



Riccardo Zoppoli
Marcello Sanguineti
Giorgio Gnecco
Thomas Parisini

Neural Approximations for Optimal Control and Decision

Communications and Control Engineering

Series Editors

Alberto Isidori, Roma, Italy

Jan H. van Schuppen, Amsterdam, The Netherlands

Eduardo D. Sontag, Boston, USA

Miroslav Krstic, La Jolla, USA

Communications and Control Engineering is a high-level academic monograph series publishing research in control and systems theory, control engineering and communications. It has worldwide distribution to engineers, researchers, educators (several of the titles in this series find use as advanced textbooks although that is not their primary purpose), and libraries.

The series reflects the major technological and mathematical advances that have a great impact in the fields of communication and control. The range of areas to which control and systems theory is applied is broadening rapidly with particular growth being noticeable in the fields of finance and biologically-inspired control. Books in this series generally pull together many related research threads in more mature areas of the subject than the highly-specialised volumes of *Lecture Notes in Control and Information Sciences*. This series's mathematical and control-theoretic emphasis is complemented by *Advances in Industrial Control* which provides a much more applied, engineering-oriented outlook.

Indexed by SCOPUS and Engineering Index.

Publishing Ethics: Researchers should conduct their research from research proposal to publication in line with best practices and codes of conduct of relevant professional bodies and/or national and international regulatory bodies. For more details on individual ethics matters please see:

<https://www.springer.com/gp/authors-editors/journal-author/journal-author-helpdesk/publishing-ethics/14214>

More information about this series at <http://www.springer.com/series/61>

Riccardo Zoppoli · Marcello Sanguineti ·
Giorgio Gnecco · Thomas Parisini

Neural Approximations for Optimal Control and Decision



Springer

Riccardo Zoppoli
DIBRIS
Università di Genova
Genoa, Italy

Giorgio Gnecco 
AXES Research Unit
IMT—School of Advanced
Studies Lucca
Lucca, Italy

Marcello Sanguineti 
DIBRIS
Università di Genova
Genoa, Italy

Thomas Parisini 
Imperial College London
London, UK
University of Trieste
Trieste, Italy

ISSN 0178-5354 ISSN 2197-7119 (electronic)
Communications and Control Engineering
ISBN 978-3-030-29691-9 ISBN 978-3-030-29693-3 (eBook)
<https://doi.org/10.1007/978-3-030-29693-3>

© Springer Nature Switzerland AG 2020

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

*To my beloved wife, Maria Giovanna,
and our sons,
Federica and Gabriele
Riccardo Zoppoli*

*To my parents, Enrico and Maria, to my aunt,
Gianna, and
in memory of my grandmother, Caterina
Marcello Sanguineti*

*To my mother, Rosanna, and in memory
of my father, Giuseppe
Giorgio Gnecco*

*To my daughter, Francesca, to my beloved
wife, Germana, and
in memory of Sandro
Thomas Parisini*

Preface

Many scientific and technological areas of major interest require one to solve infinite-dimensional optimization problems, also called functional optimization problems. In such a context, one has to minimize (or maximize) a functional with respect to admissible solutions belonging to infinite-dimensional spaces of functions, often dependent on a large number of variables. This is the case, for example, with analysis and design of large-scale communication and traffic networks, stochastic optimal control of nonlinear dynamic systems with a large number of state variables, optimal management of complex team organizations, freeway traffic congestion control, optimal management of reservoir systems, reconstruction of unknown environments, Web exploration, etc. Typically, infinite dimension makes inapplicable many mathematical tools used in finite-dimensional optimization.

The subject of the book is the approximate solution of infinite-dimensional optimization problems arising in the fields of optimal control and decision. When optimal solutions to such problems cannot be found analytically and/or numerical solutions are not easily implementable, classical approaches to find approximate solutions often incur at least one form of the so-called “curse of dimensionality” (e.g., an extremely fast growth—with respect to the number of variables of the admissible solutions—of the computational load required to obtain suboptimal solutions within a desired accuracy).

The book aims at presenting into a unified framework theories and algorithms to solve approximately infinite-dimensional optimization problems whose admissible solutions may depend on a large number of variables. A multiplicity of tools, approaches, and results originating from different fields (e.g., functional optimization, optimal control and decision, neural computation, linear and nonlinear approximation theory, stochastic approximation, statistical and deterministic machine learning theories, Monte Carlo sampling, and quasi-Monte Carlo sampling) are brought together to construct a novel, general, and mathematically sound

optimization methodology. The basic tool is the use of a family of nonlinear approximators, which include commonly used neural networks, to reduce infinite-dimensional optimization problems to finite-dimensional nonlinear programming ones.

When admissible solutions depend on a large number of variables, as it typically happens in the kind of problems we are mostly interested in the book, various instances of the abovementioned curse of dimensionality may arise with respect to a suitably defined “dimension” of the task one is facing. Hence, we devote particular attention to develop a theoretical apparatus and algorithmic tools capable of coping with such a drawback. To this end, suitable computational models and sampling techniques are combined to cope with the curse of dimensionality in approximation of multivariable functions and in sampling high-dimensional domains, respectively.

A word has to be said on the role played in the book by data and learning from data. More specifically, in our approach, we try to exploit the fruitful exchange between machine learning and optimization. Indeed, while machine learning exploits optimization models and algorithms, it simultaneously poses problems which often constitute optimization challenges. This cross-fertilization is particularly evident in the problems dealt with in the book.

The major special feature of the monograph is the fact that, to the best of the authors’ knowledge, it is the first one explicitly devoted to the subject of neural networks and other nonlinear approximators for the solution of infinite-dimensional optimization problems arising in optimal control and decision.

The main benefit of reading the book is the possibility of understanding and becoming capable to apply the abovementioned optimization methodology, whose ingredients come from scientific fields with high potentiality of interaction, but whose joint use seems to be still limited. Indeed, the book is conceived to combine into a unified framework of the authors’ different expertises. Their complementary backgrounds are exploited to develop solid mathematical foundations, a powerful solution methodology, and efficient algorithms. This represents the very strength of the book, which deals with problems whose complexity makes scientific and technological “contaminations” mandatory to be successful in finding accurate suboptimal solutions.

The examples provided, which often deal with real-world applications, are detailed numerically, so as to allow the reader not only to clearly understand the proposed methodology but also to compare it in practice with traditional approaches.

The book offers

- A thorough illustration of theoretical insights to solve approximately infinite-dimensional optimization problems whose admissible solutions depend on large numbers of variables.

- A derivation of the theoretical properties of a methodology of approximate infinite-dimensional optimization (named “Extended Ritz Method”, or shortly, ERIM), based on families of nonlinear approximators which include commonly used shallow and deep neural networks as special cases.
- An overview of classical numerical computational methods for optimal control and decision (e.g., discrete-time dynamic programming, batch and stochastic gradient techniques, and the Ritz method).
- Bounds on the errors of the approximate solutions.
- Efficient algorithms for a wide range of problems, e.g., optimal control and decision in a deterministic and in a stochastic environment, over a finite and an infinite time horizon, with and without guaranteed upper bounds on the approximation error, with a centralized and a decentralized structure.
- Several examples, often dealing with real-world applications of major interest (such as routing in communications networks, freeway traffic congestion control, and optimal management of reservoir systems, etc.), whose approximate solutions are detailed and compared numerically.

Thanks to its multidisciplinary nature, this monograph can be of interest to researchers, postgraduates, 4th–3rd-year undergraduates, and practitioners in Automatic Control, Operations Research, Computer Science, and various branches of Engineering and Economics. In general, the prerequisites are basic concepts of multivariate calculus, introductory probability theory, matrix–vector algebra, optimization, and control engineering. Sometimes, elementary concepts and tools from functional analysis are used, too. For the reader’s convenience, more advanced techniques are introduced when needed.

Finally, we are thankful to our institutions and to a number of scholars and colleagues for their contributions to the book. We wish to thank first and foremost Marco Baglietto and Angelo Alessandri for the discussions on most topics and, in particular, the former for suggestions about team optimal control theory. Then, we mention Cristiano Cervellera, who introduced us to important concepts of quasi-Monte Carlo methods and deterministic learning theory. Our work took advantage of discussions with Vera Kurková and Paul Kainen on mathematical foundations of approximation and optimization via neural networks. Mauro Gaggero and Serena Ivaldi helped us in deriving numerical solutions for some examples. Aldo Grattarola clarified us some aspects of probability theory. We are most grateful to Alfredo Bellen, Stefano Maset, and Marino Zennaro for suggestions and advice on specific approximation theory concepts and to several other colleagues, among which Federico Girosi, Frank Lewis, and Marios M. Polycarpou, whose suggestions and

collaborations over the years greatly improved the content of the book. Mrs. Daniela Solari supported the exchange of drafts among the authors and Mrs. Altea Ariano reviewed the English of the whole manuscript and Dr. Riccardo M. G. Ferrari helped in preparing several drawings. The last author acknowledges the research support funding of the European Union's Horizon 2020 Research and Innovation Programme under grant agreement No. 739551 (KIOS CoE), of ABB and of Danieli Group. The authors wish to thank their Publisher Mr. Oliver Jackson for his help, support, encouragement, and patience during this project and for the high-quality production of the book.

May 2019

Riccardo Zoppoli
University of Genoa, Genoa, Italy

Marcello Sanguineti
University of Genoa, Genoa, Italy

Giorgio Gnecco
IMT—School of Advanced Studies Lucca, Lucca, Italy

Thomas Parisini
Imperial College London, London, UK
University of Trieste, Trieste, Italy

Contents

1	The Basic Infinite-Dimensional or Functional Optimization Problem	1
1.1	General Comments on Infinite-Dimensional or Functional Optimization	3
1.1.1	IDO and FDO Problems	3
1.1.2	From the Ritz Method to the Extended Ritz Method (ERIM)	7
1.1.3	Approximation of Functions	10
1.1.4	From Function Approximation to Approximate Infinite-Dimensional Optimization	12
1.1.5	Relationships with Parametrized Control Approaches	13
1.2	Contents and Structure of the Book	14
1.3	Infinite-Dimensional Optimization	20
1.3.1	Statement of the Problem	20
1.3.2	Finite-Dimensional Versus Infinite-Dimensional Optimization	23
1.4	General Conventions and Assumptions	25
1.4.1	Existence and Uniqueness of Minimizers	26
1.4.2	Other Definitions, Conventions, and Assumptions	28
1.5	Examples of Infinite-Dimensional Optimization Problems	29
1.5.1	A Deterministic Continuous-Time Optimal Control Problem	29
1.5.2	A Continuous-Time Network Flow Problem	31
1.5.3	A T -Stage Stochastic Optimal Control Problem	32
1.5.4	An Optimal Estimation Problem	33
1.5.5	A Static Team Optimal Control Problem	34
	References	35

2 From Functional Optimization to Nonlinear Programming by the Extended Ritz Method	39
2.1 Fixed-Structure Parametrized (FSP) Functions	41
2.2 The Sequence of Nonlinear Programming Problems Obtained by FSP Functions of Increasing Complexity	43
2.2.1 The Case of Problem P	44
2.2.2 The Case of Problem PM	46
2.3 Solution of the Nonlinear Programming Problem P_n	50
2.4 Optimizing FSP Functions	53
2.5 Polynomially Complex Optimizing FSP Functions	56
2.5.1 The Growth of the Dimension d	58
2.5.2 Polynomial and Exponential Growths of the Model Complexity n with the Dimension d	60
2.6 Approximating Sets	65
2.7 Polynomially Complex Approximating Sequences of Sets	69
2.7.1 The Worst-Case Error of Approximation of Functions	70
2.7.2 Polynomial and Exponential Growth of the Model Complexity n with the Dimension d	71
2.8 Connections Between Approximating Sequences and Optimizing Sequences	73
2.8.1 From S^d -Approximating Sequences of Sets to P^d -Optimizing Sequences of FSP Functions	75
2.8.2 From P^d -Optimizing Sequences of FSP Functions to Polynomially Complex P^d -Optimizing Sequences of FSP Functions	78
2.8.3 Final Remarks	84
2.9 Notes on the Practical Application of the ERIM	85
References	86
3 Some Families of FSP Functions and Their Properties	89
3.1 Linear Combinations of Fixed-Basis Functions	91
3.2 One-Hidden-Layer Networks	92
3.2.1 The Structure of OHL Networks	92
3.2.2 A More Abstract View	94
3.2.3 Tensor-Product, Ridge, and Radial Constructions	96
3.3 Multi-Hidden-Layer Networks	99
3.4 Terminology	100
3.5 Kernel Smoothing Models	101
3.6 Density Properties	106
3.6.1 \mathcal{C} - and \mathcal{L}_p -Density Properties	106
3.6.2 The Case of Ridge OHL Networks	107

3.6.3	The Case of Radial OHL Networks	111
3.6.4	Multiple Hidden Layers and Multiple Outputs	112
3.7	From Universality of Approximation to Moderate Model Complexity	112
3.7.1	The Maurey–Jones–Barron Theorem	114
3.7.2	Case Study: Sigmoidal OHL Networks	119
3.7.3	Lower Bounds on Approximation Rates	123
3.7.4	On the Approximation Accuracy of MHL Networks	127
3.8	Extensions of the Maurey–Jones–Barron Theorem	128
3.8.1	Geometric Rates of Approximation	129
3.8.2	An Upper Bound in Terms of the \mathcal{G} -Variation Norm	130
3.8.3	Estimating the \mathcal{G} -Variation	133
3.8.4	Some OHL Networks with the Same Approximation Rate	134
3.9	The ERIM Versus the Classical Ritz Method	135
3.9.1	Some Concepts from the Theory of Hilbert Spaces	137
3.9.2	The IDO Problem Used for the Comparison	137
3.9.3	Approximate Solution by the Ritz Method	139
3.9.4	The Curse of Dimensionality in the Ritz Method and the Possibility of Avoiding it in the ERIM	141
3.10	Rates of Optimization by the ERIM	142
	References	145
4	Design of Mathematical Models by Learning From Data and FSP Functions	151
4.1	Learning from Data	152
4.2	Expected Risk, Empirical Risk, and Generalization Error in Statistical Learning Theory	156
4.2.1	The Expected Risk	156
4.2.2	The Empirical Risk	158
4.2.3	The Generalization Error	160
4.2.4	The Approximation and Estimation Errors are Components of the Generalization Error	161
4.3	Bounds on the Generalization Error in Statistical Learning Theory	164
4.3.1	Probabilistic Convergence of the Empirical Risk	165
4.3.2	The VC Dimension of a Set of Functions	168
4.3.3	Bounds on the Estimation Error	171
4.3.4	Bounds on the Generalization Error for Gaussian OHL Networks	176
4.3.5	Bounds on the Generalization Error for Sigmoidal OHL Networks	181

4.4	Deterministic Learning Theory	187
4.4.1	Estimation of Functions Without Noise: The Expected Risk, the Empirical Risk, and the Generalization Error	187
4.4.2	Deterministic Convergence of the Empirical Risk	189
4.4.3	Discrepancy of a Sequence	190
4.4.4	Variation of a Function and Conditions for Bounded Variations	191
4.4.5	Examples of Functions with Bounded Hardy and Krause Variation	195
4.4.6	The Koksma–Hlawka Inequality	197
4.4.7	Sample Sets Coming from (t, d) -Sequences	199
4.4.8	Bounds on Rate of Convergence of the Estimation Error	200
4.4.9	Estimation of Functions in the Presence of Additive Noise	203
	References	205
5	Numerical Methods for Integration and Search for Minima	207
5.1	Numerical Computation of Integrals	208
5.1.1	Integration by Regular Grids	209
5.1.2	Integration by MC Methods	210
5.1.3	Integration by Quasi-MC Methods	214
5.1.4	Limitations of the Quasi-MC Integration with Respect to the MC One	215
5.2	Numerical Computation of Expected Values	217
5.2.1	Computation of Expected Values by Quasi-MC Methods	218
5.2.2	Improving MC and Quasi-MC Estimates of Integrals and Expected Values	219
5.3	Minimization Techniques for the Solution of Problem P_n	221
5.3.1	Direct Search Methods	222
5.3.2	Gradient-Based Methods	223
5.3.3	Stochastic Approximation: Robbins–Monro Algorithm for the Solution of Nonlinear Root-Finding Problems	226
5.3.4	Robbins–Monro Algorithm and the Stochastic Gradient Method are Particular Cases of a General Stochastic Algorithm	229
5.3.5	Convergence of the General Stochastic Algorithm	231
5.3.6	Convergence of the Root-Finding Robbins–Monro Stochastic Approximation Algorithm	235

5.3.7	Choice of the Stepsize Sequence	237
5.3.8	Convergence of the Stochastic Gradient Algorithm	239
5.3.9	Global Optimization via Stochastic Approximation	242
5.4	Monte Carlo and Quasi-Monte Carlo Search for Minima	245
5.4.1	Monte Carlo Random Search	245
5.4.2	Quasi-Monte Carlo Search	248
	References	251
6	Deterministic Optimal Control over a Finite Horizon	255
6.1	Statement of Problem C1	256
6.2	A Constructive Example for Understanding the Basic Concepts of Dynamic Programming	257
6.3	Solution of Problem C1 by Dynamic Programming: The Exact Approach	264
6.4	Sampling of the State Space and Application of FSP Functions: Approximate Dynamic Programming	268
6.4.1	Sampling of the State Space	268
6.4.2	The Recursive Equation of ADP	270
6.4.3	Solving the Minimization Problems in the ADP Equation	275
6.4.4	The Kernel Smoothing Models in ADP	278
6.5	Computation of the Optimal Controls in the Forward Phase of ADP	282
6.6	Generalization Error Owing to the Use of Sampling Procedures and FSP Functions	284
6.6.1	Approximation Errors and Approximating FSP Functions	286
6.6.2	Estimation Errors in the Context of Deterministic Learning Theory	289
6.7	Reduction of Problem C1 to an NLP Problem	292
6.7.1	Solution of Problem C1 by Gradient-Based Algorithms	292
6.7.2	The Final Time of Problem C1 is not Fixed	296
	References	297
7	Stochastic Optimal Control with Perfect State Information over a Finite Horizon	299
7.1	Statement of Problems C2 and C2'	300
7.2	Solution of Problems C2 and C2' by Dynamic Programming: The Exact Approach	302
7.3	Sampling of the State Space, Application of FSP Functions, and Computation of Expected Values: Approximate Dynamic Programming	307

7.3.1	Sampling of the State Space and Approximations by FSP Functions	307
7.3.2	Computation of Expected Values in ADP Recursive Equations	311
7.3.3	Computation of the Optimal Controls in the Forward Phase of Dynamic Programming	314
7.3.4	Error Propagation of the Optimal Cost-to-Go Functions Starting from Approximations of the Optimal Control Functions	316
7.4	Some Notes on Approximate Dynamic Programming	320
7.4.1	Discretization Issues	320
7.4.2	Approximation Issues	323
7.4.3	Some Structural Properties of the Optimal Cost-to-Go and Control Functions	324
7.4.4	Reinforcement Learning	326
7.4.5	Q-Learning	331
7.5	Approximate Solution of Problems C2 and C2' by the ERIM	333
7.5.1	Approximation of the IDO Problem C2 by the Nonlinear Programming Problems C2 _n	333
7.5.2	Approximation of Problems C2' by the Nonlinear Programming Problems C2' _n	336
7.6	Solution of Problems C2 _n and C2' _n	340
7.6.1	Solution of Problem C2 _n by Nonlinear Programming	340
7.6.2	Computation of the Gradient $\nabla_{u_0, w_n} J(u_0, w_n, x_0, \xi)$	343
7.6.3	Computation of the Gradient $\nabla_{u_0, w_n} J(u_0, w_n, x_0, \xi)$ When MHL Networks Are Used	349
7.7	Comparison Between the ERIM and ADP	356
7.7.1	The Model of the Reservoir System	356
7.7.2	Computational Aspects	360
7.7.3	Concluding Comments	366
7.8	Stochastic Optimal Control Problems Equivalent to Problem C2'	368
7.8.1	Measurable Time-Invariant Random Variables	369
7.8.2	The Random Variables $\xi_0, \xi_1, \dots, \xi_{T-1}$ are Measurable	374
7.8.3	Tracking a Stochastic Reference	377
	References	379

8 Stochastic Optimal Control with Imperfect State Information over a Finite Horizon	383
8.1 Statement of Problem C3	385
8.2 Solution of Problem C3 by Dynamic Programming	386
8.2.1 Reduction of Problem C3 to Problem C2	387
8.2.2 Derivation of the Conditional Probability Densities Needed by Dynamic Programming	389
8.3 Approximate Solution of Problem C3 by the ERIM	391
8.3.1 Approximation of Problem C3 by the Nonlinear Programming Problems C3 _n	392
8.3.2 Solution of Problem C3 _n	393
8.3.3 Specialization of Stochastic Gradient Algorithms When MHL Networks are Used	394
8.4 Approximation of Problem C3 by Limiting the Dimension of the Information State Vector and by Applying the ERIM	398
8.4.1 Limited-Memory Information Vector	398
8.4.2 Approximate Solution of Problem C3-LM by the ERIM	401
8.4.3 Solution of Problem C3 _n -LM by Stochastic Approximation and Specialization to MHL Networks	402
8.5 Approximate Solution of Problem C3-LM by the Certainty Equivalence Principle and the ERIM	407
8.5.1 Certainty Equivalence Principle	408
8.5.2 Applying the CE Principle to the LM Controller	411
8.6 Numerical Results	416
8.6.1 Comparison Between an LQG Optimal Control Law and Limited-Memory Optimal Control Laws	416
8.6.2 Freeway Traffic Optimal Control	418
References	425
9 Team Optimal Control Problems	427
9.1 Basic Concepts of Team Theory	428
9.1.1 Statement of the Team Optimal Control Problem	429
9.1.2 Partially Nested Information Structures	430
9.1.3 Person-by-Person Optimality	433
9.2 Team Problems with Known Optimal Solution	435
9.2.1 LQG Static Teams	435
9.2.2 LQG Dynamic Teams with Partially Nested Information Structure	437
9.2.3 Sequential Partitions	438

9.3	The Optimal Decentralized Control Problem and its Reduction to a Nonlinear Programming Problem	439
9.3.1	Statement of Problem DC	439
9.3.2	Approximation of Problem DC by the Nonlinear Programming Problems DC _n	442
9.4	The Witsenhausen Counterexample	443
9.4.1	Statement of the Problem	444
9.4.2	Application of the ERIM for the Approximate Solution of Problem W	446
9.4.3	Numerical Results Obtained by the ERIM	447
9.4.4	Improved Numerical Results and Variants of the Witsenhausen Counterexample	452
9.4.5	Theoretical Results on the Application of the ERIM	455
9.5	Dynamic Routing in Traffic Networks	456
9.5.1	Modeling the Communication Network	457
9.5.2	Statement of the Dynamic Routing Problem	459
9.5.3	Approximate Solution of Problem ROUT by the ERIM	460
9.5.4	Computation of the Neural Routing Functions via Stochastic Approximation	462
9.5.5	Numerical Results	466
	References	468
10	Optimal Control Problems over an Infinite Horizon	471
10.1	Statement of the Basic Infinite-Horizon Optimal Control Problem	473
10.1.1	Deterministic Optimal Control over an Infinite Horizon	473
10.1.2	Stochastic Optimal Control with Perfect State Information over an Infinite Horizon	481
10.1.3	Stochastic Optimal Control with Imperfect State Information over an Infinite Horizon	484
10.2	From Finite to Infinite Horizon: The Deterministic Receding-Horizon Approximation	487
10.3	Stabilizing Properties of Deterministic RH Control Laws	490
10.4	Stabilizing Approximate RH Control Laws	499
10.5	Offline Design of the Stabilizing Approximate RH Control Law Obtained by the ERIM	502
10.6	Numerical Results	507
	References	510
	Index	513

Chapter 1

The Basic Infinite-Dimensional or Functional Optimization Problem



In the title of the book, there are three words that define its contents: “neural”, “approximations”, and “optimal”.

