x(5 * length(c') + 1 : 6 * length(c'));
a = zeros(n_parameters, n_parameters);
g = interp_sensitivities(c*x1, c*x2, t, neq(4));
a = a + g' * g;
g = interp_sensitivities(c*x3, c*x4, t, neq(4));
a = a + g' * g;
g = interp_sensitivities(c*x5, c*x6, t, neq(4));
a = a + g' * g;
state1 = A*x1 + b*u(1);
state2 = A*x2 + b*u(2);
state3 = A*x3 + b*u(3);
state4 = A*x4 + b*u(4);
state5 = A*x5 + b*u(5);
state6 = A*x6 + b*u(6);
xdot = [state1 ; state2 ; state3 ; state4 ; ...
         state5 ; state6 ; a(IND_TRIANGLE)];
```

Table C.17 `sys_g.m` file for RIOTS used in Chap. 7.

```
function J = sys_g(neq, t, x0, xf)

global sys_params IND_TRIANGLE c

n_sensor_dynamics = neq(2);
n_parameters = round(sqrt(IND_TRIANGLE(end)));

F_NUM = neq(5);

if F_NUM == 1
  fim = zeros(n_parameters, n_parameters);
  fim(IND_TRIANGLE) = xf(n_sensor_dynamics ...
                     * length(c') + 1: end);
  fim = fim';
  fim(IND_TRIANGLE) = xf(n_sensor_dynamics ...
                     * length(c') + 1: end);
  J = -log(det(fim))
else
  error('Reference to a non-existing ...
  constraint on initial/final state')
end
```

Table C.18 `sys_l.m` file for RIOTS used in Chap. 7

```
function z = l(neq,t,x,u)

global sys_params WGHT_CTRL

F_NUM = neq(5);

if F_NUM == 1
  z = 0;
else
  error('Reference to a non-existing state constraint')
end
```

Table C.19 `interp_sensitivities.m` file for RIOTS used in Chap. 7

```
function g = interp_sensitivities(x, y, t, k)
global TGRID XGRID YGRID SENSVS
wr = (t - TGRID(k)) / (TGRID(k + 1) - TGRID(k));
wl = 1.0 - wr;
g = zeros(length(x), size(SENSVS, 4));
for loop = 1: size(SENSVS, 4)
   g(:, loop) = wl * interp2(XGRID, YGRID, ...
                  SENSVS(:, :, k, loop), x, y, '*cubic') ...
                + wr * interp2(XGRID, YGRID, ...
                  SENSVS(:, :, k + 1, loop), x, y, '*cubic');
end
```

References

1. http://www.twitter.com
2. MAS-net: Mobile actuator sensor networks. http://mechatronics.ece.usu.edu/mas-net/, 2002
3. NSF Workshop on Cyber-Physical Systems. http://warma.ece.cmu.edu/cps/, October 2006
4. The First International Workshop on Cyber-Physical Systems. http://www.qhdctc.com/wcps2008/, June 2008
5. International Journal of Social Computing and Cyber-Physical Systems (2009) Interscience Publishers
6. Abdelzaher T (2006) Towards an architecture for distributed cyber-physical systems. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
7. Adam N (2009) Cyber-physical systems security. In: CSIIRW '09: proceedings of the 5th annual workshop on cyber security and information intelligence research, New York, NY, USA. ACM, New York, p 1
8. Afifi L, El Jai A, Zerrik E (2005) Regional analysis of linear distributed parameter systems. Princeton University Press, Princeton
9. Agrawal OP (1989) On a general formulation for the numerical solution of optimal control problems. Int J Control 50(2):627–638
10. Agrawal OP (2004) A general formulation and solution scheme for fractional optimal control problems. Nonlinear Dyn 38(1):323–337
11. Agrawal OP (2008) Fractional optimal control of a distributed system using eigenfunctions. ASME J Comput Nonlinear Dyn 3(2):021204-1–021204-6
12. Agrawal OP (2008) A quadratic numerical scheme for fractional optimal control problems. ASME J Dyn Syst Meas Control 130(1):011010-1–011010-6
13. Agrawal OP, Baleanu D (2007) A Hamiltonian formulation and a direct numerical scheme for fractional optimal control problems. J Vib Control 13(9-10):1269–1281
14. Amouroux M, Babary JP (1988) Sensor and control location problems. In: Singh MG (ed) Systems & control encyclopedia. Theory, technology, applications, vol 6. Pergamon Press, Oxford, pp 4238–4245
15. Atkins EM (2006) Cyber-physical aerospace: challenges and future directions in transportation and exploration systems. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
16. Atkinson AC, Donev AN (1992) Optimum experimental designs. Clarendon Press, Oxford
17. Baker DG (1982) Synoptic-scale and mesoscale contributions to objective operational maximum–minimum temperature forecast errors. Mon Weather Rev 110(3):163–169
18. Banks HT, Kunisch K (1989) Estimation techniques for distributed parameter systems. Systems & control: foundations & applications. Birkhäuser, Boston
19. Banks HT, Smith RC, Wang Y (1996) Smart material structures: modeling, estimation and control. Research in applied mathematics. Masson, Paris

20. Banks HT (1992) Control and estimation of distributed parameter systems. Society for Industrial and Applied Mathematics, Philadelphia
21. Batchelor GK (1967) An introduction to fluid dynamics. Cambridge University Press, Cambridge
22. Benestad R, Hanssen-Bauer I, Chen D (2008) Empirical-statistical downscaling. World Scientific, Singapore
23. Bennet AF (1992) Inverse methods in physical oceanography. Cambridge University Press, Cambridge
24. Bennet AF (2002) Inverse modeling of the ocean and atmosphere. Cambridge University Press, Cambridge
25. Bennet AF, Chua BS, Leslie LM (1996) Generalized inversion of a global numerical weather prediction model. Meteorol Atmos Phys 60(1–3):165–178
26. Boggs PT, Tolle JW (1995) Sequential quadratic programming. Acta Numer 4(1):1–51
27. Bonakdarpour B (2008) Challenges in transformation of existing real-time embedded systems to cyber-physical systems. SIGBED Rev 5(1):1–2
28. Burns A (2006) Modeling temporal behavior in complex cyber-physical systems: position paper. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
29. Campbell J, Goldstein S, Mowry T (2006) Cyber-physical systems. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
30. Cárdenas AA, Amin S, Sinopoli B, Giani A, Perrig A, Sastry S (2010) Challenges for securing cyber physical systems
31. Cassandras CG, Li W (2005) Sensor networks and cooperative control. Eur J Control 11(4–5):436–463
32. Chao H, Baumann M, Jensen A, Chen YQ, Cao Y, Ren W, McKee M (2008) Band-reconfigurable multi-UAV-based cooperative remote sensing for real-time water management and distributed irrigation control. In: Proceedings of the 2008 IFAC world congress, July 2008
33. Chen P (2005) Pattern formation in mobile wireless sensor networks. Master's thesis, Utah State University
34. Chen YQ (2008) Mobile actuator/sensor networks (MAS-net) for cyber-physical systems. USU ECE 6800 Graduate Colloquium, http://www.neng.usu.edu/classes/ece/6800/, September
35. Chen YQ, Moore KL (2002) Discretization schemes for fractional-order differentiators and integrators. IEEE Trans Circuits Syst I, Fundam Theory Appl 49(3):363–367
36. Chen YQ, Vinagre BM, Podlubny I (2004) Continued fraction expansion approaches to discretizing fractional order derivatives—an expository review. Nonlinear Dyn 38(1–4):155–170
37. Chen YQ (2006) Band-reconfigurable multi-UAV based cooperative remote sensing for real-time water management, distributed irrigation control and ecological inferential measurements. http://mechatronics.ece.usu.edu/uav+water/, May
38. Chen YQ (2009) Mobile actuator and sensor networks (MAS-net) for cyber-physical systems (CPS). http://www.ieeeiciea.org/2009/download/lecture4.pdf, May
39. Chen YQ, Moore KL, Song Z (2004) Diffusion boundary determination and zone control via mobile actuator-sensor networks (MAS-net): challenges and opportunities. In: Proceedings of SPIE: intelligent computing: theory and applications, vol 5421, pp 102–113
40. Chen YQ, Schwartz AL (2002) RIOTS_95—a MATLAB toolbox for solving general optimal control problems and its applications to chemical processes. In: Recent developments in optimization and optimal control in chemical engineering, pp 229–252. ISBN 81-7736-088-4
41. Chong C-Y, Kumar SP (2003) Sensor networks: evolution, opportunities, and challenges. Proc IEEE 91(8):1247–1256