Let us consider a designer in a broad sense (an engineer, an economist, and so on), who has to solve an optimization problem. It is important to point out that the word “optimization” has to be understood in a way strictly dependent on the context. For instance, a wealthy gentleman may want to split a certain amount of money into a given number of investments so as to get the maximum profit. In this case, the term “optimization” takes on a clear meaning.

However, there are situations – important as well – in which the same word represents a means, rather than a goal. Let us consider, for example, the design of a controller of a multi-input, multi-output control system. In this case, the design problem may be reformulated in the framework of optimal control. Suppose that the well-known *linear-quadratic-Gaussian* (LQG) hypotheses are verified (i.e., the state equation and the measurement channel are linear, the cost is quadratic and convex, and the random noises are Gaussian) and that the problem is stated over an infinite time horizon (IH). Of course, the hypotheses “L” and “G” stem from the nature of the original problem, whereas the “Q” hypothesis may derive from the choice of the designer, who is free to choose the elements of the weight matrices in the quadratic forms of the cost. The preliminary requirement of the controller design has little to do with the minimization of the cost but consists in the asymptotic stability of the closed-loop system. Furthermore, there are other specifications to be satisfied, such as robustness with respect to perturbations in the dynamic system and in the characteristics of the noises, absence of too large and persistent oscillations in the controlled variables, accuracy in keeping the error between the desired reference variables and the controlled ones within a predetermined threshold, etc. Such design specifications can be obtained by suitably varying the weight matrices in the cost. Essentially, the aim is quite often not to design an optimal controller – the purpose

of which is to minimize a given cost – but to determine a “good” multivariable controller. Then, the term “optimization” has a very wide meaning.

Let us now focus on the word “approximation”, the interpretation of which deserves particular attention. Let us suppose that the optimization problem at hand has been stated unambiguously. To clarify what we mean by “unambiguously”, let us go back to the abovementioned design problem of an optimal controller (with optimality being our goal). Its statement is not ambiguous if, very optimistically, one knows exactly the dynamic system, the cost, and the properties of the random variables. Now, suppose that the LQG hypotheses do not hold and, more generally, that no hypotheses allowing one to get a closed-form optimal solution are verified. Then, the designer has to face the following dilemma: either (a) simplifying the problem by approximating its “objects” in such a way to get an “approximate version” of the problem itself – which can be solved in closed form – or (b) keeping the original problem, together with its intrinsic complexity, and trying to solve it via numerical approaches – which are necessarily of approximate nature. Clearly, both choices have pros and cons.

The approach (a) leads to find the solution to a “false” problem, in the sense that the latter differs from the original one. The validity of such a choice depends on the “distance” between the approximating problem and the “true” one and on the sensitivity of the optimal solution with respect to such a distance. By proceeding according to (b), on the contrary, one gets a solution that is surely suboptimal but may result to be “reasonable”. Moreover, such a suboptimal solution may suggest possible improvements to the designer. For these reasons, in the book, we follow the approach (b).

An example may be useful to further clarify the motivations at the basis of our choice. In Sect. 9.4, we shall apply the method proposed in the book to the approximate solution of the famous (and still unsolved) Witsenhausen counterexample. We got that a possible suboptimal solution is given for the first of the two *decision-makers* (DMs) – i.e., for DM_1 – by a decision function of the form of a “slightly sloped” piecewise-linear function (as mentioned in [54, p. 384], this was an “important finding”). The segments of such a staircase function show small unexpected oscillations, owing to the numerical algorithm used to derive them (see Fig. 9.12; see also Fig. 6a in [41]). Other authors have replaced the “imperfect” segments with straight segments, thus decreasing the cost of the decision process.

Another motivation for the choice (b) can be found in the following statement by Spall [69, p. 31], actually referring to a slightly different context than ours: “In applying methods from this book, it is important to keep in mind the fundamental goals in the scientific or other investigations: ‘Better a rough answer to the right question than an exact answer to the wrong one’ (aphorism possibly due to Lord Kelvin.”

As regards the adjective “neural”, it is strictly connected to the approximate solution technique that we shall exploit throughout the book. This will become clear when, later in this chapter, we shall describe such a technique.

Finally, it is worth pointing out that in most parts of the book we shall address optimal control problems. However, sometimes we shall also consider contexts where the term “decision” has a more general meaning. Hence, instead of the terms “control” and “control function,” we shall use “decision” and “decision function,” respectively. Indeed, we shall refer to “optimal control problems” and “optimal decision problems” as to synonyms.

1.1 General Comments on Infinite-Dimensional or Functional Optimization

The distinction between *infinite-dimensional optimization* (IDO) – also called *functional optimization* – and *finite-dimensional optimization* (FDO) – also called *mathematical programming*¹ – is rather clear and can be easily explained, as we shall see shortly on. Since, generally, these two optimization models refer to different areas and require different solution methodologies, in the literature they are addressed quite independently from each other. In this book, we deal with IDO problems that are generally more difficult to solve than FDO ones. As the proposed solution method is substantially based on reducing the solution of a given IDO problem to that of “easier” FDO problems, in this section we shall emphasize differences and similarities between these two optimization models.

1.1.1 IDO and FDO Problems

In general, in formulating an optimization problem, one has to identify a set \mathcal{S} , whose elements are called *admissible solutions* to the problem, and to associate a cost to each admissible solution by means of a *cost functional* F . The target consists in minimizing F with respect to admissible solutions belonging to \mathcal{S} , which represent the alternatives at disposal to a DM. Of course, the concepts are valid, with obvious changes, also in the case where the functional F represents a figure of merit to be maximized. (Throughout the book, without loss of generality, we shall refer to the minimization of cost functionals.)

In the family of FDO problems, the admissible solutions belong to a set $\mathcal{S} \subseteq \mathbb{R}^d$; hence, they consist in a *finite number* d of components of a vector. In the family of IDO problems, instead, the admissible solutions are decision functions that belong to *infinite-dimensional* spaces of functions. IDO is also called *functional optimization*,

¹That is, linear, linear integer, mixed, and nonlinear programming as a whole. We remark from the beginning that the kind of FDO problems considered in the book (as a means to get approximate solutions to IDO problems) is limited to nonlinear programming problems.

by analogy with the expression “functional analysis,” which is related to the branch of mathematics that studies function spaces.²

To clarify the (mostly qualitative) arguments contained in this section, it is useful to formalize the two classes of problems. Let us suppose, for simplicity, that there is a unique DM.

- **FDO problems or mathematical programming problems.** Find an optimal vector $\mathbf{x}^\circ \triangleq \text{col} (x_1^\circ, \dots, x_d^\circ)$ that minimizes a cost function $f(\mathbf{x})$, where $\mathbf{x} \in \mathcal{S} \subseteq \mathbb{R}^d$ and $f : \mathcal{S} \rightarrow \mathbb{R}$. \mathcal{S} is the set of admissible solutions (vectors³) of the problem.
- **IDO problems or functional optimization problems.** Find an optimal decision function γ° that minimizes a cost functional $F(\gamma)$, where $F : \mathcal{S} \rightarrow \mathbb{R}$, $\gamma \in \mathcal{S} \subseteq \mathcal{H}$, $\gamma : D \rightarrow C$, $D \subseteq \mathbb{R}^d$, $d \geq 1$, and $C \subseteq \mathbb{R}^m$, $m \geq 1$. The set $\mathcal{S} \subseteq \mathcal{H}$ is the set of the admissible solutions (functions) to the problem and is a subset of an infinite-dimensional linear space \mathcal{H} .

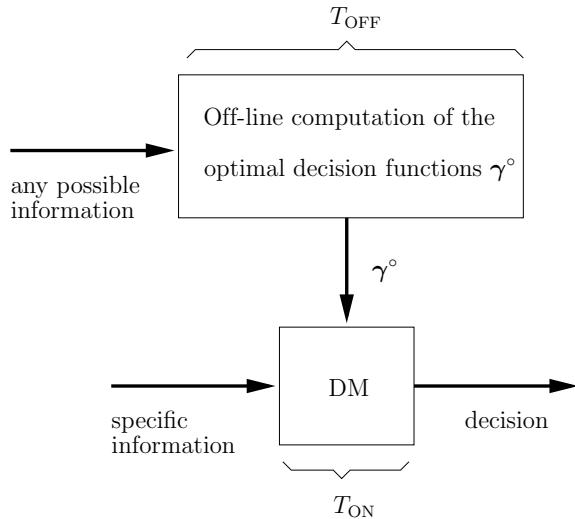
The reason why the admissible decision functions are required to belong to an infinite-dimensional space is quite obvious. When no a priori information is available, there is no rationale to specialize the functions by choosing, for example, a certain finite basis, in terms of elements of which they should be expressed as linear combinations. On one hand, the infinite dimension makes the solution more difficult to find (even approximately) but, on the other hand, it provides the decision functions with a large degree of freedom. The possibility of constraining them to belong to a well-suited finite-dimensional space is left to the capability of the designer. In doing so, the target consists in simplifying the search for a (sub)optimal solution, while guaranteeing a satisfactorily small error with respect to the optimal function, if it exists.

The applicative contexts that originate IDO problems can be quite clearly distinguished from those requiring FDO models. Considering once more, for simplicity, the case of a unique DM, let us try to understand when it requires a function on the basis of which the DM generates decisions. The need for such a function depends strongly on the computational time at the DM’s disposal. If the decision has to be generated “immediately after” the information has been acquired by the DM and the DM itself has not enough time to compute the optimal decision, it is clear that it has to rely on a *pre-computed* decision function. So, one has to *compute offline the decision function and to use it online*, via simple calculations that require an amount of time practically negligible with respect to the time required by the computation of the decision function itself (see Fig. 1.1). Such a difference between offline and online computational efforts allows one to quantify the above-used expression “immediately after.”

²The case in which one has a finite number of variables and an infinite number of constraints, or an infinite number of variables and a finite number of constraints, is referred to as *semi-infinite programming* [43, 70]. An example is the so-called “general design centering problem,” which consists in minimizing some measure, e.g., the volume, of a parametrized body contained in another body and has application, for instance, in robot manoeuvrability [39].

³Vectors will be denoted by boldface characters and considered as column vectors.

Fig. 1.1 Situation in which it is necessary to distinguish between offline and online computations. The time T_{ON} required by the DM to generate its decision is assumed to be negligible. The time T_{OFF} required to compute the optimal decision function γ° is assumed, instead, to be large



It is now worth pointing out that in the IDO problems, in which the decisions have not to be generated in a unique time instant but along time over a finite or an infinite horizon, we shall divide the time into equal intervals of duration Δt . At each instant $t_i = i \Delta t$, $i = 0, 1, \dots$, which we call *decision stage* i , the decision-maker DM_i becomes active and generates its optimal decision. Equivalently, one can consider a unique DM that exploits a specific decision function at each stage i . Usually, we shall adopt the convention of having decision-makers DM_0, DM_1, \dots , which become active sequentially over time. In the case of optimal control, this context is referred to as discrete-time optimal control.

Remark 1.1 We have used several terms for entities that make decisions. This may confuse the reader. So, we make the following distinction. If we consider an entity that generates decisions along time, we call it “decision-maker” – or “DM” for short (without subscripts) – or “controller” or “parameter estimator” or other terms according to the specific practical context. If we want to focus on the discrete-time instants or stages at which the decisions are taken, we use the terms “ DM_0, DM_1, \dots ,” with the subscripts denoting the specific time instants. There is a *decision function* for each DM_t , $t = 0, 1, \dots$ \triangleleft

There are cases in which the optimization problem intrinsically originates over stages. Consider, for instance, manufacturing systems where pieces are produced one at a time, store-and-forward mechanisms in communications and computer networks, economic or management systems in which decisions have to be made periodically, and so on. By contrast, for instance, systems governed by physical laws have to be generally described by models that are continuous-time in nature. Typically, they are based on ordinary or partial differential equations. The latter are required if distributed-parameter systems have to be modeled. In this case, the state variables

have to be characterized not only as functions of time but also as functions of spatial coordinates. By means of suitable numerical approximations using temporal (or spatial and temporal) discretizations, it is often possible to transform, in a “sufficiently accurate” way, the ordinary or partial differential equations into a finite number of discrete-time equations.

Our choice of using discrete time is motivated by the fact that this will allow us to treat the resulting optimization problems in an easier way with respect to the continuous-time formulation. In particular, discrete time will permit us to skip many mathematical technicalities (e.g., stochastic differential equations) and to focus on conceptual issues that we consider of major importance. Moreover, it is worth noting that the solution of continuous-time problems leads to optimal decision functions that are anyway implemented in discrete time through suitable digital devices. Thus, there is not a significant loss in generality in focusing on optimization problems stated in terms of discrete-time models from the very beginning.

Referring to the kind of FDO problems known as *nonlinear programming* (NLP), one aims at minimizing a function with respect to a finite number of real parameters. Hence, the term “decision function” – intended as a mapping that generates decisions on the basis of the available information – turns out to be less meaningful. The optimal design parameters are represented by the components of a “minimizing vector” belonging to the space \mathbb{R}^d for a suitable positive integer d . Thus, in an NLP problem, the minimizing vector is represented by the best choice in the set of admissible parameter values. Such parameters model, for example, the sizing requirements for electrical and electronic components and circuits, mechanical machineries and devices, civil engineering structures, proportional, integral, and derivative constants in *proportional–integral–derivative* (PID) regulators, and so on.

Typically, in IDO problems, the information becomes available to the DMs in the form of random variables. In Sect. 1.5, we shall show examples of IDO problems, which shall clarify the nature of the variables representing the information. Let us call such variables *information variables* or, as a whole, *information vector*. We anticipate here some classical examples. A large amount of the book deals with T -stage optimal control problems in stochastic frameworks. If the random inputs act only on the dynamic system and the state vector is perfectly measurable, then the state vector itself becomes the unique information vector acquired by DM_i at stage $i = 0, 1, \dots, T - 1$. Things are much more difficult if random noises also influence the state measurement channel. The same occurs if not every state component can be measured. In this case, the whole “history” of the dynamic system has to be “kept in mind.”

There are organizations in which several DMs aim at minimizing a common cost but have “personal histories,” owing to the fact that they acquire different measures on the same dynamic system. Such a context is studied by *team theory* in economics and in relevant technological problems. We cite, for example, the dynamic routing of data in traffic and data networks, where each routing node takes on the role of a DM.

Another field of great importance is represented by IDO problems in which decision functions yield estimates of the state or of parameters of dynamic systems on the

basis of measures coming from the available measurement devices (i.e., identification and state estimation problems in control theory), perform tasks of learning from data (classification, regression, etc.), implement fault diagnosis procedures for dynamic systems of practical interest (e.g., in aerospace, manufacturing and process industry, etc.), or extract information from large amounts of data obtained from complex systems. Indeed, biological, financial, medical data, data from telecommunications, trade, industry, Internet, and so on are pervasive in the present world. The challenge consists in dealing with such huge amounts of data with the aim of getting the whole picture of the complex system one is investigating. Typically, the short time available to the DMs for online information processing makes the offline computation of the optimal decision functions a must.

1.1.2 *From the Ritz Method to the Extended Ritz Method (ERIM)*

Besides the infinite dimensions of the spaces to which the admissible decision functions belong, a major feature of the problems addressed in the book consists in the large number of variables. This is typical in the previously mentioned applications. In such cases, often one incurs the so-called *curse of dimensionality* [15], where the term “dimensionality” is referred to the number of variables the decision functions depend on (not to be confused with the infinite dimension of the problem, which we have previously discussed). The curse of dimensionality shows up as a very fast growth (typically, an exponential growth) of the computational effort required to obtain accurate suboptimal solutions, when the number of variables increases but the desired accuracy is kept fixed. Developing solution methodologies and effective algorithms to face this issue in IDO problems is, to a large extent, an open problem.

Since, as previously mentioned, deriving analytically an optimal solution \boldsymbol{y}° to an IDO problem is possible only in very few cases, one generally has to devise methodologies of approximate solution.

Remark 1.2 Of course, whenever one is interested in approximating something, one has to specify how the approximation error is measured. As we shall see, this will endow the infinite-dimensional space \mathcal{H} with a norm. This issue will be anticipated in Sect. 1.3.2 and addressed in detail in Chap. 2. In this introductory chapter, we shall be less formal, just supposing that one has not yet decided how to measure the accuracy of approximation. \triangleleft

The main methodology described in the book lies in constraining the decision functions \boldsymbol{y} to take on a fixed structure, where a finite very large (if necessary) number of “free” parameters is inserted. We call such functions *fixed-structure parametrized (FSP) functions*. By substituting them into the original cost functional and into the constraints, one obtains an NLP problem. The solution of the latter is then entrusted

to an adequate NLP algorithm for the search of minimum points. The choice of the FSP functions is practically boundless. Particularly simple ones result from linear combinations of a number n of “fixed” basis functions, i.e., linear combinations, with coefficients c_1, \dots, c_n , of basis functions $\varphi_1, \dots, \varphi_n$ that span a linear subspace (e.g., polynomial expansions) of the space in which the IDO problem is set. In the case of scalar decision functions (without any loss of generality), we have

$$\sum_{i=1}^n c_i \varphi_i(\cdot), \quad (1.1)$$

where the coefficients c_1, \dots, c_n are the parameters to be optimized.

When the functional to be minimized is one that appears in the calculus of variations (e.g., an integral functional), the choice of the linear combination (1.1) leads to the classical *Ritz method*, which dates back to 1909 [62].

Remark 1.3 The Ritz method is an instance of the so-called *direct methods* in the calculus of variations (see, e.g., [27] and [35, Chap. 8]). Such methods prove the existence of minimizers in IDO problems, without resorting to differential equations that are naturally associated to the minimization problems themselves, e.g., the Euler–Lagrange equations. Direct methods provide an effective alternative to solving such equations (which is often a difficult task, especially when several variables – hence, partial differential equations – are involved, and may provide solutions only in a neighborhood of some point) and can be exploited to compute solutions according to various desired levels of accuracy. \triangleleft

It does not seem that the Ritz method has met, in more than a century, with important successes as regards problems whose decision functions depend on a large number of variables (see [32, 42, 68] and the references cited therein; for a method strictly related to the Ritz one, see also [14]). In our opinion, this may be ascribed to the fact that such a method can be subject to the following kind of curse of dimensionality.

Let $\varepsilon > 0$ be the maximum acceptable error in determining an approximation of an optimal solution. Then, it may occur that, for a fixed value of ε , the minimal number n of basis functions of the linear combination (1.1) (hence, the minimal number of coefficients of such combination), able to guarantee the approximation error ε , grows with d in an unacceptably fast way, e.g., at a rate of order⁴ $\Omega(1/\varepsilon^d)$. In the presence of decision functions that depend on a large number d of variables, the curse of dimensionality makes computationally intractable the NLP problems to which the IDO problems are reduced via parametrizations of the decision functions themselves. This is a severe drawback, since, as we have already pointed out, the situation in which the decision functions γ depend on a large number of variables often arises in applications. Theoretical results show the danger of incurring the curse of dimensionality using the Ritz method for specific instances of approximation and IDO problems (see, for example, [25, 61]).

⁴Rates of growth are defined in Remark 2.2.

Following the basic idea that allowed one to achieve very good and sometimes surprising results in the *theory of function approximation* (which we shall consider in the next section), we choose FSP functions made up of linear combinations of the kind (1.1) but inserting some “free” parameters into the basis functions. We call such FSP functions *one-hidden-layer networks* or *OHL networks*, for short. They take on the structure

$$\sum_{i=1}^n c_i \varphi(\cdot, \mathbf{k}_i), \quad (1.2)$$

where the coefficients c_1, \dots, c_n and the components of the vectors $\mathbf{k}_1, \dots, \mathbf{k}_n$ are the parameters to be optimized, once one has restricted the admissible functions γ of the IDO problems to take on the form (1.2).