42. Christensen JH, Räisänen J, Iversen T, Bjørge D, Christensen OB, Rummukainen M (2001) A synthesis of regional climate change simulations—a Scandinavian perspective. Geophys Res Lett 28:1003–1006

43. Christensen JH, Christensen OB (2003) Climate modelling: severe summertime flooding in Europe. Nature 421(6925):805–806
44. Christensen OB, Christensen JH, Machenhauer B, Botzet M (1998) Very high-resolution regional climate simulations over Scandinavia-present climate. J Climate 11:3204–3229
45. Christofides PD (2001) Nonlinear and robust control of PDE systems: methods and applications to transport-reaction processes. Systems & control: foundations & applications. Birkhäuser, Boston
46. Cook J (2006) Cyber-physical systems and the twenty-first century automobile. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
47. Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20(3):273–297
48. Culler D, Estrin D, Srivastava M (2004) Overview of sensor networks. IEEE Comput 37(8):41–49
49. Daescu DN, Navon IM (2004) Adaptive observations in the context of 4D-Var data assimilation. Meteorol Atmos Phys 85:205–226
50. Demetriou MA (2010) Design of consensus and adaptive-consensus filters for distributed parameter systems. Automatica 46:300–311
51. Demetriou MA (2010) Guidance of mobile actuator-plus-sensor networks for improved control and estimation of distributed parameter systems. IEEE Trans Autom Control 55:1570–1584
52. Demetriou MA, Hussein II (2009) Estimation of spatially distributed processes using mobile spatially distributed sensor network. SIAM J Control Optim 48:266–291
53. Douglas CC, Lodder R, Mandel J, Vodacek A (2006) Cyber-physical systems and wildland fire or contaminant identification and tracking dynamic data-driven application systems. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
54. Dunbar WB (2007) Distributed receding horizon control of dynamically coupled nonlinear systems. IEEE Trans Autom Control 52(7):1249–1263
55. El Jai A, Pritchard A (1993) Regional controllability of distributed systems. In: Curtain R, Bensoussan A, Lions J (eds) Analysis and optimization of systems: state and frequency domain approaches for infinite-dimensional systems. Lecture notes in control and information sciences, vol 185. Springer, Berlin, pp 326–335
56. El Jai A (1991) Distributed systems analysis via sensors and actuators. Sens Actuators A, Phys 29:1–11
57. Ermakov SM (ed) (1983) Mathematical theory of experimental design. Nauka, Moscow (in Russian)
58. Evensen G (2006) Data assimilation: the ensemble Kalman filter. Springer, New York
59. Eyre JR (1997) Variational assimilation of remotely-sensed observations of the atmosphere. J Meteorol Soc Jpn 75(1B):331–338
60. Fedorov VV, Hackl P (1997) Model-oriented design of experiments. Lecture notes in statistics. Springer, New York
61. Fedorov VV (1989) Optimal design with bounded density: optimization algorithms of the exchange type. J Stat Plan Inference 22:1–13
62. Fedorov VV, Hackl P (1997) Model-oriented design of experiments. Lecture notes in statistics. Springer, New York
63. Frederico G, Torres D (2006) Noether's theorem for fractional optimal control problems. In: Proc. of the 2nd IFAC workshop on fractional differentiation and its applications, Porto, Portugal, 19–21 July, 2006
64. Frederico G, Torres D (2007) Fractional conservation laws in optimal control theory. Nonlinear dynamics, November
65. Frederico G, Torres D (2008) Fractional optimal control in the sense of Caputo and the fractional Noether's theorem. Int Math Forum 3(10):479–493
66. Fuhrman T (2006) Position paper for NSF workshop on cyber-physical systems. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
67. Futagami T, Tzafestas SG, Sunahara Y (eds) (1989) Distributed parameter systems—modelling and simulation: proceedings of the IMACS—IFAC international symposium. Elsevier, New York