Using (1.2) instead of (1.1) has led us to replace the term “Ritz method” with the *extended Ritz method* (ERIM). As will be shown in the book, even better results can be obtained by using FSP functions made up of “cascades” of OHL networks, called *multi-hidden-layer* (MHL) networks. Note that the term “network” is used by analogy with the structure of *multilayer feedforward neural networks* (MLFNNs), widely used in applications. It has to be remarked that very good results in the approximate solution of IDO problems via the ERIM have also been obtained by using FSP functions different from the above-described ones. In particular, in Chap. 3, we shall present the *kernel smoothing models*, besides the families of FSP functions considered till now. It is also worth remarking that the term “ERIM” is motivated only when one exploits OHL Networks (1.2) instead of (1.1) to approximately solve IDO problems. However, we shall refer to it whenever we use any kind of FSP function.

Despite its apparent simplicity, the methodology based on the ERIM contains many delicate and technical points that, on one hand, complicate its study and, on the other hand, may make it an efficient tool. Choosing the FSP functions that are best suited to get approximate solutions to IDO problems requires to investigate the approximating properties of various families of such functions. Such properties can be evaluated by considering the following three quantities:

1. The *model complexity* n of an FSP function. It is a positive integer such that the number of “free” parameters to be optimized in the FSP function can be expressed as a function $\mathcal{N}(n)$. A very simple example is given by (1.1), where n is the number of coefficients of the linear combination. One can also immediately see that the model complexity of OHL networks (1.2) is given by the number of basis functions parametrized by the vectors \mathbf{k}_i .
2. The *number d of variables* an FSP function depends on.
3. The *approximation error* ε , given by the distance between the optimal solution and the approximate one obtained by applying the ERIM.

Clearly, the quantities n , d , and ε are linked to one another. Let us detail this point a little. Once fixed the maximum acceptable approximation error ε , the FSP functions exhibiting the following property are particularly interesting. Given ε , let us consider (if it exists) the optimal solution $\gamma_n^{d\circ}$ belonging to a certain class of FSP functions with model complexity n (in the notation, we have emphasized its dependence on

the model complexity n and the number d of variables but not on ε , as the latter is fixed). Intuitively, if d grows, then the number $\mathcal{N}(n)$ of parameters to be optimized has to grow too for the same value of ε . Now, for a given ε , if the rate of growth of $\mathcal{N}(n)$ with respect to d is at most polynomial, then we say that the family of FSP functions $\gamma_n^{d_0}$ (we use the term “family” as n and d can get any value) is a family of *polynomially complex optimizing FSP functions*. Examples, in suitable contexts, are certain sigmoidal neural networks, some free-knot splines, and other families of functions that we shall mention later on. The polynomially complex optimizing FSP functions contrast with other FSP functions in which, in order to have an error at most ε , the number of parameters to be optimized grows faster than polynomially (typically exponentially) with d , thus incurring the curse of dimensionality. Clearly, such FSP functions are unsuitable for approximating optimal decision functions in high-dimensional settings.

A quantity of successful results in several application domains has induced us to express the conjecture that the property of the “moderate” increase of the number of parameters to be optimized as the number d of variables grows also holds in the area of approximate IDO (this property has been demonstrated in the theory of *function approximation*). Limiting ourselves to report only our most significant numerical results (numerical and application results in the literature are countless), we refer the reader to [22, 31, 34, 36, 57–60, 76] concerning optimal control problems; we also mention [2–8] in the area of optimal filtering and parameter estimation, and [11, 12, 37] as far as distributed-information control and routing in networks are concerned. A large part of the statements reported in the book has taken the hint from such numerical evidences and the abovementioned conjecture. Recent results, discussed in Chaps. 2 and 3, have shown that, under suitable assumptions, the conjecture is true. However, further research is required in this direction. One has to remark that theoretical results of the same extent as those obtained in the area of function approximation are not yet available for the ERIM. This is not surprising as the problem of minimizing a cost functional is generally much more difficult than the problem of function approximation. Indeed, in IDO the cost functional F is usually much more complex than a suitably defined “distance”, typically induced by a norm, by which one estimates the error between a given function belonging to an infinite-dimensional space and an FSP function.

1.1.3 Approximation of Functions

As pointed out in the previous section, the theory of function approximation is much more developed than the theory of approximate solution of IDO problems by FSP functions. Since many numerical results and some theoretical ones emphasize the existence of deep connections between the two areas, understanding whether some basic properties of approximation theory can be “transferred” to the area of approximate solution of IDO problems is of great importance.

In very simple terms, the *function approximation problem* can be described as follows. A given function belonging to an infinite-dimensional space has to be approximated by an FSP function, in such a way to minimize an error (for instance, a quadratic distance between the function to be approximated and the approximating FSP function). Often, the error is measured in the norm with which the infinite-dimensional space has been endowed. For simplicity, in the following discussion, we ignore the fact that the function to be approximated is typically known only pointwise, maybe with values corrupted by noises, so giving rise to an *estimation error*, too. Then, we consider only the *approximation error*, which is caused by the fact that, in general, the function to be approximated cannot be represented (i.e., expressed exactly) by an FSP function of the chosen type and/or the chosen structure of FSP functions does not contain a sufficient number of free parameters to be optimized. The approximation and the estimation error will be addressed in detail in Chap. 4.

Let us now try to explain the reason why the function approximation problem by FSP functions has been introduced. The methodology proposed in this book has been originated in correspondence of the spread and the successful performances of the previously mentioned artificial intelligence paradigm known as MLFNN. Artificial neural networks have a long history, to which many researchers contributed. We refer the interested reader, for instance, to the work of Barto (see, in particular, his “personal view of the history of connectionistic research” [13, Fig. 1]) and to the section “Historical notes and terminology” in [67]. In such a history, a strong renaissance of artificial neural networks took place in the second half of the ’80s of the last century, essentially by Rumelhart et al. [63, 64]. Among other contributions, they developed the ideas contained in the works by Werbos, mainly originated from his Ph.D. dissertation [73].

From a mathematical point of view, MLFNNs are FSP functions, whose particular structure can be exploited to solve the function approximation problem. Such a structure allows one to use the *backpropagation method*, independently developed by various researchers (see, e.g., [63, 64, 74]).

OHL neural networks, and, more generally, MLFNNs were used (i.e., optimized) with success – often surprising success – to solve applicative problems that can be brought back to function approximation. This unstoppable success has continued over the years, passing through the “deep belief networks” and arriving till sort of “deep learning revolution.” We refer the reader, e.g., to the seminal papers [44, 45] and the recent reference book [38]. Such successful performances drew the attention of mathematicians, computer scientists, statisticians, etc., who started a theoretical investigation of the approximation capabilities of OHL networks with the structure (1.2). Among the first networks examined, we cite sigmoidal neural networks, radial basis function (RBF) networks with variable centers and widths, trigonometric polynomials with free frequencies, spline functions with free knots, etc. For such OHL networks, the possibility was proved of solving the function approximation problem satisfying the following basic property (expressed here in rather rough terms; we shall be more formal in Chap. 2):

Given a fixed maximum value $\bar{\varepsilon}$ of the quadratic approximation error, let $\tilde{\gamma}_n^d$ be an OHL network that approximates the function γ^d with an error not larger than $\bar{\varepsilon}$ (we have emphasized in γ^d the number d of variables, and in $\tilde{\gamma}_n^d$ also the model complexity n). Suppose that the function γ^d to be approximated is provided with suitable regularity requirements. Then, in order that the OHL network keeps the approximation error bounded from above by $\bar{\varepsilon}$, the number $\mathcal{N}(n)$ of parameters grows with d at most polynomially.

Note that in (1.2) the basis functions $\varphi(\cdot, \mathbf{k}_1), \dots, \varphi(\cdot, \mathbf{k}_n)$ are obtained from a “mother function” φ dependent on vectors $\mathbf{k}_1, \dots, \mathbf{k}_n$ of adjustable parameters, which have to be optimized together with the coefficients c_1, \dots, c_n . In general, the presence of the “inner” parameter vectors $\mathbf{k}_1, \dots, \mathbf{k}_n$ “destroys” linearity and (1.2) belongs to the family of the so-called *variable-basis approximators* [48, 49, 51–53].

Beyond the above specific cases, we say that the family of FSP functions, equipped with the above property, is a family of *polynomially complex approximating FSP functions*. The behavior of such FSP functions allows one to mitigate the curse of dimensionality; we address this basic issue in Chap. 3.

1.1.4 From Function Approximation to Approximate Infinite-Dimensional Optimization

On the basis of what we discussed in the last two sections, one can conclude that the way of proceeding followed by the ERIM to solve IDO problems approximately, but with an arbitrarily small error, should be articulated into the following three steps (see Fig. 1.2 for a better understanding of the sequence of steps): (1) identifying a

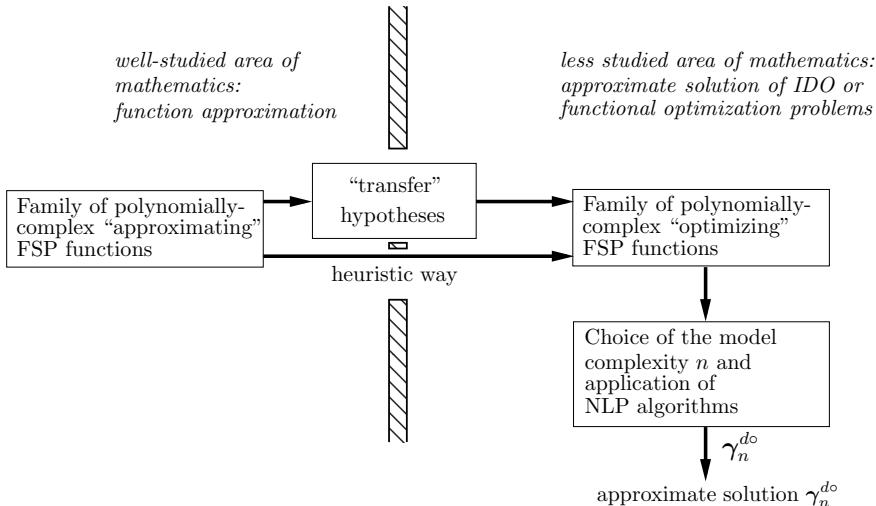


Fig. 1.2 The polynomially complex “approximating” functions “become” polynomially complex “optimizing” functions

family of polynomially complex approximating FSP functions. (2) Verifying that suitable “*transfer hypotheses*” can be satisfied. Such transfer hypotheses allow one to transform the families of polynomially complex approximating FSP functions into corresponding families of polynomially complex optimizing FSP functions. At present, unfortunately, these hypotheses are rather restrictive; even ascertaining whether the chosen FSP functions verify them may be a very difficult task. This important aspect of the ERIM deserves further research efforts. Clearly, if the satisfaction of the hypotheses cannot be carried out, one may skip this step and just proceed heuristically. Indeed, starting with families of polynomially complex approximating FSP functions is encouraging anyway. So, in this case, the step consists simply in considering the family mentioned in step (1) as a family of polynomially complex optimizing FSP functions. (3) Determining the model complexity n of the FSP function on the basis of available information and/or by a trial-and-error approach and finding the approximate solution to the original IDO problem by means of NLP algorithms.

1.1.5 Relationships with Parametrized Control Approaches

Once the adjective “neural” in the title of the book has been motivated, and keeping in mind that actually the FSP functions that we consider might have nothing to do with neural networks, we point out that the expression “parametrized” instead of “neural” may be more appropriate. Indeed, it is contained in the words “fixed-structure parametrized function.” There are important research precedents that are worth recalling.

Optimizing some parameters for a closed-loop control law of preassigned form dates back to so-called *specific optimal control* (see, for instance, [65] and the references cited therein). Similarly, in LQG optimal control problems, the designer may wish to simplify the control law by constraining it to be a linear function of the output vector instead of a reconstructed state vector. Thus, he has to derive the matrix gain of a suboptimal controller. Analogous simplifications may be sought in determining fixed-structure low-order controllers and in solving decentralized optimal control problems. A survey of the computational methods for deriving the various *parametric* controllers (e.g., the Levine–Athans and Anderson–Moore methods) is reported in [55]. More recently – driven by the seminal paper [17], in which a finite- and infinite-horizon optimal control law for constrained linear system has been given – several new approaches have been proposed to address the approximate offline determination of optimal control laws to be exploited in the *model predictive control* (MPC) framework (the so-called “explicit MPC”). Specifically, the problem of obtaining explicit MPC controllers with quadratic cost and linear constraints for linear systems can be solved by using parametric quadratic programming techniques (see, for instance, [47] and, again, [17]). The multiparametric optimization approach has also been used to obtain robust explicit feedback laws for uncertain linear systems (see [16, 26, 29, 46, 56] and the references therein for an overview of the

subject). As regards nonlinear systems, the explicit solution of MPC problems has more recently attracted a lot of research efforts (see, for instance, [40, 50]; see also the very interesting survey [9] and the references cited therein). We also cite the recent monograph [66] on the approximate solution of stochastic control problems via quantization.

In spite of the analogy to such parametric approaches in control theory, the purpose of the method proposed in the book is substantially different. We assign a given structure to the control law or to the state estimator not to obtain a simplified sub-optimal solution but just because we are not able to derive an optimal solution in an analytical form. So, our approach turns out to be closer to the Ritz method for optimal control. However, even more recent applications of such a method (see, e.g., [14]) are strongly limited by the curse of dimensionality. By contrast, our approach may reveal itself to be determinant in mitigating this danger.

1.2 Contents and Structure of the Book

In simplified terms, the book can be divided into two parts: the first five chapters describe IDO, the ERIM, and several computational tools. They report quite theoretical topics, which lay the bases to make the method both mathematically well-sounded and operative. The other five chapters describe the application of the method. There are strict connections between the first part and the second one. Roughly speaking, the first part does not consider the variable “time” and presents the basic topics in a form that we may call “static”. In the second part, instead, the applicative problems are formulated by introducing the variable “time” or, better to say, the decision stages. Indeed, as already said, we address discrete-time dynamic systems. Sometimes, passing from the static case to the multistage problems requires a not trivial deepening. The interconnections among the ten chapters are schematically shown in Fig. 1.3.

In the remaining of this chapter, we state the basic infinite-dimensional optimization or functional optimization problem in a formal way. We call it *Problem P*. We extend this problem to the case where there are M decision-makers acting one after the other in the already mentioned M -stage optimal control problem. The M decision-makers can also generate decisions in the same instant as members of a cooperative organization (“team theory”). We call this extension of Problem P *Problem PM*. Then, we present a certain number of IDO problems to point out the variety of problems that the ERIM can face.

In Chap. 2, the scheme of Fig. 1.2 is detailed and the various families of FSP functions are formally defined. In the area of the well-consolidated function approximation theory, we provide the definition of (1) the approximating sequences of sets that enjoy the property of density in the sets of functions that one wants to approximate; (2) the polynomially complex approximating sequences of sets that benefit from the property of being able to approximate, with arbitrary accuracy, functions provided with suitable regularity properties. Such sequences of sets use, for a desired approximation accuracy $\varepsilon > 0$, a number $\mathcal{N}(n)$ of “free” parameters that increases

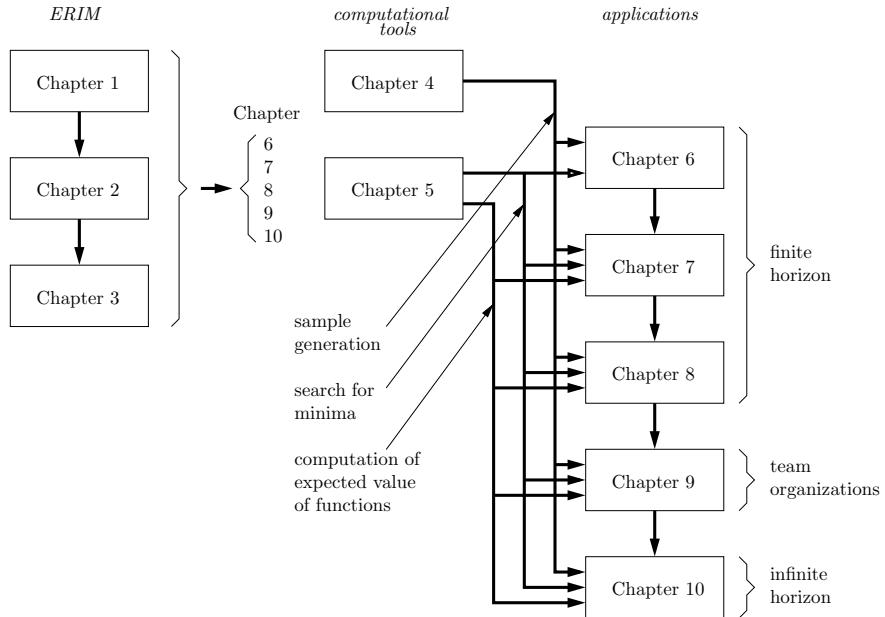


Fig. 1.3 Scheme of the ten chapters of the book

at most polynomially when the number d of variables, on which the functions to be approximated depend, grows. This should mitigate the danger of the curse of dimensionality. With a little abuse of terminology, we define as “approximating functions” and “polynomially complex approximating functions” the FSP functions belonging to the sets dealt with at points (1) and (2), respectively.

It is important to notice that in the book we want to contrast various types of the curse of dimensionality, “that always waits in ambush.” Even if in different contexts, the curse of dimensionality has always the expressive meaning of the term introduced by Bellman (see, e.g., [15]).

In the much less studied area of the approximate solution of IDO or functional optimization problems, at first we define the optimizing sequences of FSP functions, and then the polynomially complex optimizing sequences of FSP functions. Here, we arrive at a key point of the book. Still referring to the scheme of Fig. 1.2, we present some results on the basis of which we can assert that, if appropriate hypotheses occur, polynomially complex approximating sequences of sets give rise to polynomially complex optimizing sequences of FSP functions. Such sequences should provide approximate solutions to Problem P possibly mitigating the curse of dimensionality.

Chapter 3 reports examples of the most widespread families of FSP functions which enjoy the property of being able to build polynomially complex approximating sequences of sets. Moreover, the chapter is focused on the deepening of the approximating properties of the OHL networks, with particular reference to the OHL neural networks. Finally, the chapter presents a theoretical comparison between the ERIM

and the classical Ritz Method and estimates the accuracy of suboptimal solutions to IDO problems via the ERIM. The chapter is particularly important for the reader interested in the mathematical aspects of the said properties. The reader who does not share such an interest can avoid reading the chapter without losing the central thread along which the ERIM unravels.

Chapter 4 deals with topics which are treated in the literature, but which are useful for the following chapters. The chapter begins with well-known concepts in statistical learning theory. In reference to the problem of the modeling of input-output (I/O) relationships by FSP functions, we introduce the concepts of expected risk, empirical risk, and generalization error. This last error is then decomposed into approximation and estimation errors. Besides the accuracy ε , the number d of variables, and the model complexity n (the relationship among such quantities is dealt with in the previous chapters), a fourth variable is emphasized. Such a variable is given by the number L of samples and is obviously important in the estimation procedures. About this variable, we point out the importance of the generation of samples by means of deterministic algorithms. Therefore, the so-called “*quasi-Monte Carlo*” (quasi-MC) approach is placed alongside the traditional generation of samples on the basis of probability density functions (the well-known “*Monte Carlo*” (MC) approach).

The former approach led to introduce the so-called *deterministic learning theory* (DLT) explicitly, on which the second part of the chapter focuses. The advantages of the quasi-Monte Carlo techniques are basically two: the possibility of reducing the number of generated samples (under the same accuracy) and the possibility of obtaining upper bounds on the errors in deterministic terms rather than in probabilistic ones. Therefore, the contents of Chap. 4 result very important in Chaps. 6 and 7, where we shall try to mitigate the curse of dimensionality deriving from the use of regular grids in the calculation of the optimal cost-to-go functions required by the application of dynamic programming (see the scheme in Fig. 1.3).

Also Chap. 5 presents topics treated in the literature. They are fundamentally two. The first addresses the numerical computation of expected values of functions depending on random variables. Such calculation implies the computation of integrals of functions that may depend on a large number of random variables. Then, the use of regular grids implies the risk of the curse of dimensionality. Therefore, MC and quasi-MC techniques are taken into consideration in order to reduce it. The second argument considers the solution of NLP problems – obtained from the approximation of the IDO ones – by the ERIM (it becomes useful starting from Chap. 6). Three techniques are considered: (i) discretization of the region of admissibility by Monte Carlo and quasi-Monte Carlo methods, (ii) direct search and gradient-based descent algorithms, and (iii) stochastic approximation, that enables one to avoid the computation of the expected values of the costs and to adapt the optimal solution online whenever the characteristics of the random variables change in an unforeseeable way. This third technique is by far the most used throughout the book.

As most of the instances of IDO problems addressed in the book belong to the field of optimal control, we consider the application of the ERIM to such a context in more detail. In optimal control problems, the variables of the decision functions are (besides the time) the information that becomes available to the DM or the controller

only online. If no random variable acts on the dynamic system, we are in the area of deterministic optimal control. Then, an optimal solution takes on the form of an open-loop control structure. This class of problems, defined as Problems C1, are addressed in Chap. 6 under the assumption that the control process has a finite horizon (FH). Almost all problems that are dealt with in the book are not solvable in a closed form (typically, the classical LQG assumption are not verified). Then, in T -stage optimal control problems, we put the powerful instrument of the *dynamic programming* (DP) alongside the ERIM. Both methodologies allow us to obtain accurate approximate solutions numerically.

It is important to observe that Problem C1 is not an IDO problem but an FDO one. As a matter of fact, the admissible solutions are constituted by T controls and T state vectors. Therefore, Problem C1 is already in the form of an NLP problem. Problem C1 is well suited to be solved by DP. The chapter is devoted to Bellman's principle of optimality and to the solution mechanism of DP. Consequently, it can be omitted by the reader who already masters this methodology. Nevertheless, the procedures of sampling the state space and the approximate calculation of the optimal cost-to-go and control functions in the backward phase of DP may result interesting. This approximation is obtained by introducing FSP approximating functions at every stage and by solving two "small" IDO problems by the ERIM, one for the costs and one for the controls. The sampling is done by means of quasi-MC techniques. The overall procedure can be considered as a sort of "neuro-dynamic programming" [20] based on the DLT. On the basis of the same theory, we also give bounds on the estimation error.

From Chap. 7 onward, we enter the area of optimization problems influenced by random variables. As a consequence, such problems must be generally formulated as IDO problems, for which the ERIM becomes a solution instrument of great importance. In Chap. 7, we suppose that (i) there are disturbances acting on the dynamic system, and (ii) the state can be measured perfectly. This second hypothesis is fundamental since the DMs' control functions depend only on the currently measured state, and both DP and the ERIM can be applied without any particular conceptual difficulties. The T -stage stochastic optimal control problem, addressed in the chapter, is defined as Problem C2.

The first part of Chap. 7 considers the numerical solution of the "backward phase" of DP and follows the lines exposed in Chap. 6. Of course, now we need to calculate the optimal cost-to-go functions performing in addition an averaging operation over the disturbances. Always in order to reduce the danger of the curse of dimensionality, the averaging operation is executed by MC or quasi-MC techniques, and the bounds on the estimation errors are presented on the basis of the concepts exposed in Chap. 5. In the second part of the chapter, Problem C2 is solved by means of the ERIM. More precisely, once the IDO problem has been transformed into an FDO problem represented by an NLP problem, the gradient of the cost with respect to the parameters of the chosen FSP function is calculated. The recursive equation, which is obtained to determine the gradient of the cost, is the classical adjoint equation for the T -stage deterministic optimal control problem (see Problem C1) with the addition of a term that takes into account the introduction of an FSP function as control function. This

additional term is calculated in the case the chosen FSP function is a multilayer feedforward neural network.

Among some applicative problems, we use an example on the optimal management of a network of water reservoirs to compare DP with the ERIM. Obviously, we can only draw widely indicative suggestions. Nevertheless, in a T -stage decision process, where DP expresses its utmost efficiency, we can say that the ERIM does not cut a poor figure. In this regard, it is important to point out that, in the set of IDO problems, the subset where the DP can be applied is somewhat small. Indeed, the optimization problems belonging to this subset require, as preliminary condition, to be structured in sequences of independent decision stages. Such a condition occurs rarely in most IDO problems, like in parameter estimation and system identification, design of receivers in the telecommunications field, team control problems, etc. Instead, the ERIM can be applied with the same ease also in these cases.

Whereas in Problem C2 all the state components are assumed to be perfectly measured by the controller, in Problem C3, addressed in Chap. 8, we give up this assumption, which is not actually always realistic. Then, it is necessary for the controller (or, equivalently, the decision-makers $DM_0, DM_1, \dots, DM_{T-1}$) to retain in memory the “history” of the dynamic system, i.e., all the measures acquired up to the stage t and all the controls generated up to stage $t - 1$. We stack all these vectors into a single column vector \mathbf{I}_t that we call “information vector.” Note that the dimension of \mathbf{I}_t increases linearly with t . Clearly, the memory of the controller must be “perfect”. Equivalently, every DM_t must “deliver” the information vector to its successor DM_{t+1} . As is well known, should the LQG hypotheses be verified, the optimal control functions would be given by a negative feedback constituted by the product of a time-varying gain matrix and the state estimate provided by the Kalman filter.

Since all the problems that we consider in the book are of general type, also in the case of Problem C3 it is an almost impossible task to find an optimal solution in closed form, apart from very simple problems. As regards DP, we recall that the difficulties are substantially two: (i) the increase of $\dim(\mathbf{I}_t)$ with time and (ii) the computationally too onerous determination of the conditional probability density function of the state. Such a density is necessary to perform the averages in the backward phase of DP. What has been said above falls within the classical concepts of T -stage stochastic optimal control and is reported for the reader’s convenience. For more recent research directions – of particular interest for the case of the infinite-horizon framework – we refer to good surveys like [18], where approximate suboptimal schemes (e.g., rollout, open-loop feedback control, the previously mentioned MPC, etc.) are presented.

Differently from DP, the ERIM can be applied as usual, at least in part. The relationships for the calculation of the gradient of the cost are computed for generic FSP control functions. We get to the classic adjoint equation of T -stage optimal control with the addition of a term that takes into account the introduction of the FSP control functions. This term is specialized in case that the FSP functions are given by multilayer feedforward neural networks. Of course, as is intuitive, the calculation of the gradient also becomes onerous when the dimension of the vector \mathbf{I}_t becomes very large, i.e., when the number T of stages is very high. Then, we propose the

following approximation. To truncate \mathbf{I}_t and to keep only the “most recent past” by “forgetting”, at stage t , the measures that have been acquired and the controls that have been generated before stage $t - M + 1$ (possibly weighted by a coefficient decreasing to 0 with the oldest data) would be one of the most natural approximations. Therefore, a “limited-memory information vector” \mathbf{I}_t^M is kept in memory. The abrupt truncation of \mathbf{I}_t may be mitigated by introducing a fixed-dimensional vector s_t by means of which the controller can maintain some form of memory of the measures and the controls that it has put aside at stage $t - M$. We have thought it right to introduce a suitable dynamic system in order to generate s_t . The vectors \mathbf{I}_t^M and s_{t-M} become arguments of the control function. We have not yet addressed its theoretical issues but this suboptimal control scheme yielded nice numerical results and performed better than the one based only on the truncated information vector \mathbf{I}_t^M .

Chapter 9 deals with the case in which several DMs share different information in the same instants and take decisions aimed at minimizing a common cost functional. This organization can be mathematically described within the framework of “team theory.” Differently from Problems C1, C2, and C3, the LQG hypotheses are sufficient neither to obtain an optimal solution in closed form nor to understand whether an optimal solution exists. In order to obtain an optimal solution in closed form, we must introduce suitable assumptions on the “information structure” of the team in addition to the LQG hypotheses. The “information structure” describes the way in which each DM’s information vector is influenced by the stochastic environment and by the decisions taken by the other DMs. In general, DP cannot be applied unless the information structure takes particular forms. On the contrary, we can always apply the ERIM. This is one of the families of IDO problems in which the ERIM turns out to be superior to the DP. Basic concepts of team theory are recalled in the first part of the chapter. Then, the ERIM is tested in two case studies. The former is the well-known Witsenhausen counterexample. The latter is an optimal routing problem in a store-and-forward packet switching network, in which the routing nodes play the role of cooperating DMs of a team. The LQG hypotheses are not verified and the information structure is quite complex.

In Chap. 10, optimal controls have to be generated and applied over an infinite number of decision stages, that is, over an infinite horizon (IH). In the first part of the chapter, Problems C1, C2, and C3 are restated over an IH. As far as Problem C1-IH is concerned, both the open-loop and the closed-loop formulations are given and conditions for the existence of a stationary IH optimal control law are provided. The formulations of Problems C2-IH and C3-IH are also provided for the sake of completeness but the IH stochastic case is not dealt any further in the chapter. Unless strong assumptions are made on the dynamic system and the random variables (if present), the design of the optimal IH controllers is an almost impossible task. Then, we resort to the well-known approximation of the “receding horizon” (RH) control scheme and restate Problem C1-IH accordingly. In the second part of the chapter, with reference to the deterministic case, we deal with the fundamental issue of closed-loop stability that arises owing to the infinite number of decision stages. More specifically, we address the stability properties of the closed-loop deterministic system under the action of approximate RH control laws obtained by the ERIM.

and implemented through FSP functions. Conditions are established on the maximum allowable approximation errors so as to ensure the boundedness of the state trajectories.

1.3 Infinite-Dimensional Optimization

After general considerations on the objectives of the book and its contents, the time has come to formally state the infinite-dimensional optimization Problem P. The frameworks of infinite- and finite-dimensional optimization are also compared in some detail.

1.3.1 Statement of the Problem

An *optimization problem* deals with a set \mathcal{S} whose elements are called *admissible* (or *feasible*) *solutions* of the problem (i.e., alternatives at a DM's disposal) and with a *cost functional*

$$F : \mathcal{S} \rightarrow \mathbb{R}. \quad (1.3)$$

The aim is to find a feasible solution that minimizes the cost functional. Clearly, this very general concept of an optimization problem can encompass almost all the situations we can imagine.

To address the optimization problems we are interested in, we require that \mathcal{S} be a subset of an infinite-dimensional real linear space \mathcal{H} of functions

$$\gamma : D \rightarrow C, \quad (1.4)$$

where

$$D \subseteq \mathbb{R}^d, \quad d \geq 1, \quad C \subseteq \mathbb{R}^m, \quad m \geq 1.$$

$\mathcal{S} \subseteq \mathcal{H}$ is then the set of admissible functions in that it specifies constraints on such functions. Of course, if there are no constraints, then $\mathcal{S} = \mathcal{H}$.

In the optimization problems dealt with in the book, the functions γ will be called *decision functions* or *control functions*. They generate *decision* or *control* vectors belonging to C on the basis of *information* vectors belonging to D .

Throughout the book, for the sake of simplicity, we shall usually assume that a minimizing function exists for the finite- or infinite-dimensional optimization problem at hand. In Sect. 1.4.1 we shall discuss in detail this issue. However, it is convenient to anticipate right now the hypothesis that *minimizing functions do exist* (see Assumption 1.1, where the uniqueness of the optimal solution is also required). So, the *IDO* or *functional optimization problem* can be stated as follows.

Problem P. Let \mathcal{H} be an infinite-dimensional real linear space and $\mathcal{S} \subseteq \mathcal{H}$. Find an optimal decision function $\boldsymbol{\gamma}^\circ$ that minimizes over \mathcal{S} the cost functional F , i.e.,

$$\text{find } \boldsymbol{\gamma}^\circ \in \underset{\boldsymbol{\gamma} \in \mathcal{S}}{\operatorname{argmin}} F(\boldsymbol{\gamma}). \quad (1.5)$$

△

Note that we have not yet required that the infinite-dimensional space \mathcal{H} is a normed linear space. As already remarked, we shall need to endow it with a norm because we shall have to measure approximation errors and to investigate convergence properties. See Sect. 1.3.2 for some more technical comments about the infinite-dimensional feature of Problem P and about norm-related issues in infinite-dimensional spaces. Approximation errors will be dealt with for the first time in Sect. 2.4. See Sect. 1.4.2 for the notation adopted in this book for norms.

As said previously, in most cases one has to consider more than one unknown decision function. This occurs, for example, whenever one controller has to make decisions at successive instants in a stochastic T -stage optimal control problem (see Problem 1.4). In such a classical problem, we have to determine T optimal control functions by means of which the controller generates optimal controls at each decisional stage. Equivalently, we may say that T decision-makers $DM_0, DM_1, \dots, DM_{T-1}$ generate their own decisions sequentially one after the other.

Another interesting example is given by a team organization (see the description of Chap. 9 in the previous section), where M decision-makers DM_1, \dots, DM_M generate their decisions at the same time instant. They cooperate to the minimization of the same cost functional but their information patterns are different.

As the aforesaid IDO problems show and as is evident in many applications, it is necessary to extend Problem P in order to be able to optimize several decision functions. Such an extension requires some more formalisms but does not involve any conceptual difficulties.

Then, with reference to the notation used for Problem P, let us consider M decision functions

$$\boldsymbol{\gamma}_i : D_i \rightarrow C_i, \quad (1.6)$$

where

$$D_i \subseteq \mathbb{R}^{d_i}, \quad C_i \subseteq \mathbb{R}^{m_i}, \quad i = 1, \dots, M.$$

For each $i = 1, \dots, M$, the functions $\boldsymbol{\gamma}_i$ are the elements of infinite-dimensional real linear spaces \mathcal{H}_i . In general, the functions (1.6) may be jointly constrained to belong to an admissible set

$$\mathcal{S}_M \subseteq \mathcal{H}_1 \times \cdots \times \mathcal{H}_M. \quad (1.7)$$

We now define the function⁵

⁵To avoid burdening the notation and without risk of ambiguity, we shall use the same symbol $\boldsymbol{\gamma}$ as in Problem P, where a single decision function is optimized.

$$\boldsymbol{\gamma} \triangleq \text{col}(\boldsymbol{\gamma}_1, \dots, \boldsymbol{\gamma}_M) : D_1 \times \dots \times D_M \rightarrow C_1 \times \dots \times C_M, \quad (1.8)$$

with $\boldsymbol{\gamma} \in \mathcal{S}_M$. Let

$$F : \mathcal{S}_M \rightarrow \mathbb{R}$$

be a cost functional defined on \mathcal{S}_M . Then, we can state as follows the extension of Problem P to a number M of decision functions.

Problem PM. *Let $\mathcal{H}_1, \dots, \mathcal{H}_M$ be infinite-dimensional real linear spaces and $\mathcal{S}_M \subseteq \mathcal{H}_1 \times \dots \times \mathcal{H}_M$. Find an optimal collection $\boldsymbol{\gamma}^\circ = \text{col}(\boldsymbol{\gamma}_1^\circ, \dots, \boldsymbol{\gamma}_M^\circ)$ of decision functions that minimizes over \mathcal{S}_M the cost functional F , i.e.,*

$$\text{find } \boldsymbol{\gamma}^\circ \in \underset{\boldsymbol{\gamma} \in \mathcal{S}_M}{\operatorname{argmin}} F(\boldsymbol{\gamma}).$$

△

Problem PM is formally identical to Problem P and all the theoretical properties which we shall consider for Problem P can be extended directly to Problem PM. However, as the number M of decision functions will play an important role throughout the book, we shall keep a distinction between Problem P and Problem PM. In practice, not to get unnecessarily burdened with heavy notation, the theoretical facts regarding infinite-dimensional optimization will be established by addressing Problem P. Instead, application problems (like optimal control problems, optimal estimation problems, etc.) will be modeled and examined (whenever $M > 1$) in terms of Problem PM.

In Sect. 1.5, we shall present some examples of IDO problems. Some of them will be addressed in detail throughout the book.

Most IDO problems considered in the book are stated within stochastic frameworks. It is important to anticipate that often we shall have to solve functional optimization problems in which the cost functional F is expressed as the average of another functional J dependent on a random vector \mathbf{z} and M decision functions $\boldsymbol{\gamma}_1, \dots, \boldsymbol{\gamma}_M$ to be optimized, i.e.,

$$F(\boldsymbol{\gamma}) = \underset{\mathbf{z}}{\mathbb{E}} J(\boldsymbol{\gamma}, \mathbf{z}), \quad (1.9)$$

where we have used the compact notation (1.8). In some problems presented in Sect. 1.5, we shall highlight that the cost functionals may take the form of the right-hand side of (1.9). In Sect. 2.3 and especially in Chap. 5, we shall see that to keep explicit the operation of averaging may be convenient to solve these problems. Unless otherwise specified, we shall assume that the probability density function $p(\mathbf{z})$ exists and is known; see Assumption 1.2 in Sect. 1.4.2 for the formal statement of this assumption.

1.3.2 Finite-Dimensional Versus Infinite-Dimensional Optimization

The procedure of approximate solution of Problem P by the ERIM has to be considered in the light of the fact that the spaces \mathcal{H} , \mathcal{H}_1 , ..., \mathcal{H}_M in the definitions of Problems P and PM are infinite-dimensional. It is worth remarking some basic differences between optimization in infinite-dimensional spaces and optimization in finite-dimensional spaces. First of all, let us recall some dimension-related notions.

As is well known, if, for every $n \in \mathbb{Z}^+$, there exist n linearly independent elements in \mathcal{H} , then we say that \mathcal{H} is of *infinite dimension*. If \mathcal{H} is not infinite-dimensional, then there exists $n \in \mathbb{Z}^+$ such that in \mathcal{H} there are n linearly independent elements but, for every $N > n$, any N elements in \mathcal{H} are linearly dependent. Such a value n is called the *dimension* of \mathcal{H} , which is said to be *finite-dimensional*. Any set of n linearly independent elements is called a *basis* for such a space \mathcal{H} .

In applications, two cases may occur about the set \mathcal{S} of admissible solutions:

1. There exists a linear n -dimensional subspace \mathcal{H}_n of \mathcal{H} such that $\mathcal{S} \subseteq \mathcal{H}_n$ (typically, \mathcal{S} is not itself a linear space, as happens, for example, in the presence of constraints on the functions belonging to \mathcal{S}).
2. There exists no finite-dimensional subspace of \mathcal{H} containing \mathcal{S} .

In Case 1, Problem P can be restated in the finite-dimensional subspace \mathcal{H}_n , so the infinite-dimensional nature is not essential. Once a basis for \mathcal{H}_n has been fixed, every admissible solution can be expressed as a linear combination of such basis functions. Thus, in this case, Problem P consists in finding the values of the coefficients of the linear combination that minimize a function of n variables (i.e., the coefficients). This is a finite-dimensional optimization problem (i.e., a constrained or unconstrained NLP problem in \mathbb{R}^n) that can be solved, at least in principle, by some suitable NLP algorithm.

On the basis of the discussion above, it is clear that the only case in which the infinite-dimensional nature of Problem P is substantial is the Case 2. In such a context, it is natural to assume that \mathcal{H} is the “smallest” infinite-dimensional linear space containing \mathcal{S} .

Example 1.1 Consider the set

$$\begin{aligned} \mathcal{S} \triangleq & \{ \text{polynomials } \text{pol}^p(x) \text{ defined over } [0, 1], \\ & \text{with degree higher than 1 and at most } p \\ & \text{and satisfying the boundary conditions } \text{pol}^p(0) = \text{pol}^p(1) = 0 \}. \end{aligned}$$

Clearly, \mathcal{S} can be thought of as a subset of an infinite-dimensional space of functions but, obviously, it is also a subset of the finite-dimensional space of all polynomials defined over $[0, 1]$, with degree at most p . \triangleleft

In situations like the one considered in Example 1.1, the infinite-dimensional feature is not substantial; hence, one can remove it and turn to a finite-dimensional context.

Note that, in Case 1, one might not exploit the original characterization in finite dimension of the space \mathcal{H}_n in terms of a basis of cardinality n . This might happen, for example, in the following two situations:

- The basis that makes Problem P finite-dimensional consists of “complicated” functions and one is interested in finding suboptimal solutions that can be expressed in terms of easier basis functions, either fixed or parametrized.
- Available expressions of optimal or suboptimal solutions in terms of the original fixed basis or in terms of another fixed basis consist of a large number of terms, e.g., they exhibit the curse of dimensionality. In this case, one might search for suboptimal solutions expressible as linear combinations of a smaller number of parametrized basis functions.

In the two abovementioned situations, searching for solutions expressed as linear combinations of functions from a fixed basis different from the original one, or as linear combinations of functions from a parametrized basis, can still be considered as an application of the Ritz method or the ERIM, respectively (at least formally). However, as in these contexts there is no infinite-dimensional nature, we are not interested in them.

Once the infinite-dimensional nature of an instance of Problem P has been established, let us discuss an essential difference between FDO and IDO problems; this difference (in general) makes the latter problems intrinsically more difficult to solve. Some technical statements and examples on the differences between finite and infinite dimension in optimization are reported in [23, Sect. 9]. From a conceptual point of view, the main source of difficulty is the fact that in infinite-dimensional spaces compactness of sets is “easily lost,” i.e., a closed and bounded set is not necessarily compact. In other words, in infinite dimension being compact is “much harder” than in finite dimension. Indeed, in finite-dimensional spaces (e.g., \mathbb{R}^d) a set is compact if and only if it is closed and bounded. This is no more the situation in infinite dimension. Specifically, let \mathcal{H} be endowed with a norm $\|\cdot\|$ and suppose that it is complete with respect to such a norm (i.e., it is a Banach space). Then, the closed ball $\{\gamma \in \mathcal{H} : \|\gamma\| \leq 1\}$ is not compact (see, e.g., [75, p. 274] and [30, Proposition 7, p. 30]). Sufficient conditions on compactness in infinite-dimensional spaces include, for example, the Ascoli–Arzelà Theorem [1, Theorem 1.33, p. 11] in spaces of continuous functions and the Rellich–Kondrachov Theorem [24, Theorem 9.16, p. 285] in spaces of Lebesgue-integrable functions.

To further clarify how infinite dimension can make things more difficult, let us consider the following example from [30].

Example 1.2 Let $F : \mathcal{H} \rightarrow \mathbb{R} \cup \{+\infty\}$ be a lower semicontinuous functional and the normed linear space \mathcal{H} be Banach. Suppose that the so-called “sublevel sets” of F , i.e., the sets $L_\alpha(F) \triangleq \{\gamma \in \mathcal{H} : F(\gamma) \leq \alpha\}$ are bounded for every $\alpha \in \mathbb{R}$. By definition of lower semicontinuity, the sublevel sets are also closed (see, e.g., [30,

Table 1.1 Some substantial differences between FDO \equiv mathematical programming and IDO \equiv infinite-dimensional programming \equiv functional optimization

Finite-dimensional optimization (FDO) \equiv mathematical programming	Infinite-dimensional optimization (IDO) \equiv infinite-dimensional programming \equiv functional optimization
Finite dimension	Infinite dimension
Every problem can be set in the space \mathbb{R}^d for a suitable d	Need for choosing the space where the problem is set
Admissible solutions are vectors	Admissible solutions are functions
All norms are equivalent	In general norms are not equivalent
Compactness is “easy to achieve”	Compactness is “difficult to achieve”

Proposition 3, p. 29]). Of course, there exists α_0 such that, for $\alpha \geq \alpha_0$, minimizing F over L_α is the same as minimizing F over the whole \mathcal{H} . If \mathcal{H} is finite-dimensional, then L_α is compact. As (both in finite and in infinite dimension) a lower semicontinuous function achieves its minimum over a compact set (see, e.g., [30, Proposition 4, p. 29]), in finite dimension the minimum is achieved at some $y^\circ \in \mathcal{H}$. However, in infinite dimension this may not be the case: under the above hypotheses, the sublevel sets L_α might not be compact and so the minimum might not be achieved. \triangleleft

Table 1.1 summarizes some substantial differences between finite- and infinite-dimensional optimization. In the table, we mention the choice of a norm as a key point. As already remarked, this will be addressed in Chap. 2.