68. Garrett J, Moura J, Sandefilippo M (2006) Sensor-data driven proactive management of infrastructure systems. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
69. Gill H, Zhao W, Znati T (2008) Call for papers for a special issue of on distributed cyber physical systems. IEEE Trans Parallel Distrib Syst 19(8):1150–1151
70. Goddard S, Deogun JS (2006) Future mobile cyber-physical systems: spatio-temporal computational environments. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
71. Grades H Healthgrades fourth annual patient safety in American hospitals study
72. (CPS Steering Group) (2008) Cyber-physical systems executive summary. In: Cyber-physical systems summit. http://varma.ece.cmu.edu/summit/CPS-Executive-Summary.pdf
73. Gruver WA, Sachs E (1980) Algorithmic methods in optimal control. Pitman, London
74. Heemink AW, Verlaan M, Segers AJ (2000) Variance reduced ensemble Kalman filtering
75. Hou J, Sha L (2006) A reference architecture for building cyber-physical spaces for independent/assisted living. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
76. Hsu CS, Hou D (1990) Linear approximation of fractional transfer functions of distributed parameter systems. Electron Lett 26(15):1211–1213
77. Huang XY, Yang X (1996) Variational data assimilation with the Lorenz model. Springer, Berlin
78. Iqbal M, Lim HB (2009) A cyber-physical middleware framework for continuous monitoring of water distribution systems. In: SenSys '09: proceedings of the 7th ACM conference on embedded networked sensor systems. ACM, New York, pp 401–402
79. El Jai A (1991) Distributed systems analysis via sensors and actuators. Sens Actuators A, Phys 29(1):1–11
80. El Jai A, Simon M, Zerrik E (1993) Regional observability and sensor structures. Sens Actuators A, Phys 39(2):95–102
81. Jain N, Agrawal DP (2005) Current trends in wireless sensor network design. Int J Dist Sensor Netw 1:101–122
82. Jennings LS, Fisher ME, Teo KL, Goh CJ (2002) MISER 3: optimal control software, version 2.0. Theory and user manual. Department of Mathematics, University of Western Australia, Nedlands
83. Jensen A, Baumann M, Chen YQ (2008) Low-cost multispectral aerial imaging using autonomous runway-free small flying wing vehicles. In: Proceedings of the 2008 geoscience and remote sensing symposium (IGARSS 2008)
84. Jeremić A, Nehorai A (1998) Design of chemical sensor arrays for monitoring disposal sites on the ocean floor. IEEE Trans Oceanic Eng 23(4):334–343
85. Jeremić A, Nehorai A (2000) Landmine detection and localization using chemical sensor array processing. IEEE Trans Signal Process 48(5):1295–1305
86. Kaheil YH, Gill MK, McKee M, Bastidas LA, Rosero E (2008) Downscaling and assimilation of surface soil moisture using ground truth measurements. IEEE Trans Geosci Remote Sens 46(5):1375–1384
87. Kiefer J, Wolfowitz J (1959) Optimum designs in regression problems. Ann Math Stat 30:271–294
88. Kim C, Yun H, Sun M, Mohan S, Al-Nayeem A, Sha L, Abdelzaher T (2009) A framework for wireless integration in interoperable real-time medical systems. Department of Computer Science, University of Illinois at Urbana-Champaign
89. Kim JW, Chang JT, Baker NL, Wilks DS, Gates WL (1984) The statistical problem of climate inversion: determination of the relationship between local and large-scale climate. Mon Weather Rev 112(10):2069–2077
90. Klein WH (1948) Winter precipitation as related to the 700-millibar circulation. Bull Am Meteorol Soc 29(9):439–453
91. Koushanfar F, Zhong L (2006) Building an integrated cyber-infrastructure: a micro/macro health services case study. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
92. Kubrusly CS, Malebranche H (1985) Sensors and controllers location in distributed systems—a survey. Automatica 21(2):117–128

93. Kumar S, Seinfeld J (2003) Optimal location of measurements for distributed parameter estimation. IEEE Trans Autom Control 23(4):690–698