1.4 General Conventions and Assumptions

We gather into this section some remarks on notation, conventions, and assumptions about quite general points, which will be supposed to hold throughout the book without any further specification. We present such remarks here because many of them will be useful in the examples presented in the next section. Other general remarks of the same kind will be introduced throughout the book when needed.

The mathematical background that is generally needed in order to get insight into the concepts presented in the book is limited to standard calculus, introductory probability theory, and matrix–vector algebra. Here and there in the book, elementary concepts and tools from functional analysis are used. The prerequisites include basic concepts of optimization and control theory, too.

1.4.1 Existence and Uniqueness of Minimizers

The statement of Problem P deserves some comments that we shall shortly point out. In the minimization of $F(\gamma)$, three situations may occur.

(i) A minimizing function γ° actually exists such that

$$F(\gamma^\circ) \leq F(\gamma), \quad \forall \gamma \in \mathcal{S}.$$

Then, Problem P has been stated correctly.

(ii) A minimizing function does not exist. However, we are interested in the *greatest lower bound* (i.e., in the *infimum*) of F over \mathcal{S} . In this case, the statement of Problem P should be changed as follows:

$$\text{find } \bar{F} \triangleq \inf_{\gamma \in \mathcal{S}} F(\gamma). \quad (1.10)$$

(iii) As in situation (ii), no minimizing function exists but we are interested in both the infimum \bar{F} and in functions γ for which $F(\gamma)$ is arbitrarily close to \bar{F} (we assume the set $\{F(\gamma) : \gamma \in \mathcal{S}\}$ to be bounded from below). Then, the following problem has to be solved:

Problem 1.1 For a given $\varepsilon > 0$, find a function $\gamma_\varepsilon^\circ \in \mathcal{S}$ such that

$$F(\gamma_\varepsilon^\circ) \leq \inf_{\gamma \in \mathcal{S}} F(\gamma) + \varepsilon.$$

△

The function γ_ε° in Problem 1.1 is called ε -minimizer of F over \mathcal{S} .

Although several theoretical results and algorithmic procedures that we shall deal with hold also in the presence of multiple minimizers and for ε -minimizers, for the sake of simplicity and to avoid complex and non-substantial discussions, throughout the book we shall usually assume that a minimizer *does exist and is unique*. This will allow us to state the optimization problems we shall encounter (in both finite- or infinite-dimensional form) by using “min” in place of “inf”. Let us state explicitly these two hypotheses in the following assumption:

Assumption 1.1 An optimal solution γ° to Problem P exists and is unique, i.e., the infimum of the functional F over \mathcal{S} is a minimum and is attained only for γ° . More in general, such properties are required in almost any finite- or infinite-dimensional optimization problem stated in the book, i.e., an optimal solution exists and is unique.

△

On the basis of Assumption 1.1, (1.5) can be written as follows:

$$\text{find } \gamma^\circ = \operatorname{argmin}_{\gamma \in \mathcal{S}} F(\gamma). \quad (1.11)$$

From here on, in the book, every optimization problem will be stated in the form (1.11). The simplified notation

$$F(\boldsymbol{\gamma}^*) = \min_{\boldsymbol{\gamma} \in \mathcal{S}} F(\boldsymbol{\gamma})$$

will be used, too. Similarly, in order to avoid cumbersome discussions on optimal solutions, we shall generally refer to “the” optimal solution of a given problem, instead of “an” optimal solution. We emphasize, once again, that we are calling *optimal solution* the element that minimizes (or maximizes) a given objective (e.g., the function $\boldsymbol{\gamma}^*$ in Problem P, a vector in mathematical programming) and *optimal value* the value taken by the objective in correspondence of the optimal solution.

As regards the existence of optimal solutions stated in Assumption 1.1, in the finite-dimensional case, in which the functional F is replaced by a function $f : X \rightarrow \mathbb{R}$ for $X \subseteq \mathbb{R}^d$, $d \geq 1$, a sufficient condition is given by the continuity of f together with closedness and boundedness (i.e., compactness) of X . This is the Weierstrass theorem of classical calculus [71, Sect. 3.1]; the case of lower semi-continuous functions is covered by the so-called Generalized Weierstrass theorem [71, Theorem 9.13]. In the infinite-dimensional case, similar conditions hold (one has existence if the functional F is continuous and \mathcal{S} is compact) but being merely closed and bounded does not imply compactness, as discussed in Sect. 1.3.2, where we remarked that in infinite dimension compactness is not as “easy to achieve” as in finite dimension. There exist various infinite-dimensional versions of the Weierstrass Theorem, e.g., for continuous functionals, semicontinuous functionals, weakly semicontinuous functionals, growing functionals, and convex functionals. We refer the reader to [21, Sect. 2.3] and [28, Sects. 1.4, 1.5].

In most numerical examples presented in the book, we shall replace the unknown decisions functions with multilayer feedforward neural networks. As said previously, this replacement approximates the original IDO problem with a finite-dimensional NLP one. It is now important to remark that in general the new approximating problem has many optimal solutions. The cost of this new problem takes on the form of a composite function containing I/O maps of neural networks. To fix ideas, consider the simple case of nets with a single hidden layer, a single output variable, and $\tanh(\cdot)$ as activation function (see Figs. 3.2 and 3.3). In [72], a network is called “minimal” or “irreducible” if its input/output map is not equivalent to a net with fewer neural units. This property can be achieved under simple assumptions. It is easy to see that the I/O map of a minimal network does not change if its n neural units are permuted and the signs of all the weights associated to a particular node are changed. The number of all these transformations is $2^n n!$. Hence, if an optimal solution to the NLP problem above exists, then there are at least $2^n n!$ optimal solutions. In this case, when we need to study the properties of an optimal solution, we shall restrict ourselves to consider the neighborhood of *any* minimizer.

1.4.2 Other Definitions, Conventions, and Assumptions

In this section, some other definitions and assumptions, which will be used in the next section and later in the book, are given.

Norms

Let z be an element of a normed space \mathcal{H} .

1. If the normed linear space is \mathbb{R} , then $|z|$ denotes the absolute value of z .
2. If the normed linear space has finite dimension $d > 1$, then the norm of z is denoted by enclosing it within two “|” symbols; the subscript, if present, specifies the type of norm. For the sake of simplicity, the standard Euclidean norm on \mathbb{R}^d is denoted merely by “ $|\cdot|$ ” (note that this is consistent with the notation “ $|z|$ ” used for the one-dimensional case, as $|z| = \sqrt{z^2}$).
3. If the normed linear space has infinite dimension, then two “ $\|$ ” symbols are used and the subscript, if present, specifies the type of norm.

For example, in the d -dimensional case, if $z \triangleq \text{col}(z_1, \dots, z_d) \in \mathbb{R}^d$, then we let

$$|z| \triangleq \sqrt{\sum_{i=1}^d z_i^2} = \sqrt{z^\top z}, \quad |z|_1 \triangleq \sum_{i=1}^d |z_i|, \quad |z|_\infty \triangleq \max_{i=1, \dots, d} |z_i|,$$

and, for a symmetric positive-definite $d \times d$ matrix Q , $|z|_Q \triangleq \sqrt{z^\top Q z}$.

Probability Densities

Unless otherwise stated, throughout the book we shall suppose that every continuous random variable has a well-defined probability density function. This will enable us to avoid resorting to some technicalities. To simplify the notation, we shall denote the probability density function of a continuous random variable z merely by $p(z)$ (instead of $p_z(z)$).

For IDO problems stated in the stochastic framework considered in Sect. 1.5, i.e., when the cost functional F takes on the form $F(\cdot) = E_z J(\cdot, z)$, we make the following assumption.

Assumption 1.2 When the functional in Problems P or PM takes on the form (1.9), the probability density function of z is assumed to exist and to be known. \triangleleft

It is important to point out that several other definitions and assumptions are typically needed to ensure that the underlying stochastic framework is well-posed (see [19] and the references cited therein). This is out of the scope of the present introductory chapter. Further details will be given in later chapters when dealing with the optimal control of stochastic systems.

Subscripts and Superscripts

Throughout the book, subscripts, superscripts, and other symbols will be sometimes reported and sometimes not. Generally, they will be omitted whenever they are not strictly necessary. Often, if there are no problems of ambiguity, the reader will not be warned of this but there will be no risk of getting confused.

1.5 Examples of Infinite-Dimensional Optimization Problems

In this section, we describe some instances of the IDO problems that refer to very different areas. It has to be remarked that Problem P is characterized by three elements, namely, the infinite-dimensional linear space \mathcal{H} , the set $\mathcal{S} \subseteq \mathcal{H}$ of admissible functions, and the objective functional $F: \mathcal{S} \rightarrow \mathbb{R}$. We shall not define \mathcal{H} and $\mathcal{S} \subseteq \mathcal{H}$ for the examples of this section, so, in a strict sense, they are not instances of Problem P. A similar remark holds for Problem PM. However, from the description of the problems the reader can understand that the linear spaces of functions and the subsets of admissible solutions are, loosely speaking “the most general ones can imagine,” or, more precisely, ones satisfying the minimal requirements for the existence of optimal solutions to the associated IDO problems. Hence, with a little terminological abuse, we refer to them as “instances of Problems P or PM.”

Moreover, as done till now in this chapter, in this section we do not specify any norm. Indeed, here we still do not deal with approximate solutions. Then, at the moment we do not need to measure the distances between the optimal functions γ° and the FSP approximating functions (i.e., the approximation errors) that will be a central issue throughout the book. Instead, when we address IDO problems to be approximated by FDO ones, the spaces of functions should be characterized by specific regularity conditions and endowed with a norm.

1.5.1 A Deterministic Continuous-Time Optimal Control Problem

We consider a continuous-time dynamic system described by the state equation

$$\dot{\mathbf{x}}(t) = f_c[\mathbf{x}(t), \mathbf{u}(t), t]. \quad (1.12)$$

At a certain initial time t_0 , the system is in the state $\mathbf{x}(t_0) = \hat{\mathbf{x}}_0$. At a given final time t_f , the system state $\mathbf{x}(t_f)$ must reach a terminal region specified by the condition

$$s[\mathbf{x}(t_f)] = \mathbf{0}, \quad (1.13)$$

where s is a vector function of dimension $n_s \leq \dim [\mathbf{x}(t)]$. The cost functional to be minimized is chosen as

$$J = \int_{t_0}^{t_f} h_c[\mathbf{x}(t), \mathbf{u}(t), t] dt + h_f[\mathbf{x}(t_f)]. \quad (1.14)$$

In many practical situations, constraints may be imposed on the state and control variables. Usually, such constraints can be expressed by a set of inequalities or equalities of the forms

$$\begin{aligned} g_1[\mathbf{x}(t), \mathbf{u}(t)] &\geq \mathbf{0}, \\ g_2[\mathbf{x}(t), \mathbf{u}(t)] &= \mathbf{0}, \end{aligned} \quad (1.15)$$

respectively, where g_1 and g_2 are vector functions of appropriate dimensions. We can now state the following continuous-time optimal control problem.

Problem 1.2 Find the optimal control law $\mathbf{u}^\circ(t)$, $t_0 \leq t \leq t_f$, that transfers the system from the initial state \mathbf{x}_0 to the terminal region $s[\mathbf{x}(t_f)] = \mathbf{0}$, minimizes Cost Functional (1.14), and verifies State Equation (1.12) and Constraints (1.15).

△

Problem 1.2 is in the form of Problem P. Indeed, we have only one unknown function to determine, that is, $\mathbf{u}(\cdot)$. Hence, it is simply $M = 1$. Formally, we may let $\gamma \triangleq \mathbf{u}(\cdot)$ and write the cost (1.14) as a functional $F(\gamma)$.

Problem 1.2 and its more complex variants are very important in optimal control theory. There are suitable tools for solving Problem 1.2, namely, calculus of variations, maximum (or minimum) principle, and dynamic programming. In some special cases, optimal solutions may be derived analytically. Otherwise, they have to be found by numerical techniques. If the latter are required, quite a natural approach consists in approximating Problem 1.2 by a suitable FDO problem. The simplest way of obtaining such a problem lies in the discretization of the interval $[t_0, t_f]$ into a given finite number of time intervals (for the sake of simplicity, we assume that these time intervals are of equal duration). Then the integral (1.14) is approximated by a summation and the differential state equation (1.12) by a discrete-time state equation. The original Problem 1.2 is reduced to an FDO problem in which the unknowns are the components of the state and control vectors considered at each discrete time, the cost function is the summation, and the constraints are the discrete-time state equations, the final condition, and the inequalities and equalities (1.15). Of course, one must then rely on the possibility that some mathematical programming algorithm may exist that solves the obtained FDO problem. Clearly, if Problem 1.2 is approximated by a mathematical programming problem with a finite number of variables, we no longer have an IDO problem but an FDO one. We shall present an example of the above-mentioned discretization in Sect. 6.7.

1.5.2 A Continuous-Time Network Flow Problem

A classical problem of finite-dimensional linear programming is the network flow problem, in which one has to maximize the flow of a commodity between a source and a destination of a network, whose arcs are subject to capacity constraints. When such capacity limitations are constant over time, the solution can be efficiently found by the Ford–Fulkerson algorithm [33]. However, there are applications in which one has to consider time-varying capacities. This leads to an *infinite-dimensional version* of the maximum network flow problem [10, Sect. 1.2.2].

Consider a system of r water reservoirs R_1, \dots, R_r of respective capacities $c_1(t), \dots, c_r(t)$ (of course, other liquids may be considered) that have to satisfy a time-varying demand $d(t)$ from a destination over a time interval $[0, T]$. The rate at which water flows into R_i is $f_i(t)$; F_i is the maximum rate at which water can be fed from the i th reservoir, $i = 1, \dots, r$. The water initially stored in R_i is denoted by s_i . If it happens that the water flowing into a reservoir exceeds the capacity of the latter, then it is spilled to waste. Moreover, suppose that no water can be fed back to the reservoirs.

Denoting by $u_i(t)$ the rate at which water is fed from the i th reservoir at time t and by $w_i(t)$ the rate of water spillage, the feed and storage constraints can be expressed, respectively, as

$$0 \leq u_i(t) \leq F_i, \quad t \in [0, T], \quad i = 1, \dots, r \quad (1.16)$$

and

$$0 \leq \int_0^t [f_i(\tau) - u_i(\tau) - w_i(\tau)] d\tau + s_i \leq c_i(t), \quad t \in [0, T], \quad i = 1, \dots, r. \quad (1.17)$$

Moreover, there are the constraints that the total rate fed does not exceed the demand and that the spillage rate is nonnegative:

$$\sum_{i=1}^r u_i(t) \leq d(t), \quad t \in [0, T], \quad (1.18)$$

$$w_i(t) \geq 0, \quad t \in [0, T], \quad i = 1, \dots, r. \quad (1.19)$$

The target consists in meeting “as closely as possible” the overall water demand over the period $[0, T]$, given by

$$\int_0^T \left[\sum_{i=1}^r u_i(\tau) \right] d\tau. \quad (1.20)$$

Thus, the problem can be stated as follows.

Problem 1.3 Find the optimal water feeding functions $u_1^\circ(t), \dots, u_r^\circ(t)$, $t \in [0, T]$, that maximize Functional (1.20) subject to Constraints (1.16)–(1.19). \triangleleft

Problem 1.3 is an infinite-dimensional linear programming problem, which takes on the form of Problem PM with $M = r$.

1.5.3 A T -Stage Stochastic Optimal Control Problem

Let us consider a dynamic system that evolves stage after stage according to the following discrete-time stochastic state equation:

$$\mathbf{x}_{t+1} = \mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t), \quad t = 0, 1, \dots, T - 1, \quad (1.21)$$

where $\mathbf{x}_t \in \mathbb{R}^d$ is the state vector, $\mathbf{x}_0 = \hat{\mathbf{x}}$ is a given initial state, $\mathbf{u}_t \in \mathbb{R}^m$ is the control vector, $\boldsymbol{\xi}_t \in \mathbb{R}^q$ is a random noise vector, and T is the number of decision stages. The state vector is assumed to be perfectly measurable.

The cost function is given by

$$J = \sum_{t=0}^{T-1} h_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t) + h_T(\mathbf{x}_T), \quad (1.22)$$

where $h_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t)$ is the cost incurred at each stage t , and $h_T(\mathbf{x}_T)$ is the final cost that has to be paid at the end of the decisional process.

The random vectors $\boldsymbol{\xi}_t$, $t = 0, 1, \dots, T - 1$ are defined on suitable probability spaces and are assumed to be mutually independent with known probability density functions $p_t(\boldsymbol{\xi}_t)$.

As the state \mathbf{x}_t summarizes the past “history” of the dynamic system up to the stage t , the control functions take on the feedback form

$$\mathbf{u}_t = \boldsymbol{\mu}_t(\mathbf{x}_t), \quad t = 0, 1, \dots, T - 1. \quad (1.23)$$

Then, we can state the optimal control problem as follows.

Problem 1.4 Find the optimal control functions $\boldsymbol{\mu}_0^\circ(\mathbf{x}_0), \boldsymbol{\mu}_1^\circ(\mathbf{x}_1), \dots, \boldsymbol{\mu}_{T-1}^\circ(\mathbf{x}_{T-1})$ that minimize the expected value of Cost (1.22) subject to Constraints (1.21). \triangleleft

It is straightforward to interpret Problem 1.4 in terms of Problem PM with $M = T$. First of all, we define a *control law* as the collection of *control functions* (the distinction between control law and control functions will be used frequently throughout the book). We express this collection in a vectorial form, i.e.,

$$\boldsymbol{\gamma} \triangleq \text{col}(\boldsymbol{\mu}_0, \dots, \boldsymbol{\mu}_{T-1}). \quad (1.24)$$

Now, let us merge all the random vectors ξ_t into a single vector z , that is,

$$z \triangleq \text{col}(\xi_t, t = 0, 1, \dots, T - 1). \quad (1.25)$$

We eliminate the state vector x_t ($t = 1, \dots, T$) from the cost (1.22) by using repeatedly State Equation (1.21). Then, using Definitions (1.24) and (1.25), such a cost can be written in the form

$$J(\gamma, z) = J(\mu_0, \dots, \mu_{T-1}, \xi_0, \dots, \xi_{T-1}). \quad (1.26)$$

Thus, Problem 1.4 can be restated as follows.

Problem 1.5 *Find the optimal control law γ° that minimizes the expected value of Cost (1.26), i.e.,*

$$F(\gamma) \triangleq \underset{z}{\mathbb{E}} J(\gamma, z). \quad (1.27)$$

□

Problems 1.4 or 1.5 will be faced in Chap. 7.

1.5.4 An Optimal Estimation Problem

Let us address the state equation (1.21). The state vector is observed through the noisy measurement channel

$$\mathbf{y}_t = \mathbf{g}_t(x_t, \eta_t), \quad t = 0, 1, \dots, T, \quad (1.28)$$

where $\mathbf{y}_t \in \mathbb{R}^p$ is the observation vector and $\eta_t \in \mathbb{R}^r$ is a measurement random noise. Analogously to the framework considered in Sect. 1.5.3, the probability density functions of all random vectors x_0 , $\{\xi_t, t = 0, 1, \dots, T - 1\}$, and $\{\eta_t, t = 0, 1, \dots, T\}$ are assumed to be known. As in (1.25), we merge them into a single vector z , that is,

$$z \triangleq \text{col}(x_0, \xi_t, t = 0, 1, \dots, T - 1, \eta_t, t = 0, 1, \dots, T). \quad (1.29)$$

The controls u_0, \dots, u_{T-1} are regarded as known vectors. We want to estimate the state and the random vectors as functions of the information vector

$$\mathbf{I}_T \triangleq \text{col}(\mathbf{y}_0, \dots, \mathbf{y}_T, \mathbf{u}_0, \dots, \mathbf{u}_{T-1}).$$

Then, the estimation functions of x_t , ξ_t , and η_t take on the forms

$$\hat{x}_t = \mathbf{a}_t(\mathbf{I}_T), \quad t = 0, 1, \dots, T,$$

$$\hat{\xi}_t = \mathbf{b}_t(\mathbf{I}_T), \quad t = 0, 1, \dots, T-1,$$

$$\hat{\eta}_t = \mathbf{c}_t(\mathbf{I}_T), \quad t = 0, 1, \dots, T,$$

respectively. The *estimation law* is given by the following collection of *estimation functions*:

$$\boldsymbol{\gamma} \triangleq \text{col} (\mathbf{a}_0, \dots, \mathbf{a}_T, \mathbf{b}_0, \dots, \mathbf{b}_{T-1}, \mathbf{c}_0, \dots, \mathbf{c}_T).$$

Let us define the estimation cost

$$J = |\hat{\mathbf{x}}_0 - \bar{\mathbf{x}}_0|_{V_x}^2 + \sum_{t=0}^T |\mathbf{y}_t - \mathbf{g}_t(\hat{\mathbf{x}}_t, \hat{\eta}_t)|_{V_y}^2 + \sum_{t=0}^{T-1} |\hat{\xi}_t|_{V_{\xi}}^2 + \sum_{t=0}^T |\hat{\eta}_t|_{V_{\eta}}^2, \quad (1.30)$$

where $\bar{\mathbf{x}}_0 \triangleq \mathbb{E}(\mathbf{x}_0)$, $\mathbb{E}(\xi_t) = \mathbf{0}$, $\mathbb{E}(\eta_t) = \mathbf{0}$, V_x , V_y , V_{ξ} , and V_{η} are given symmetric positive-definite matrices. (More general functions might be considered in (1.30); for instance, instead of $|\hat{\mathbf{x}}_0 - \bar{\mathbf{x}}_0|_{V_x}^2$, we might write $\chi(|\hat{\mathbf{x}}_0 - \bar{\mathbf{x}}_0|)$, where $\chi(v) : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ is an increasing function for $v \geq 0$, with $\chi(0) = 0$.) Similar functions might replace the other quadratic functions in (1.30). Then, we can state the estimation problem in the following form.

Problem 1.6 Find the optimal estimation functions $\mathbf{a}_0^\circ, \dots, \mathbf{a}_T^\circ, \mathbf{b}_0^\circ, \dots, \mathbf{b}_{T-1}^\circ, \mathbf{c}_0^\circ, \dots, \mathbf{c}_T^\circ$ that minimize the expected value of Cost (1.30). \triangleleft

After the elimination of \mathbf{x}_t and \mathbf{y}_t , Cost (1.30) can be written as follows:

$$J(\boldsymbol{\gamma}, z) = J(\mathbf{a}_0, \dots, \mathbf{a}_T, \mathbf{b}_0, \dots, \mathbf{b}_{T-1}, \mathbf{c}_0, \dots, \mathbf{c}_T, z). \quad (1.31)$$

As the unknown functions are $M = 3T + 2$, the optimal estimation problem can be stated in the form of Problem PM, with $M = 3T + 2$.

Problem 1.7 Find the optimal estimation law $\boldsymbol{\gamma}^\circ$ that minimizes the expected value of Cost (1.31), i.e.,

$$F(\boldsymbol{\gamma}) \triangleq \mathbb{E}_z J(\boldsymbol{\gamma}, z). \quad (1.32)$$

\triangleleft

1.5.5 A Static Team Optimal Control Problem

In many situations, several decision-makers $\{DM_1, \dots, DM_M\}$ – sharing different information patterns – cooperate on the accomplishment of a common goal. Typical examples may be encountered in communication networks, large-scale traffic

systems, distributed control systems, etc. Each decision-maker DM_i implements a control function of the form

$$\boldsymbol{u}_i = \boldsymbol{\mu}_i(\boldsymbol{I}_i), \quad i = 1, \dots, M,$$

on the basis of a *personal* information vector \boldsymbol{I}_i obtained through its own measurement channel. In “static” teams (see Sect. 9.1.1), the measurement channel is given by

$$\boldsymbol{I}_i = \boldsymbol{g}_i(\boldsymbol{z}), \quad i = 1, \dots, M.$$

The random vector \boldsymbol{z} represents all uncertainties of the external world. The team control law is given by

$$\boldsymbol{\gamma} \triangleq \text{col}(\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_M). \quad (1.33)$$

All the DMs cooperate to minimize the expected value of the common cost

$$J(\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_M, \boldsymbol{z}). \quad (1.34)$$

They are assumed to know the whole a priori information characterizing the optimization framework, namely, the functions $\boldsymbol{g}_1, \dots, \boldsymbol{g}_M$, the cost J , and the probability density function of \boldsymbol{z} . Then, the team optimal control problem is stated as follows.

Problem 1.8 *Find the team optimal control law $\boldsymbol{\gamma}^\circ$ that minimizes the expected value of Cost (1.34), i.e.,*

$$F(\boldsymbol{\gamma}) \triangleq \underset{\boldsymbol{z}}{\text{E}} J(\boldsymbol{\gamma}, \boldsymbol{z}). \quad (1.35)$$

△

This problem too takes on the form of Problem PM. Generalized versions of Problem 1.8 will be examined in Chap. 9.

References

1. Adams RA, Fournier JJF (2003) Sobolev spaces. Academic Press
2. Alessandri A, Baglietto M, Battistelli G, Gaggero M (2011) Moving-horizon state estimation for nonlinear systems using neural networks. *IEEE Trans Neural Netw* 22:768–780
3. Alessandri A, Baglietto M, Parisini T, Zoppoli R (1999) A neural state estimator with bounded errors for nonlinear systems. *IEEE Trans Autom Control* 44:2028–2042
4. Alessandri A, Cervellera C, Macchiò D, Sanguineti M (2010) Optimization based on quasi-Monte Carlo sampling to design of state estimators for nonlinear systems. *Optimization* 59:963–984
5. Alessandri A, Cervellera C, Sanguineti M (2007) Design of asymptotic estimators: an approach based on neural networks and nonlinear programming. *IEEE Trans Neural Netw* 18:86–96
6. Alessandri A, Parisini T, Zoppoli R (1997) Neural approximations for nonlinear finite-memory state estimation. *Int J Control* 67:275–302

7. Alessandri A, Parisini T, Zoppoli R (2001) Sliding-window neural state estimation in a power plant heater line. *Int J Adapt Control Signal Process* 15:815–836
8. Alessandri A, Sanguineti M (2005) Optimization of approximating networks for optimal fault diagnosis. *Optim Methods Softw* 20:235–260
9. Alessio A, Bemporad A (2009) A survey on explicit model predictive control. In: Magni L, Raimondo D, Allgöwer F (eds) Nonlinear model predictive control. Lecture notes in control and information sciences, vol 384. Springer, pp 345–369
10. Anderson EJ, Nash P (1987) Linear programming in infinite-dimensional spaces. Wiley
11. Baglietto M, Parisini T, Zoppoli R (2001) Distributed-information neural control: the case of dynamic routing in traffic networks. *IEEE Trans Neural Netw* 12:485–502
12. Baglietto M, Parisini T, Zoppoli R (2001) Numerical solutions to the Witsenhausen counterexample by approximating networks. *IEEE Trans Autom Control* 46:1471–1477
13. Barto A (1990) Connectionist learning for control: an overview. In: Miller WT III, Sutton RS, Werbos PJ (eds) Neural networks for control. MIT Press, pp 5–58
14. Beard RW, McLain TW (1998) Successive Galerkin approximation algorithms for nonlinear optimal and robust control. *Int J Control* 71:717–743
15. Bellman R (1957) Dynamic programming. Princeton University Press
16. Bemporad A, Borrelli F, Morari M (2003) Min-max control of constrained uncertain discrete-time linear systems. *IEEE Trans Autom Control* 48:1600–1606
17. Bemporad A, Morari M, Dua V, Pistikopoulos EN (2002) The explicit linear quadratic regulator for constrained systems. *Automatica* 38:3–20
18. Bertsekas DP (2005) Dynamic programming and suboptimal control: a survey from ADP to MPC. *Eur J Control* 11:310–334
19. Bertsekas DP, Shreve SE (1978) Stochastic optimal control—the discrete time case. Academic Press
20. Bertsekas DP, Tsitsiklis JN (1996) Neuro-dynamic programming. Athena Scientific
21. Bobylev NA, Emel'yanov SV, Korovin SK (1999) Geometrical methods in variational problems. Mathematics and its applications, vol 485. Springer
22. Bolla R, Davoli F, Maryni P, Parisini T (1998) An adaptive neural network admission controller for dynamic bandwidth allocation. *IEEE Trans Syst Man Cybern Part B Cybern* 28(592):601
23. Borwein J, Lewis AS (2000) Convex analysis and nonlinear optimization: theory and examples. CMS books in mathematics, Springer
24. Brezis H (2011) Functional analysis. Springer, Sobolev spaces and partial differential equations. Springer
25. Chatelin F (2011) Spectral approximation of linear operators. Classics in applied mathematics. SIAM
26. Chu D, Chen T, Marquez HJ (2006) Explicit robust model predictive control using recursive closed-loop prediction. *Int J Robust Nonlinear Control* 16:519–546
27. Dacorogna B (2008) Direct methods in the calculus of variations, 2nd edn. Springer
28. Daniel JW (1971) The approximate minimization of functionals. Prentice Hall
29. de la Peña M, Bemporad A, Filippi C (2006) Robust explicit MPC based on approximate multi-parametric convex programming. *IEEE Trans Autom Control* 51:1399–1403
30. Ekeland I, Turnbull T (1983) Infinite-dimensional optimization and convexity. The University of Chicago Press
31. Di Febraro A, Parisini T, Sacone S, Zoppoli R (2001) Neural approximations for feedback optimal control of freeway systems. *IEEE Trans Veh Technol* 50:302–313
32. Felgenhauer U (1999). On Ritz type discretizations for optimal control problems. In: Proceedings of the 18th IFIP-ICZ conference. Chapman-Hall, pp 91–99
33. Ford LR, Fulkerson DR (1962) Flows in networks. Princeton University Press
34. Gaggero M, Gnecco G, Sanguineti M (2013) Dynamic programming and value-function approximation in sequential decision problems: error analysis and numerical results. *J Optim Theory Appl* 156:380–416
35. Gelfand IM, Fomin SV (1963) Calculus of variations. Prentice Hall

36. Gnecco G, Sanguineti M (2010) Suboptimal solutions to dynamic optimization problems via approximations of the policy functions. *J Optim Theory Appl* 146:764–794
37. Gnecco G, Sanguineti M, Gaggero M (2012) Suboptimal solutions to team optimization problems with stochastic information structure. *SIAM J Optim* 22:212–243
38. Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press
39. Graettinger TJ, Krogh BH (1988) The acceleration radius: a global performance measure for robotic manipulators. *IEEE J Rob Autom* 4:60–69
40. Granchiarova A, Johansen TA, Tondel P (2007) Computational aspects of approximate explicit nonlinear model predictive control. In Findeisen R, Allgöwer F, Biegler LT (eds) Assessment and future directions of nonlinear model predictive control. Lecture notes in control and information sciences, vol 358. Springer, pp 181–192
41. Grover P, Park SY, Sahai A (2013) Approximately optimal solutions to the finite-dimensional Witsenhausen's counterexample. *IEEE Trans Autom Control* 58:2189–2204
42. Hager WW (1975) The Ritz-Trefftz method for state and control constrained optimal control problems. *SIAM J Numer Anal* 12:854–867
43. Hettich R, Kortanek KO (1993) Semi-infinite programming: theory, methods, and applications. *SIAM Rev* 35:380–429
44. Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. *Neural Comput* 18:1527–1554
45. Hinton GH (2007) Learning multiple layers of representation. *Trends Cognit Sci* 11:428–434
46. Jones CN, Grieder P, Raković SV (2006) A logarithmic-time solution to the point location problem for parametric linear programming. *Automatica* 42:2215–2218
47. Jones CN, Morari M (2008) The double description method for the approximation of explicit MPC control laws. In: Proceedings of the IEEE conference on decision and control, pp 4724–4730
48. Kainen P, Kůrková V, Sanguineti M (2003) Minimization of error functionals over variable-basis functions. *SIAM J Optim* 14:732–742
49. Kainen PC, Kůrková V, Sanguineti M (2012) Dependence of computational models on input dimension: tractability of approximation and optimization tasks. *IEEE Trans Inf Theory* 58:1203–1214
50. Kerrigan EC, Maciejowski JM (2004) Feedback min-max model predictive control using a single linear program: robust stability and the explicit solution. *Int J Robust Nonlinear Control* 14:395–413
51. Kůrková V, Sanguineti M (2001) Bounds on rates of variable-basis and neural-network approximation. *IEEE Trans Inf Theory* 47:2659–2665
52. Kůrková V, Sanguineti M (2002) Comparison of worst case errors in linear and neural network approximation. *IEEE Trans Inf Theory* 48:264–275
53. Kůrková V, Sanguineti M (2008) Geometric upper bounds on rates of variable-basis approximation. *IEEE Trans Inf Theory* 54:5681–5688
54. Lee JT, Lau E, Ho Y-C (2001) The Witsenhausen counterexample: a hierarchical search approach for nonconvex optimization problems. *IEEE Trans Autom Control* 46:382–397
55. Mäkilä PM, Toivonen HT (1987) Computational methods for parametric LQ problems—A survey. *IEEE Trans Autom Control* 32:658–671
56. Muñoz de la Peña D, Ramirez DR, Camacho EF, Alamo T (2006) Explicit solution of min-max MPC with additive uncertainties and quadratic criterion. *Systems Control Lett* 55:266–274
57. Parisini T, Sanguineti M, Zoppoli R (1998) Nonlinear stabilization by receding-horizon neural regulators. *Int J Control* 70:341–362
58. Parisini T, Zoppoli R (1994) Neural networks for feedback feedforward nonlinear control systems. *IEEE Trans Neural Netw* 5:436–449
59. Parisini T, Zoppoli R (1995) A receding-horizon regulator for nonlinear systems and a neural approximation. *Automatica* 31:1443–1451
60. Parisini T, Zoppoli R (1998) Neural approximations for infinite-horizon optimal control of nonlinear stochastic systems. *IEEE Trans Neural Netw* 9:1388–1408
61. Pinkus A (1985) *n-widths in approximation theory*. Springer

62. Ritz W (1909) Über eine neue Methode zur Lösung gewisser Variationsprobleme der mathematischen Physik. *Journal für die reine und angewandte Mathematik* 135:1–61
63. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning internal representations by error propagation. In: Rumelhart DE, McClelland JL (eds) *Parallel distributed processing: explorations in the microstructure of cognition—Vol. 1: foundations*. MIT Press, pp 318–362
64. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations by back-propagating errors. *Nature* 323:533–536
65. Sage AP (1968) *Optimum systems control*. Prentice Hall
66. Saldi N, Linder T, Yüksel S (2018) *Finite approximations in discrete-time stochastic control*. Birkhäuser
67. Sanguineti M (2008) Universal approximation by ridge computational models and neural networks: a survey. *Open Appl Math J* 2:31–58
68. Sirisena HR, Chou FS (1979) Convergence of the control parametrization Ritz method for nonlinear optimal control problems. *J Optim Theory Appl* 29:369–382
69. Spall JC (2003) *Introduction to stochastic search and optimization: estimation, simulation and control*. Wiley
70. Stein O (2003) *Bi-level Strategies in semi-infinite programming*. Kluwer
71. Sundaram RK (1996) *A first course in optimization theory*. Cambridge University Press
72. Sussman HJ (1992) Uniqueness of the weights for minimal feedforward nets with a given input-output map. *Neural Netw* 5:589–593
73. Werbos PJ (1974) Beyond regression: new tools for prediction and analysis in the behavioral sciences. PhD thesis, Harvard University, Committee on applied mathematics
74. Werbos PJ (1990) Backpropagation through time: what it does and how to do it. *Proc IEEE* 78:1550–1560
75. Yoshida K (1965) *Functional analysis*. Academic Press
76. Zoppoli R, Sanguineti M, Parisini T (2002) Approximating networks and extended Ritz method for the solution of functional optimization problems. *J Optim Theory Appl* 112:403–440

Chapter 2

From Functional Optimization to Nonlinear Programming by the Extended Ritz Method



In this chapter, we describe the methodology of approximate solution of Problem P – called extended Ritz method (ERIM) in Chap. 1 – using certain parametrized families of functions. To this end, in Sect. 2.1, we introduce the fixed-structure parametrized (FSP) functions. For every positive integer n , called *model complexity*, and every finite-dimensional vector \mathbf{w}_n , these are functions $\gamma^d(\cdot, \mathbf{w}_n)$ with a fixed structure, where the components of \mathbf{w}_n are “*free*” parameters to be optimized. The FSP functions typically satisfy the following nestedness property: when $n_2 > n_1$, the set of mappings that can be represented by FSP functions with n_1 fixed parameters is contained in the set corresponding to n_2 parameters.

By substituting the FSP functions into the cost functional F , we obtain a sequence of nonlinear programming (NLP) problems – described in Sect. 2.2 – easier to solve than the original infinite-dimensional optimization (IDO) (or functional optimization) problem and whose optimal solutions better and better approximate (in a sense to be specified) the optimal solution to Problem P. Techniques for solving the approximating problems are considered in Sect. 2.3. Suppose that each NLP problem of the sequence admits an optimal solution given by an optimal FSP function $\gamma(\cdot, \mathbf{w}_n^\circ)$ and the related minimum cost $F(\gamma_n^\circ)$, where $\gamma_n^\circ(\cdot) \triangleq \gamma(\cdot, \mathbf{w}_n^\circ)$ and $F_n^\circ \triangleq F(\gamma_n^\circ)$. Since we shall be interested in estimating the accuracy of the solutions γ_n° , i.e., their distances from γ° , we endow the linear space \mathcal{H} with a norm $\|\cdot\|_{\mathcal{H}}$. Then, \mathcal{H} becomes a normed linear space. Suppose that Problem P has an optimal solution given by an optimal decision function γ° and the related minimum cost F° . If sequences $\{\gamma_n^\circ\}_{n=1}^\infty$ and $\{F_n^\circ\}_{n=1}^\infty$ converge to γ° in the norm $\|\cdot\|_{\mathcal{H}}$ and to F° , respectively, then we say that $\{\gamma_n^\circ\}_{n=1}^\infty$ is an *optimizing sequence* and that its FSP functions are *optimizing FSP functions*.¹ These concepts are introduced in Sect. 2.4.

¹Four sequences of approximating FSP functions are defined and discussed. Such sequences are connected to each other by properties that the reader may find somewhat difficult and tedious. To

Of course, searching for approximate solutions to Problem P does not mean that we have to take the model complexity n to infinity. Actually, what we have to do is to choose a maximum admissible error $\varepsilon > 0$ and to consider merely optimizing FSP functions such that both the errors $\|\gamma_n^\circ - \gamma^\circ\|_{\mathcal{H}}$ and $F_n^\circ - F^\circ$ do not exceed ε , which occurs for suitably large values of n . Then, the dimension d of the vector x – on which the decision functions γ and γ_n depend – comes into play. Clearly, the three quantities n , ε , and d are linked to one another. For a given dimension d and a maximum allowable error ε , let $n^\circ(d, \varepsilon)$ be the minimum model complexity for which an upper bound smaller than or equal to ε is guaranteed on the abovementioned error. Given a sequence $\{\mathbf{P}^d\}_{d=1}^\infty$ of IDO problems characterized by increasing values of the dimension d , we have to expect the minimum model complexity $n^\circ(d, \varepsilon)$ to grow when d increases. When IDO problems are stated in high-dimensional settings, it is very important that the growth of $n^\circ(d, \varepsilon)$ with respect to d remains “moderate”, in order to limit the complexity of the FSP functions, hence, the online computational burden of the FSPs that implement the decision-makers (DMs). A polynomial growth of $n^\circ(d, \varepsilon)$ can be considered “moderate”, whereas an exponential one cannot, since it gives rise to an instance of the phenomenon of the *curse of dimensionality*. When the minimum model complexity $n^\circ(d, \varepsilon)$ of the FSP functions grows at most polynomially with respect to d , the sequence is called *polynomially complex optimizing sequence*. This is addressed in Sect. 2.5.

Here and there in the book, it is only necessary to approximate functions, like the optimal cost-to-go and control functions that make their appearance in dynamic programming. Therefore, it is natural to wonder which FSP functions are able to approximate “large” families of functions, such as continuous or square-integrable functions, to any degree of accuracy when their model complexities n increase. For certain families of FSP functions, the answer is positive: hence, we say that they are *approximating FSP functions* in the corresponding spaces. They are considered in Sect. 2.6. When they benefit by suitable requirements on the growth of their model complexities with increasing dimension d , we call them *polynomially complex approximating FSP functions*. This family of FSP functions is addressed in Sect. 2.7.

In Sect. 2.8, we investigate the relationships between polynomially complex approximating FSP functions and polynomially complex optimizing ones. In particular, we show that the existence of the former is an important premise for the existence of the latter. This is an issue that is worth pointing out as the polynomially complex approximating FSP functions have been studied in the literature much more deeply than the polynomially complex optimizing ones (in both cases, different terminologies are used in the literature). In Sect. 2.8 we present theoretical results that enable one to transfer basic properties – established in the area of function approximation – to the area of IDO optimization (see the “transfer” hypotheses in Fig. 1.2).

better understand the relationships among the four families, the reader may refer to Tables 2.1 and 2.2, and to Fig. 2.10.

2.1 Fixed-Structure Parametrized (FSP) Functions

In Chap. 1, we pointed out that solving Problem P analytically is a practically impossible task, unless special assumptions are verified. Therefore, approximating techniques have to be sought, once one has decided in which sense the term “approximate” is to be understood and how (e.g., by which norm on \mathcal{H}) the distances of approximate decision functions from the optimal one are measured. In designing methods for approximately solving Problem P, a possible approach consists in replacing the original problem with a sequence of approximating problems, which are supposed to be easier to solve than Problem P. Several examples can be found of this kind of “replacement” of a complex problem with a sequence of simpler ones. Consider, for instance, the method of penalty functions for solving a constrained NLP problem. This method transforms the constraints into larger and larger penalty terms that are added to the cost function. The original constrained problem is then converted into a sequence of unconstrained ones.

For a positive integer n , let \mathbf{w}_n be a finite-dimensional *vector of “free” parameters*. Our approach consists in constraining the function γ to take on a *given structure* $\gamma_n(\cdot, \mathbf{w}_n)$, in which the vector \mathbf{w}_n has to be determined to minimize the cost functional. The integer n , used as a subscript, is aimed at quantifying the “complexity” of the structure of the functions $\gamma_n(\cdot, \mathbf{w}_n)$. As we shall see later in this chapter, for the kind of structures chosen in this book, the number of free parameters or, equivalently, the number of components of the vector \mathbf{w}_n , depends on n through a known function

$$\mathcal{N} : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+. \quad (2.1)$$

Hence, $\mathbf{w}_n \in \mathbb{R}^{\mathcal{N}(n)}$. Intuitively, the preassigned structure has to be chosen sufficiently “mouldable” by \mathbf{w}_n , so that it can approximate the unknown optimal function γ° – which solves Problem P – to any desired degree of accuracy (measured by the norm on \mathcal{H}), possibly with a number of free parameters that grows only “moderately” with the dimension of the argument vector \mathbf{x} . The term “moderately” will be specified later on.