94. Lee EA (2007) Computing foundations and practice for cyber-physical systems: a preliminary report. Technical report UCB/EECS-2007-72, http://chess.eecs.berkeley.edu/pubs/306.html, University of California, Berkeley, May
95. Lee EA (2008) Cyber physical systems: design challenges. Technical report, EECS Department, University of California, Berkeley, January
96. Lee I, Sokolsky O (2010) Medical cyber physical systems. In: Proceedings of the 47th design automation conference. ACM, New York, pp 743–748
97. Liberatore V (2007) Networked cyber-physical systems: an introduction. In: Proceedings of the 2007 NSF workshop on data management for mobile sensor networks
98. Lorenc AC (1986) Analysis methods for numerical weather prediction. Q J R Meteorol Soc 112(474):1177–1194
99. McMillin B, Gill C, Crow ML, Liu F, Niehaus D, Potthast A, Tauritz D (2006) Cyber-physical systems engineering: the advanced power grid. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
100. Menke W (1984) Geophysical data analysis: discrete inverse theory. Academic Press, San Diego
101. Mitola J (2000) An integrated agent architecture for software defined radio
102. Moore KL, Chen YQ, Song Z (2004) Diffusion-based path planning in mobile actuator–sensor networks (MAS-net): some preliminary results. In: Proceedings of SPIE: intelligent computing: theory and applications, vol 5421, pp 58–69
103. Nakano K, Sagara S (1981) Optimal measurement problem for a stochastic distributed parameter system with movable sensors. Int J Syst Sci 12(12):1429–1445
104. Navon IM (1997) Practical and theoretical aspects of adjoint parameter estimation and identifiability in meteorology and oceanography. Dyn Atmos Ocean 27:55–79
105. Nehorai A, Porat B, Paldi E (1995) Detection and localization of vapor-emitting sources. IEEE Trans Signal Process 43(1):243–253
106. Noble BD, Flinn J (2006) Wireless, self-organizing cyber-physical systems. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
107. Ögren P, Fiorelli E, Leonard NE (2004) Cooperative control of mobile sensor networks: adaptive gradient climbing in a distributed environment. IEEE Trans Autom Control 49(8):1292–1302
108. Oldham K, Spanier J (1974) The fractional calculus: theory and applications of differentiation and integration to arbitrary order. Mathematics in science and engineering, vol V. Academic Press, San Diego
109. Omatu S, Seinfeld JH (1989) Distributed parameter systems: theory and applications. Oxford mathematical monographs. Oxford University Press, New York
110. Oustaloup A, Levron F, Mathieu B (2000) Frequency-band complex noninteger differentiator: characterization and synthesis. IEEE Trans Circuits Syst I, Fundam Theory Appl 47(1):25–39
111. Patan M (2004) Optimal observation strategies for parameter estimation of distributed systems. PhD thesis, University of Zielona Góra, Zielona Góra, Poland
112. Patan M, Tricaud C, Chen YQ (2008) Resource-constrained sensor routing for parameter estimation of distributed systems. In: Proc. 17th IFAC world congress, Seoul, Korea. Published on CD-ROM
113. Paustian K (2007) Fundamentals of soil ecology, 2nd ed. Agric Syst 94(2):603–604
114. Polak E (1993) On the use of consistent approximations in the solution of semi-infinite optimization and optimal control problems. Math Program: Ser A and B 62(2):385–414
115. Polak E (1997) Optimization. Algorithms and consistent approximations. Applied mathematical sciences. Springer, New York
116. Porat B, Nehorai A (1996) Localizing vapor-emitting sources by moving sensors. IEEE Trans Signal Process 44(4):1018–1021
117. Pukelsheim F (2006) Optimal design of experiments (classics in applied mathematics), vol 50. Society for Industrial and Applied Mathematics, Philadelphia

118. Quereshi ZH, Ng TS, Goodwin GC (1980) Optimum experimental design for identification of distributed parameter systems. Int J Control 31(1):21–29