For any given value of the integer n , we consider the sets composed of all the functions $\gamma(\cdot, \mathbf{w}_n)$ obtained when the parameter vector \mathbf{w}_n varies in $\mathbb{R}^{\mathcal{N}(n)}$. These sets are defined as follows:

$$\mathcal{A}_n \triangleq \left\{ \gamma(\cdot, \mathbf{w}_n) : \mathbb{R}^d \rightarrow \mathbb{R}^m, \mathbf{w}_n \in \mathbb{R}^{\mathcal{N}(n)} \right\}, \quad d = 1, 2, \dots, n = 1, 2, \dots \quad (2.2)$$

The following definition summarizes the foregoing.

Definition 2.1 Let d and n be positive integers, $\mathcal{N} : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$ a given mapping, and $\mathbf{w}_n \in \mathbb{R}^{\mathcal{N}(n)}$ a parameter vector. For every $\mathbf{w}_n \in \mathbb{R}^{\mathcal{N}(n)}$, the functions

$$\gamma_n(\cdot, \mathbf{w}_n) : \mathbb{R}^d \rightarrow \mathbb{R}^m, \quad n = 1, 2, \dots \quad (2.3)$$

are called “fixed-structure parametrized functions.” The integer n is called “model complexity.” In the case of one-hidden-layer (OHL) networks with the structure (1.1) and (1.2), this integer coincides with the number of basis functions and is called “basis cardinality.” \triangleleft

We make the following assumption on the sequence of sets $\{\mathcal{A}_n\}_{n=1}^\infty$.

Assumption 2.1 The sequence of sets $\{\mathcal{A}_n\}_{n=1}^\infty$ has the (infinite) nested structure

$$\mathcal{A}_1 \subset \mathcal{A}_2 \subset \cdots \subset \mathcal{A}_n \subset \cdots. \quad (2.4)$$

\triangleleft

Moreover, throughout the book, the following definition will be used.

Definition 2.2 The FSP functions that verify Assumption 2.1 are said to enjoy the “nestedness property.” \triangleleft

The nestedness property guarantees that the optimal value of the functional, obtained by restricting the optimization over functions taking on the form of FSP functions with model complexity n , is monotonically nonincreasing with n (see Sect. 2.2 and, in particular, Eq. (2.14)). So, by the nestedness property, if one is able to find a value \bar{n} for which $F_{\bar{n}}^\circ - F^\circ \leq \varepsilon$, then $F_n^\circ - F^\circ \leq \varepsilon$ for all $n \geq \bar{n}$. The following example shows that the nestedness property does not follow from Definition 2.1, so it has to be introduced as a separate assumption.

Example 2.1 For $d = m = 1$, let us consider FSP functions with the following structure:

$$\gamma_n(x, \mathbf{w}_n) = \sum_{i=1}^n a_i(w_{ni}) \sin(2\pi i x), \quad (2.5)$$

where

$$a_i(w_{ni}) = \begin{cases} 1 - e^{-|w_{ni}|}, & \text{for } n \geq 2 \text{ and } i = 1, \dots, n-1, \\ 1 - |w_{ni}|, & \text{for } i = n. \end{cases}$$

Note that the parameter vector $\mathbf{w}_n \in \mathbb{R}^n$ is unconstrained and that, for each $n \geq 2$, there exists a choice $\bar{\mathbf{w}}_n$ for \mathbf{w}_n such that $\gamma_n(x, \bar{\mathbf{w}}_n) = \sin(2\pi n x)$ but there exists no choice of \mathbf{w}_n that allows one to obtain the functions $\sin(2\pi i x)$ for $i = 1, \dots, n-1$. Then, the corresponding sequence of sets $\{\mathcal{A}_n\}_{n=1}^\infty$ does not satisfy Assumption 2.1. Indeed, for every n there is one function that belongs to \mathcal{A}_n but does not belong to any other element of the sequence of sets. This example, formulated for $d = m = 1$, can be easily generalized to any other choices for d and m . \triangleleft

The families of FSP functions characterized by the nestedness property, although often provided with powerful approximation properties, represent too large and diversified a set to be able to devise a general unified approximation theory. So, in Chap. 3

we shall introduce four families of FSP functions that are used in the book: *linear combinations of fixed-basis functions*, *one-hidden-layer* (OHL) *networks*, *multi-hidden-layer* (MHL) *networks*, and *kernel smoothing models*.

Remark on Notation 2.1 The following notational conventions will be adopted throughout the book for functions that depend on the parameter vector \mathbf{w}_n , like FSP Functions (2.3). We apologize to the reader if the notation may seem complicate and tedious. However, once one gets used to it, it will prove easy and useful.

1. We have included the subscript n in the function $\gamma_n(\cdot, \mathbf{w}_n)$ because, in general, the structure of the function may depend on n . However, for the kind of structures under consideration, we shall see that there is no risk of ambiguity if we drop n and simply write $\gamma(\cdot, \mathbf{w}_n)$.
2. It frequently happens that we wish to write a function $\gamma(\cdot, \mathbf{w}_n)$ without \mathbf{w}_n and other variables on which the function may depend. In this case, dropping the subscript n would generate ambiguity, so, in this situation, we shall maintain the subscript n and write γ_n .
3. Often the parameter vector \mathbf{w}_n contains other subscripts and superscripts. For example, we may have to write $\gamma(\cdot, \mathbf{w}_{tn_i}^\circ)$. Whenever we want to write the function without its arguments, we add the subscripts and superscripts characterizing \mathbf{w} to the function itself. Then, according to the previous point, in these cases we shall write $\gamma_{tn_i}^\circ$.
4. If an index is present in both the function and the parameter vector, the function cannot be written without its arguments. For instance, $\gamma_t(\cdot, \mathbf{w}_{tn_i}^\circ)$ cannot become $\gamma_{tn_i}^\circ$ without generating ambiguity. Indeed, $\gamma_{tn_i}^\circ$ could derive both from $\gamma_t(\cdot, \mathbf{w}_{tn_i}^\circ)$ and $\gamma(\cdot, \mathbf{w}_{tn_i}^\circ)$. Hence, in such cases, we shall keep the complete notation $\gamma_t(\cdot, \mathbf{w}_{tn_i}^\circ)$.

△

2.2 The Sequence of Nonlinear Programming Problems Obtained by FSP Functions of Increasing Complexity

All that has been said till now in this chapter is related to Problem P. Therefore, we have considered only one decision function $\gamma(\cdot)$ to be optimized, only one FSP function $\gamma(\cdot, \mathbf{w}_n)$ that has to replace $\gamma(\cdot)$, only one model complexity n , etc. So, we begin describing the ERIM for Problem P, for which we can use a simple notation. Then, we extend the description to Problem PM, hence, describing the mechanism of replacing the M unknown decision functions $\gamma_1(\cdot), \dots, \gamma_M(\cdot)$ to be optimized with M FSP functions. This requires a little more complex notation but no heavier conceptual difficulties. Subsequently, without loss of generality, we shall come back to Problem P.

2.2.1 The Case of Problem P

In this chapter, we shall measure distances between functions belonging to \mathcal{H} (the ambient space of Problem P) in order to determine approximation errors, to address convergence issues, etc. To this end, we introduce a suitable *norm* on \mathcal{H} . Let us point out this necessity by the following assumption, which will be used throughout the book.

Assumption 2.2 The linear space \mathcal{H} , in which Problem P is stated, is endowed with a norm $\|\cdot\|_{\mathcal{H}}$, thus becoming a normed linear space. Likewise, all linear spaces in which IDO and FDO problems are stated are normed linear spaces, whose norms will be specified from time to time. \triangleleft

For each positive integer n specifying the model complexity of the FSP functions $\gamma(\cdot, \mathbf{w}_n)$, we consider the minimization of the cost functional $F: \mathcal{S} \rightarrow \mathbb{R}$ (see (1.3)), where $\mathcal{S} \subseteq \mathcal{H}$, over the sets

$$\mathcal{S}_n \triangleq \mathcal{A}_n \cap \mathcal{S}, \quad n = 1, 2, \dots, \quad (2.6)$$

given by the intersections of each \mathcal{A}_n with the set \mathcal{S} of feasible solutions of Problem P.

Assumption 2.3 The FSP functions are chosen in such a way that for every positive integer n and every parameter vector \mathbf{w}_n one has $\gamma(\cdot, \mathbf{w}_n) \in \mathcal{H}$ and $\mathcal{S}_n \neq \emptyset$. \triangleleft

Note that the situation $\mathcal{S}_n = \emptyset$ for $1 \leq n < \bar{n}$ and $\mathcal{S}_n \neq \emptyset$ for $n \geq \bar{n}$ (where \bar{n} is some appropriate integer) might occur and be acceptable. However, for simplicity (e.g., just to have the nested structure (2.9) below), we exclude this situation.

Let $W_n \subseteq \mathbb{R}^{\mathcal{N}(n)}$ be the set of admissible vectors \mathbf{w}_n , obtained by “transferring” the constraints that define \mathcal{S} into the space $\mathbb{R}^{\mathcal{N}(n)}$, i.e.,

$$W_n \triangleq \{ \mathbf{w}_n : \gamma(\cdot, \mathbf{w}_n) \in \mathcal{A}_n \cap \mathcal{S} \}, \quad n = 1, 2, \dots. \quad (2.7)$$

Then, we have

$$\mathcal{S}_n = \{ \gamma(\cdot, \mathbf{w}_n) \in \mathcal{H} : \mathbf{w}_n \in W_n \subseteq \mathbb{R}^{\mathcal{N}(n)} \}, \quad n = 1, 2, \dots \quad (2.8)$$

and, instead of the nested structure (2.4), we have to consider the structure (see Fig. 2.1)

$$\mathcal{S}_1 \subset \mathcal{S}_2 \subset \cdots \subset \mathcal{S}_n \subset \cdots \subset \mathcal{S} \quad (2.9)$$

(we assume that the nestedness property is preserved).

Let us now substitute the FSP functions $\gamma(\cdot, \mathbf{w}_n)$ into the functional F and perform the operations required by F itself (summations, differentiations, integrations, etc.).

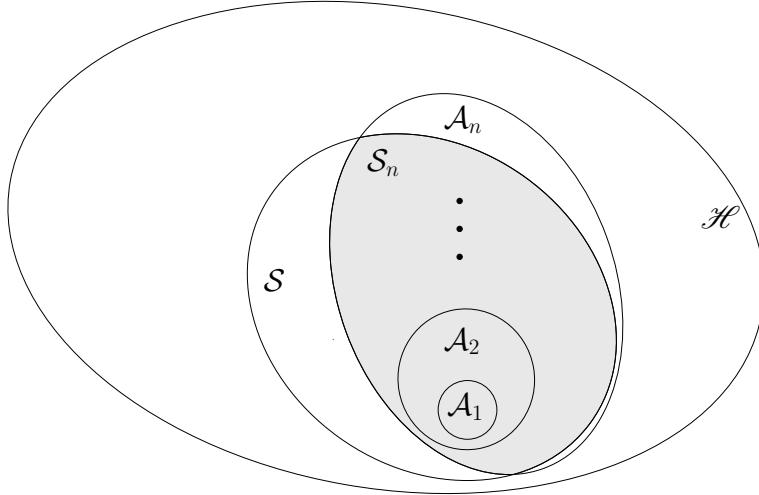


Fig. 2.1 The nested structure of the sequences $\{\mathcal{A}_n\}_{n=1}^{\infty}$ and $\{\mathcal{S}_n\}_{n=1}^{\infty} = \{\mathcal{A}_n \cap \mathcal{S}\}_{n=1}^{\infty}$. The set $\mathcal{S}_n = \mathcal{A}_n \cap \mathcal{S}$ is represented by the gray area. The dimension d of \mathbf{x} is fixed

Evidently, once all such operations have been executed, the functional becomes a function \tilde{F} of the vector \mathbf{w}_n , that is,

$$\tilde{F}(\mathbf{w}_n) \triangleq F[\gamma(\cdot, \mathbf{w}_n)]. \quad (2.10)$$

As the components of the vector \mathbf{w}_n are finitely many $\mathcal{N}(n)$ variables, we have to solve a sequence of “approximating” nonlinear (in general) programming problems. Each of them can be stated as follows (remember that in Assumption 1.1, we have supposed, for simplicity, that all optimization problems have an optimal solution and that this solution is unique).

Problem \mathbf{P}_n . Find the optimal parameter vector

$$\mathbf{w}_n^\circ = \arg \min_{\mathbf{w}_n \in W_n} \tilde{F}(\mathbf{w}_n). \quad (2.11)$$

◇

For the reader’s convenience, in Fig. 2.2 we summarize the various steps that led us to approximate the original functional optimization Problem P by the NLP Problem \mathbf{P}_n .

Since we have assumed the existence and uniqueness of the optimal solution $\gamma_n^\circ = \gamma(\cdot, \mathbf{w}_n^\circ)$ for any problem of the sequence

$$\mathbf{P}_1, \mathbf{P}_2, \dots, \quad (2.12)$$

we obtain the sequence

$$\gamma_1^\circ, \gamma_2^\circ, \dots \quad (2.13)$$

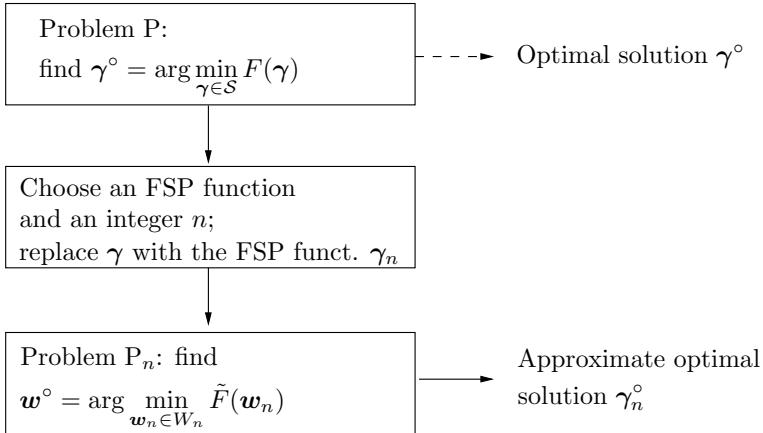


Fig. 2.2 Reduction of the original functional optimization Problem P to the NLP Problem P_n

of optimal solutions. Thanks to the nestedness property of the sequence $\{\mathcal{S}_n\}_{n=1}^{\infty}$, the minimum cost $\tilde{F}(\mathbf{w}_n^\circ)$ is nonincreasing with respect to n , that is,

$$\tilde{F}(\mathbf{w}_1^\circ) \geq \tilde{F}(\mathbf{w}_2^\circ) \geq \dots, \quad (2.14)$$

and, by letting n grow, one may get better and better approximate solutions to Problem P. In practice, one also has the hope of a strict reduction of the minimum cost $\tilde{F}(\mathbf{w}_n^\circ)$ with n , although in general this is problem-dependent.

2.2.2 The Case of Problem PM

If we have to consider $M > 1$ decision functions $\gamma_1(\mathbf{x}_1), \dots, \gamma_M(\mathbf{x}_M)$, we are in the framework of Problem PM stated in Sect. 1.3.1. Then, we introduce M -approximating FSP functions of the form described in Definition 2.1, that is,

$$\gamma_1(\mathbf{x}_1, \mathbf{w}_{1n_1}), \dots, \gamma_M(\mathbf{x}_M, \mathbf{w}_{Mn_M}). \quad (2.15)$$

The families of FSP functions may be or may not be of the same kind. Besides, there is no reason for supposing that the FSP functions have the same model complexities and then the same numbers of parameters.

As we have defined the function γ in a compact form (see (1.8)), we do the same for the M FSP functions and we gather the FSP functions (2.15) into

$$\boldsymbol{\gamma}(\mathbf{x}, \mathbf{w}_n) \triangleq \text{col} [\gamma_1(\mathbf{x}_1, \mathbf{w}_{1n_1}), \dots, \gamma_M(\mathbf{x}_M, \mathbf{w}_{Mn_M})], \quad (2.16)$$

where

$$\mathbf{x} \triangleq \text{col} (\mathbf{x}_1, \dots, \mathbf{x}_M), \quad (2.17)$$

$$\mathbf{w}_n \triangleq \text{col} (\mathbf{w}_{1n_1}, \dots, \mathbf{w}_{Mn_M}), \quad (2.18)$$

and

$$\mathbf{n} \triangleq \text{col} (n_1, \dots, n_M). \quad (2.19)$$

Note that the model complexity of the family of FSP Functions (2.16) is now described by the vector \mathbf{n} whose components are given by the model complexities n_1, \dots, n_M . In order to reduce the functional optimization Problem PM to an NLP problem of the form of Problem P_n , it is useful to describe the “global” complexity of the family of the M FSP functions by a single integer n , likewise made for a single FSP function. What we have to do is just to introduce a vector-valued function

$$\boldsymbol{\nu}: \mathbb{Z}^+ \rightarrow \mathbb{Z}^{+M}, \quad (2.20)$$

so that the model complexities n_1, \dots, n_M of the M FSP functions depend on n (of course, we require them to be nondecreasing functions of n). This becomes clear if we write the function (2.20) making its components explicit. We have

$$\mathbf{n} = \boldsymbol{\nu}(n) = \text{col} (n_1, \dots, n_M) = \text{col} [\nu_1(n), \dots, \nu_M(n)], \quad (2.21)$$

where the functions

$$n_i = \nu_i(n), \quad i = 1, \dots, M, \quad (2.22)$$

describe the dependences of the M model complexities n_i on n . Clearly, the choice of the functions $\nu_i(n)$ cannot be but somewhat arbitrary and dictated by practical considerations. Simple examples of functions $\nu_i(n)$ are shown in Fig. 2.3.

Now, the positive integer n plays the role of *global model complexity of a given set of FSP functions*. Indeed, n enables us to determine

- The model complexity n_i of each FSP function (see (2.22)).
- The numbers $\mathcal{N}_i(n_i)$ of parameters to be optimized in each FSP function and then the dimensions of the vectors \mathbf{w}_{in_i} .
- The total number $\mathcal{N}(n) \triangleq \sum_{i=1}^M \mathcal{N}_i(n_i)$ of parameters.

On such a basis, we can define the following sets of FSP functions (see (2.2)):

$$\mathcal{A}_{in_i} \triangleq \left\{ \gamma_i(\cdot, \mathbf{w}_{in_i}) : \mathbb{R}^{d_i} \rightarrow \mathbb{R}^{m_i}, \mathbf{w}_{in_i} \in \mathbb{R}^{\mathcal{N}(n_i)} \right\}, \quad i = 1, \dots, M. \quad (2.23)$$

Then, we can extend Definition (2.2), that is,

$$\mathcal{A}_{Mn} = \mathcal{A}_{\boldsymbol{\nu}(n)} \triangleq \mathcal{A}_{1n_1} \times \cdots \times \mathcal{A}_{Mn_M}, \quad n = 1, 2, \dots, \quad (2.24)$$

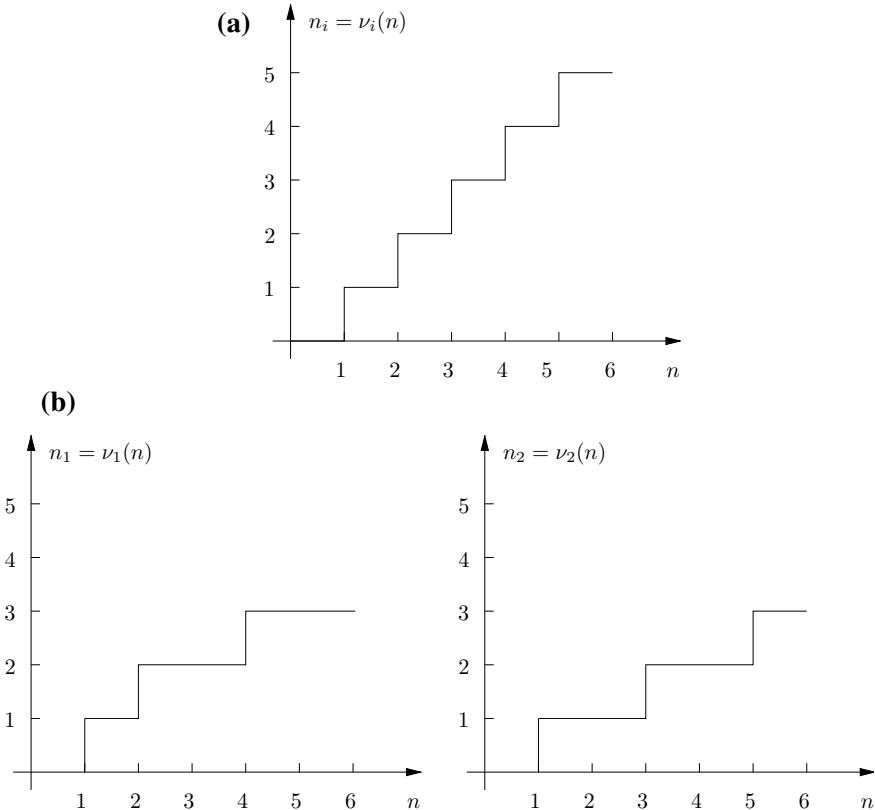


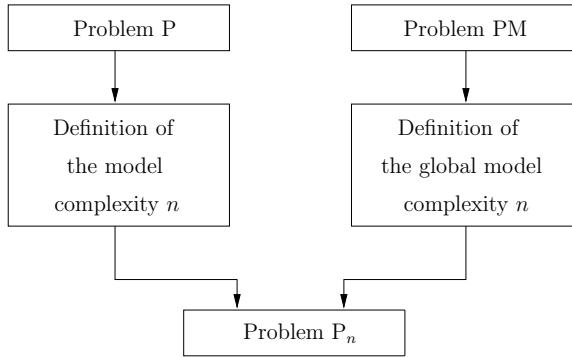
Fig. 2.3 **a** All the model complexities n_i of the FSP functions ($i = 1, \dots, M$) grow together with the global complexity n ; **b** two FSP functions are considered ($M = 2$). In this particular example, when n increases to $n + 1$, only the model complexity of one FSP function increases at a time

where we have written $\mathcal{A}_{Mn} = \mathcal{A}_{\nu(n)}$ as there is no ambiguity, once the function $\mathbf{n} = \nu(n)$ has been fixed. Of course, the growth mechanism must give rise to a sequence $\{\mathcal{A}_{Mn}\}_{n=1}^{\infty}$ provided with the same nested structure as the one described in (2.4).