119. Rafajłowicz E (1986) Optimum choice of moving sensor trajectories for distributed parameter system identification. Int J Control 43(5):1441–1451
120. Rafajłowicz E (1996) Algorithms of experimental design with implementations in MATHEMATICA. Akademicka Oficyna Wydawnicza PLJ, Warsaw (in Polish)
121. Ramaprasad H (2008) Providing end-to-end guarantees in cyber-physical systems. Technical report, Southern Illinois University Carbondale, October
122. Reichle RH, Walker JP, Koster RD, Houser PR (2002) Extended versus ensemble Kalman filtering for land data assimilation. J Hydrometeorol 3:728–740
123. Reichle RH (2008) Data assimilation methods in the earth sciences. Adv Water Resour 31(11):1411–1418 Hydrologic Remote Sensing
124. Sagara S, Nakano K, Soji K (1980) Optimal sensor allocation strategies considering observability in linear distributed-parameter systems. Electr Eng Jpn 100(4):135–142
125. Schwartz AL (1996) Theory and implementation of numerical methods based on Runge-Kutta integration for solving optimal control problems. PhD thesis, University of California, Berkeley
126. Schwartz AL, Polak E, Chen YQ (1997) A MATLAB toolbox for solving optimal control problems. Version 1.0 for Windows, May
127. Seuffert G, Wilker H, Viterbo P, Drusch M, Mahfouf J-F (2004) The usage of screen-level parameters and microwave brightness temperature for soil moisture analysis. J Hydrometeorol 5:516–531
128. Sinopoli B, Sharp C, Schenato L, Schaffert S, Sastry SS (2003) Distributed control applications within sensor networks. Proc IEEE 91(8):1235–1246
129. Smith RC, Demetriou M (2003) Research directions in distributed parameter systems. Society for Industrial and Applied Mathematics, Philadelphia
130. Song Z, Chen YQ, Liang J, Uciński D (2005) Optimal mobile sensor motion planning under nonholonomic constraints for parameter estimation of distributed parameter systems. In: IEEE/RSJ International Conference on Intelligent Robots and Systems
131. Song Z, Chen YQ, Sastry CR, Tas NC (2009) Optimal observation for cyber-physical systems: a fisher-information-matrix-based approach. Springer, Berlin
132. Sun N-Z (1994) Inverse problems in groundwater modeling. Theory and applications of transport in porous media. Kluwer Academic, Dordrecht
133. Tabuada P (2006) Cyber-physical systems: position paper. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
134. Tan Y, Goddard S, Pérez LC (2008) A prototype architecture for cyber-physical systems. SIGBED Rev 5(1):1–2
135. Tan Y, Vuran MC, Goddard S (2009) Spatio-temporal event model for cyber-physical systems. In: International conference on distributed computing systems workshops, pp 44–50
136. Tang H, McMillin BM (2008) Security property violation in CPS through timing. In: ICDCSW ’08: proceedings of the 2008 28th international conference on distributed computing systems workshops, Washington, DC, USA, 2008. IEEE Comput Soc, Los Alamitos, pp 519–524
137. Tarantola A (1987) Inverse problem theory. Methods for data fitting and model parameter estimation. Amsterdam, Elsevier
138. Tarantola A, Valette B (1982) Inverse problems = quest for information. J Geophys 50:159–170
139. Tilmans HAC (1997) Equivalent circuit representation of electromechanical transducers: II. Distributed-parameter systems. J Micromech Microeng 7(4):285
140. Tippett MK, Anderson JL, Bishop CH, Hamill TM, Whitaker JS (2003) Ensemble square root filters. Mon Weather Rev 131:1485–1490
141. Tricaud C, Chen YQ (2008) Optimal mobile sensing policy for parameter estimation of distributed parameter systems finite horizon closed-loop solution. In: Proceedings of the 18th international symposium on mathematical theory of networks and systems (MTNS08), July 2008. SIAM, Philadelphia

142. Tricaud C, Chen YQ (2008) Solving fractional order optimal control problems in RIOTS_95—a general purpose optimal control problems solver. In: Proceedings of the 3rd IFAC workshop on fractional differentiation and its applications, November 2008
143. Tricaud C, Chen YQ (2010) Time-optimal control of systems with fractional dynamics. International Journal of Differential Equations
144. Tricaud C, Patan M, Uciński D, Chen YQ (2008) D-optimal trajectory design of heterogeneous mobile sensors for parameter estimation of distributed systems. In: Proc. 2008 American control conference, Seattle, Washington, USA, 2008. Published on CD-ROM
145. Tsai JJP, Yu PS (2009) Machine learning in cyber trust: security, privacy, and reliability. Springer, Berlin
146. Uciński D (2000) Optimal sensor location for parameter estimation of distributed processes. International Journal of Control 73(13)
147. Uciński D (2005) Optimal measurement methods for distributed-parameter system identification. CRC Press, Boca Raton
148. Uciński D (1998) A robust approach to the design of optimal trajectories of moving sensors for distributed-parameter systems identification. In: Beghi A, Finesso L, Picci G (eds) Proc 13th int symp mathematical theory of networks and systems, Padova, Italy, 6–10 July, 1998. Il Poligrafo, Padova, pp 551–554
149. Uciński D (1998) Towards a robust-design approach to optimal location of moving sensors in parameter identification of DPS. In: Domek S, Kaszyński R, Tarasiejski L (eds) Proc 5th int symp methods and models in automation and robotics, Międzyzdroje, Poland, 25–29 August, 1998, vol 1. Wyd Uczelniane Polit Szczecińskiej, Szczecin, pp 85–90
150. Uciński D (1999) Measurement optimization for parameter estimation in distributed systems. Technical University Press, Zielona Góra
151. Uciński D (1999) A technique of robust sensor allocation for parameter estimation in distributed systems. In: Frank PM (ed) Proc 5th European control conf, Karlsruhe, Germany, August 31–September 3. Published on CD-ROM. 1999, EUCA
152. Uciński D (2005) Sensor network design for parameter estimation of distributed systems using nonsmooth optimality criteria. In: Proc. 44th IEEE conference on decision and control, and the European control conference 2005, Seville, Spain, 2005. Published on CD-ROM
153. Uciński D, Chen YQ (2005) Time-optimal path planning of moving sensors for parameter estimation of distributed systems. In: Proc 44th IEEE conference on decision and control, and the European control conference 2005, Seville, Spain, 2005. Published on CD-ROM
154. Uciński D, Chen YQ (2006) Sensor motion planning in distributed parameter systems using Turing's measure of conditioning. In: Proc 45th IEEE conference on decision and control, San Diego, CA, 2006. Published on CD-ROM
155. Uciński D, Korbicz J (1999) On robust design of sensor trajectories for parameter estimation of distributed systems. In: Proc. 14th IFAC world congress, Beijing, China, 5–9 July, 1999. Modeling, identification, signal processing I, vol H, pp 73–78
156. Uciński D, Korbicz J (2001) Optimal sensor allocation for parameter estimation in distributed systems. J Inverse Ill-Posed Probl 9(3):301–317
157. Ulieru M (2009) eNetworks in an increasingly volatile world: Design for resilience of networked critical infrastructures. In: Proceedings of the IEEE conference on digital ecosystems 2009
158. van de Wal M, de Jager B (2001) A review of methods for input/output selection. Automatica 37:487–510
159. von Storch H, Hewitson B, Mearns L (2000) Review of empirical downscaling techniques. Technical report, Regional Climate Development Under Global Warming
160. von Storch H, Zorita E, Cubasch U (1993) Downscaling of global climate change estimates to regional scales: an application to Iberian rainfall in wintertime. J Climate 6:1161–1171
161. von Stryk O (1999) User's guide for DIRCOL, a direct collocation method for the numerical solution of optimal control problems. Version 2.1. Fachgebiet Simulation und Systemoptimierung, Technische Universität Darmstadt, November
162. Walter É, Pronzato L (1997) Identification of parametric models from experimental data. Communications and control engineering. Springer, Berlin

163. Bai EW, Fu M, Tempo R, Ye Y (1998) Analytic center approach to parameter estimation: convergence analysis, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.52.4371
164. West R, Parmer G (2006) A software architecture for next-generation cyber-physical systems. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
165. Wikipedia. Control systems. Wikibooks, the open-content textbooks collection, 2008
166. Wilby RL, Hassan H, Hanaki K (1998) Statistical downscaling of hydrometeorological variables using general circulation model output. J Hydrol 205(1–2):1–19
167. Wilks DS (2006) Statistical methods in the atmospheric sciences: an introduction. Amsterdam, Elsevier
168. Work D, Bayen A, Jacobson Q (2008) Automotive cyber-physical systems in the context of human mobility. National workshop on high-confidence automotive cyber-physical systems
169. Xia F (2008) QoS challenges and opportunities in wireless sensor/actuator networks. Sensors 8(2):1099–1110
170. Xie F (2006) Component-based cyber-physical systems. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
171. Xue D, Chen YQ, Atherton D (2007) Linear feedback control: analysis and design with MATLAB. Philadelphia, SIAM
172. Yau DK, Hou JC, Mallikarjun S, Rao NS (2006) Systems support for radiational plume detection, identification, and tracking sensor-cyber networks. In: Proceedings of the 2006 NSF workshop on cyber-physical systems
173. Zamani M, Karimi-Ghartemani M, Sadati N (2007) FOPID controller design for robust performance using particle swarm optimization. Fract Calc Appl Anal 10(2):169–188
174. Zhao F, Guibas LJ (2004) Wireless sensor networks: an information processing approach. Kaufmann, Amsterdam
175. Zöbel D (2006) Autonomous driving in goods transport. In: Proceedings of the 2006 NSF workshop on cyber-physical systems

Index

A
A-optimality criterion, 26
Actuation function, 82
Actuator, 18

C
Communication topology, 71
Controllability, 21
Cyber-physical system (CPS), 1
  applications, 3
  modeling, 6
  resilience, 6
  security, 5

D
D-optimality criterion, 26, 32, 35
Data assimilation, 120, 127
Distributed parameter system, 7, 17, 31
  applications, 7
  definition, 17, 24, 32
Downscaling, 117
  definition, 118

E
E-optimality criterion, 26
Extended Kalman filter, 123

F
Fisher information matrix, 25, 32, 34
Fractional derivative
  Caputo, 98
  Riemann–Liouville, 98
Fractional optimal control problem, 13
  definition, 99
  linear time invariant, 107
  linear time variant, 108
  optimal sensing, 111
  reformulation, 101, 105
Fractional order cyber-physical systems, 12
Fractional order systems, 12

G
Global circulation model, 118
Global climate model, 118

H
Hankel matrix, 103
Harmful algal blooms, 10
Heterogeneous sensor, 33

I
Impulse response, 102
Interlaced scheme, 66

K
Kaheil and McKee's algorithm, 124

M
MAS-net, 9
Measurement, 25
Mobile actuator, 81

O
Observability, 8, 24
Optimal actuation, 81, 82
Optimal control formulation, 56

P
Parameter estimation, 9
Parameter identification, 24, 25
Pathwise state constraints, 35

Q
Quality of service (QoS), 6

R
Rational approximation, 13
  fractional integrator, 104
  impulse response based, 102
  Oustaloup recursive approximation, 99
  state space realization, 101
Remote sensing, 52
Remote sensing function, 53, 59
Robust design, 28
  bounded parameters, 45
Runge–Kutta integration, 40

S
Sensitivity criterion, 26
Sensitivity function, 26
Sensor, 19
Sensor clustering, 26
Soil moisture, 124
State estimation, 8
Support vector machine, 127

U
Unmanned aerial imagery, 119

V
Variational data assimilation, 123

W
Wildfire control, 11
Wynn–Fedorov algorithm, 39