We have now to take into account the possible presence of constraints on the admissible decision functions. We remind the reader that, in general, the DMs' decision functions may be jointly constrained to belong to an admissible set \mathcal{S}_M (see (1.7)). Accordingly, for each n , the set of the admissible FSP functions can be defined as follows (compare with (2.8)):

$$\mathcal{S}_{Mn} \triangleq \{\gamma(\cdot, \mathbf{w}_n) \in \mathcal{H}_1 \times \cdots \times \mathcal{H}_M : \mathbf{w}_n \in W_n \subseteq \mathbb{R}^{\mathcal{N}(n)}\}, \quad n = 1, 2, \dots, \quad (2.25)$$

Fig. 2.4 Reduction of the functional Problems P and PM to the NLP Problem P_n



where W_n is the set of admissible vectors \mathbf{w}_n obtained by “transferring” the constraints that define \mathcal{S}_M into the space $\mathbb{R}^{\mathcal{N}(n)}$ (compare with (2.7)). Summing it up, suitably extending Assumption 2.3, $\{\mathcal{S}_{Mn}\}_{n=1}^\infty$ is a nested sequence, that is,

$$\mathcal{S}_{M1} \subset \mathcal{S}_{M2} \subset \cdots \subset \mathcal{S}_{Mn} \subset \cdots \subset \mathcal{S}_M. \quad (2.26)$$

In order to reduce Problem PM to an NLP problem, we now apply the ERIM in the same way as we did for Problem P. This means that we replace Function (1.8) in the functional $F(\gamma)$ with Function (2.16) (which we now write $\gamma(\cdot, \mathbf{w}_n)$) and we perform the operations required to compute the functional itself. This substitution reduces the functional F to the following function of the vector \mathbf{w}_n (see (2.10)):

$$\tilde{F}(\mathbf{w}_n) \triangleq F[\gamma(\cdot, \mathbf{w}_n)]. \quad (2.27)$$

As in the case of Problem P, for $n = 1, 2, \dots$, we have to solve a sequence of approximating *nonlinear (in general) programming problems*. Each of them can be stated as follows.

Problem P_n . Find the optimal parameter vector

$$\mathbf{w}_n^\circ = \arg \min_{\mathbf{w}_n \in W_n} \tilde{F}(\mathbf{w}_n). \quad (2.28)$$

△

Note that Problem P_n , derived from Problem P (see (2.11)), and Problem P_n , derived from Problem PM (see (2.28)), take an identical form. This is a significant point, since the techniques that can be used to solve the NLP problems approximating Problem P and Problem PM are the same. In Fig. 2.4, we enhance the reduction of the two IDO problems to Problem P_n .

2.3 Solution of the Nonlinear Programming Problem P_n

For a brief overview of how to solve Problem P_n , we anticipate in this section some concepts that will be dealt with in more detail in Chap. 5. Solving the NLP Problem P_n is in general a very hard task. To begin with, deriving the analytical expression of the function $\tilde{F}(\mathbf{w}_n)$ is usually very difficult. One of the main reasons stems from the fact that, as discussed in Sect. 1.3.1, most IDO problems considered in the book are stated within stochastic frameworks. In this regard, some examples described in Sect. 1.5 do not address stochastic optimal decision problems by chance (see the problems stated in Sects. 1.5.3, 1.5.4, and 1.5.5).

To be more specific, one generally has to solve IDO problems in which the cost functional F is expressed as the average of another functional J that depends on a random vector z and M decision functions $\gamma_1, \dots, \gamma_M$ to be optimized. As in (1.8), we write these functions in a compact form, that is, we let $\boldsymbol{\gamma} \triangleq \text{col}(\gamma_1, \dots, \gamma_M)$. In Assumption 1.2, the probability density function $p(z)$ has been assumed to exist and to be known. We have found averages of the form (1.9) in the stochastic examples of Sect. 1.5, where it has been possible to write the functionals in the following common form (see (1.27), (1.32), and (1.35)):

$$F(\boldsymbol{\gamma}) = \underset{z}{\mathbb{E}} J(\boldsymbol{\gamma}, z). \quad (2.29)$$

Now, considering Problem P_n , the cost to be minimized is given by $\tilde{F}(\mathbf{w}_n)$. Letting $\tilde{J}(\mathbf{w}_n, z) \triangleq J(\boldsymbol{\gamma}_n, z)$, we have (see (2.27) and (2.29))

$$\tilde{F}(\mathbf{w}_n) = F[\boldsymbol{\gamma}(\cdot, \mathbf{w}_n)] = F(\boldsymbol{\gamma}_n) = \underset{z}{\mathbb{E}} J(\boldsymbol{\gamma}_n, z) = \underset{z}{\mathbb{E}} \tilde{J}(\mathbf{w}_n, z).$$

Then, a preliminary choice is whether to compute

$$\tilde{F}(\mathbf{w}_n) = \underset{z}{\mathbb{E}} \tilde{J}(\mathbf{w}_n, z) \quad (2.30)$$

ignoring that \tilde{F} is obtained via an expectation or to try to exploit the possible advantages arising from the structure (2.29), where the averaging operation is left in an explicit form. We state again Problem P_n by highlighting the presence of the averaging operator.

Problem P_n . *Find the optimal parameter vector*

$$\mathbf{w}_n^o = \arg \min_{\mathbf{w}_n \in W_n} \underset{z}{\mathbb{E}} \tilde{J}(\mathbf{w}_n, z). \quad (2.31)$$

△

For clarity, in the sequel we shall distinguish between those choices by specifying *Problem P_n in the form (2.11)* or *Problem P_n in the form (2.31)*. If one opts for the *first choice*, that is, for the solution of Problem P_n in the form (2.11), then one has to resort to NLP algorithms. Thus, one has to decide between *direct search methods*

(which use only the values of $\tilde{F}(\mathbf{w}_n)$) and *gradient-based methods*. We discard the use of the Hessian matrix of $\tilde{F}(\mathbf{w}_n)$ as, in general, it is too heavy to compute either analytically or numerically. Direct methods take on significant importance because, in general, the expected value of J cannot be computed but numerically (note also that the dimension of the random vector z is usually very large in the application problems presented in the book). Then, a fortiori, the numerical computation of the gradient $\nabla \tilde{F}(\mathbf{w}_n)$ is difficult. The following assumption plays a key role.

Assumption 2.4 The cost function $\tilde{J}(\mathbf{w}_n, z)$ can be computed exactly (or “in a sufficiently accurate way”) for all values of \mathbf{w}_n and z . \triangleleft

Of course, in practice the expression “in a sufficiently accurate way” has to be evaluated each time.

If Assumption 2.4 holds, then Monte Carlo (MC) and quasi-Monte Carlo (quasi-MC) methods are efficient techniques for computing estimates of the expected values of functions (this issue will be exposed in Sect. 5.2). Both techniques rely on the computation of the estimates

$$\tilde{F}(\mathbf{w}_n) = \underset{z}{\mathbb{E}} \tilde{J}(\mathbf{w}_n, z) \simeq \frac{1}{L} \sum_{l=1}^L \tilde{J}(\mathbf{w}_n, z_l), \quad (2.32)$$

where the cost $\tilde{J}(\mathbf{w}_n, z)$ is evaluated in correspondence of the L vectors z_1, \dots, z_L . In the Monte Carlo approach, such vectors are *independent identically distributed* (i.i.d.) samples chosen according to the probability density $p(z)$. In the quasi-Monte Carlo framework, they are generated by suitable deterministic algorithms. As we shall see in Chap. 5 and quite intuitively, the accuracy of Estimate (2.32) depends on the number L of samples and on the smoothness of the function \tilde{J} . We also start emphasizing that the evaluation of the function $\tilde{J}(\mathbf{w}_n, z)$ at the nodes of a regular grid constructed in the domain of z should be excluded for high dimensions of z , owing to the danger of incurring the curse of dimensionality. This danger is mitigated by the use of Monte Carlo or quasi-Monte Carlo techniques. Once the expected cost $\tilde{F}(\mathbf{w}_n)$ has been determined via (2.32), Problem P_n can be solved by a suitable direct NLP method.

Suppose now that we want to solve Problem P_n by a gradient-based method instead. Clearly, the following assumption is fundamental.

Assumption 2.5 For every value of z , the cost function $\tilde{J}(\mathbf{w}_n, z)$ is of class \mathcal{C}^1 with respect to \mathbf{w}_n and, for all values of \mathbf{w}_n and z , the gradient $\nabla_{\mathbf{w}_n} \tilde{J}(\mathbf{w}_n, z)$ can be computed exactly (or “in a sufficiently accurate way”). \triangleleft

The expression “in a sufficiently accurate way” has the same meaning as for Assumption 2.4. In particular, if the gradient $\nabla \tilde{F}(\mathbf{w}_n)$ cannot be computed analytically, it means that its components can be determined by some finite-difference techniques and one must have reasons to believe that the corresponding approximations are sufficiently accurate.

Let us now address the possible advantages that can be drawn from the particular form of (2.31), i.e., from leaving the averaging operation in an explicit form. This corresponds to the *second choice* we mentioned before.

If Assumption 2.5 is verified, then, under suitable additional hypotheses specified in Theorem 5.2, $\tilde{F}(\mathbf{w}_n)$ is also of class \mathcal{C}^1 . Then, we can write

$$\nabla \tilde{F}(\mathbf{w}_n) = \nabla_{\mathbf{w}_n} \int \tilde{J}(\mathbf{w}_n, z) p(z) dz . \quad (2.33)$$

Now, if the assumptions of Theorem 5.2 are verified, then the operations of computing the gradient and the integral in (2.33) can be interchanged and $\nabla \tilde{F}(\mathbf{w}_n)$ can be written in the form

$$\nabla \tilde{F}(\mathbf{w}_n) = \int \nabla_{\mathbf{w}_n} \tilde{J}(\mathbf{w}_n, z) p(z) dz . \quad (2.34)$$

$\nabla \tilde{F}(\mathbf{w}_n)$ is expressed either in the form (2.33) or in the form (2.34). This implies again the necessity of computing numerically the integrals by Monte Carlo or quasi-Monte Carlo techniques, since, in general, their analytical computation is impossible and resorting to a regular grid in the domain of z is unfeasible in high-dimensional settings. The expression (2.34) is particularly interesting as it implies that the realization $\nabla_{\mathbf{w}_n} \tilde{J}(\mathbf{w}_n, z)$ (called *stochastic gradient* for its dependence on the random vector z) yields an *unbiased* estimate of the *deterministic gradient* $\nabla \tilde{F}(\mathbf{w}_n)$. Indeed, we have

$$\mathbb{E}_{\mathbf{z}} \nabla_{\mathbf{w}_n} \tilde{J}(\mathbf{w}_n, z) = \nabla \tilde{F}(\mathbf{w}_n) . \quad (2.35)$$

This is particularly useful in “training” (i.e., in optimizing) multilayer feedforward neural networks (which are a typical choice for FSP functions). The usual deterministic steepest descent algorithm for unconstrained optimization is given by

$$\mathbf{w}_n(k+1) = \mathbf{w}_n(k) - \alpha_k \nabla_{\mathbf{w}_n} \mathbb{E}_{\mathbf{z}} \tilde{J}[(\mathbf{w}_n(k), z)] , \quad k = 0, 1, \dots , \quad (2.36)$$

where k is the iteration count, $\mathbf{w}_n(0)$ an initial “guess”, and α_k a positive stepsize, which can be either an a priori fixed constant (i.e., $\alpha_k = \alpha$) or can be determined by some one-dimensional minimization technique.

Replacing the deterministic gradient $\nabla \tilde{F}(\mathbf{w}_n)$ in (2.36) with its unbiased estimate provided by the stochastic gradient yields the *stochastic gradient algorithm*, widely used throughout the book,

$$\mathbf{w}_n(k+1) = \mathbf{w}_n(k) - \alpha_k \nabla_{\mathbf{w}_n} \tilde{J}[\mathbf{w}_n(k), z(k)] , \quad k = 0, 1, \dots , \quad (2.37)$$

where the index $k = 0, 1, \dots$ denotes both the iteration of the recursive procedure and the time at which the sampled values $z(k)$ of the vector z are independently generated on the basis of the probability density function $p(z)$. The stepsize α_k has to decrease suitably according to some specific rule, in order to achieve convergence. Clearly, not every iteration step of Algorithm (2.37) is efficient in itself, as it may

not ensure a decrease of the cost $\underset{z}{\mathbb{E}} \tilde{J}[\mathbf{w}_n(k), z]$. However, suitable conditions on the function \tilde{J} , the rate of decrease of α_k , and the sequence $\{z(k)\}_{k=0}^{\infty}$ may lead the algorithm to converge (in a probabilistic sense) to an optimal solution \mathbf{w}_n^* in the unconstrained case $W_n = \mathbb{R}^{N(n)}$. Sufficient conditions for convergence, as well as some more details for the search for minima, are given in Sect. 5.3. Search for minima by Monte Carlo and quasi-Monte Carlo techniques is described in Sect. 5.4.

2.4 Optimizing FSP Functions

In Assumption 1.1, we have supposed that Problem P and, in general, all optimization problems addressed in the book (hence Problem P_n , too) can be stated in terms of search for minima. We have also assumed that such minima are uniquely achieved.

For the reader's convenience, let us emphasize separately the existence and uniqueness of the minimum points for Problems P and P_n .

Assumption 2.6 Problem P has a unique optimal solution γ^* , i.e., the infimum of the functional F is a minimum, and is attained only in correspondence of γ^* . \triangleleft

Assumption 2.7 For any $n \in \mathbb{Z}^+$, an optimal solution \mathbf{w}_n^* to Problem P_n exists and is unique, i.e., the infimum of the function $\tilde{F}(\mathbf{w}_n)$ is a minimum, and is attained only in correspondence of \mathbf{w}_n^* . \triangleleft

On the basis of Assumptions 2.6 and 2.7, we introduce the following notation²:

$$\begin{aligned} F^* &\triangleq F(\gamma^*) = \min_{\gamma \in \mathcal{S}} F(\gamma), \\ F_n^* &\triangleq F[\gamma(\cdot, \mathbf{w}_n^*)] = \min_{\mathbf{w}_n \in W_n} \tilde{F}(\mathbf{w}_n), \\ \gamma_n^*(\cdot) &\triangleq \gamma(\cdot, \mathbf{w}_n^*). \end{aligned}$$

We are now in a position to clarify the meaning of the expression “Problems P_n approximate Problem P.” More specifically, we can formally state that a sequence of Problems P_n approximates Problem P if both the following conditions hold true:

- (1) The sequence $\{\gamma_n^*\}_{n=1}^{\infty}$ has the limit function $\gamma^* \in \mathcal{S}$, i.e.,

$$\lim_{n \rightarrow \infty} \|\gamma_n^* - \gamma^*\|_{\mathcal{H}} = 0. \quad (2.38)$$

Note that, by virtue of Assumption 2.2, the linear space \mathcal{H} has now become a normed linear space, endowed with the norm $\|\cdot\|_{\mathcal{H}}$.

²The reader should now understand why Remark on Notation 2.1 is useful to simplify the notation; see Point 3 in the remark.

(2) The sequence $\{\gamma_n^\circ\}_{n=1}^\infty$ is such that

$$\lim_{n \rightarrow \infty} F_n^\circ = F^\circ. \quad (2.39)$$

In the literature, sequences verifying Convergence (2.39) are called *minimizing sequences* (see, e.g., [12, p. 193]). Imposing both (2.38) and (2.39) is related to the concept of *epi-convergence*. We refer the interested reader to [4, 8], for example. It is important to remark that Convergence (2.38) does not always imply Convergence (2.39) and vice versa. So, just out of curiosity, let us consider the following example where (2.39) does not imply (2.38).

Example 2.2 We describe a situation in which (2.39) holds but an optimal solution γ° does not even exist in the set \mathcal{S} of admissible solutions to Problem P (see [12, p. 194]). Toward this end, we consider the following instance of Problem P.

Let \mathcal{H} be the space $\mathcal{C}^1([-1, 1], \mathbb{R})$ of continuously differentiable functions on $[-1, 1]$ equipped with the supremum norm, $\mathcal{S} = \{\gamma \in \mathcal{C}^1([-1, 1], \mathbb{R}) : \gamma(-1) = -1, \gamma(1) = 1\}$, and, for every $\gamma \in \mathcal{S}$,

$$F(\gamma) = \int_{-1}^1 x^2 \gamma'(x)^2 dx,$$

where γ' denotes the first-order derivative of γ . We have $F(\gamma) > 0$ for any admissible γ and $\inf_{\gamma \in \mathcal{S}} F(\gamma) = 0$.

Let us consider the sequence $\{\gamma_n\}_{n=1}^\infty$ defined as

$$\gamma_n(x) \triangleq \frac{\arctan(n x)}{\arctan n} \xrightarrow{n \rightarrow \infty} \hat{\gamma}(x) = \begin{cases} 1 & , x > 0 \\ 0 & , x = 0 \\ -1 & , x < 0 \end{cases}$$

where the convergence above is merely pointwise and not in $\mathcal{C}^1([-1, 1], \mathbb{R})$ with the supremum norm.

In Fig. 2.5, the integrand function $x^2 \gamma'_n(x)^2$ is depicted for some values of n and the subtended areas are pointed out.

We have

$$\begin{aligned} F(\gamma_n) &= \int_{-1}^1 \frac{n^2 x^2}{(\arctan n)^2 (1 + n^2 x^2)^2} dx \\ &< \frac{1}{(\arctan n)^2} \int_{-1}^1 \frac{dx}{1 + n^2 x^2} = \frac{2}{n \arctan n}. \end{aligned}$$

Hence, $\lim_{n \rightarrow \infty} F(\gamma_n) = 0 = \inf_{\gamma \in \mathcal{S}} F(\gamma) = 0$. However, for $n \rightarrow \infty$, the limit function $\hat{\gamma}$ of the sequence $\{\gamma_n\}_{n=1}^\infty$ belongs neither to \mathcal{S} nor to $\mathcal{C}^1([-1, 1], \mathbb{R})$ as $\hat{\gamma}$ is not continuous in $[-1, 1]$. Then, Convergence (2.39) is verified but (2.38) is not and the sequence is not a P-optimizing one, according to Definition 2.3. \triangleleft

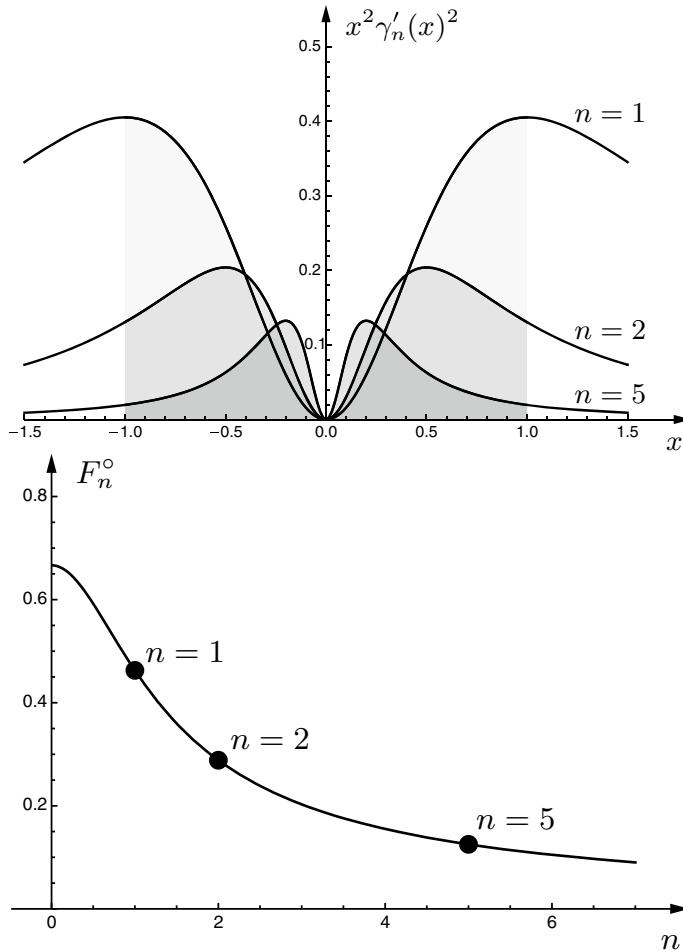


Fig. 2.5 The integrand function $x^2 \gamma_n'(x)^2$, where $\gamma_n(x) = \frac{\arctan nx}{\arctan n}$, for some values of n , and the corresponding integrals $\int_{-1}^1 x^2 \gamma_n'(x)^2 dx$

In the sequel, we avoid the delicate and quite technical issues of epi-convergence and we define the following class of sequences of FSP functions.

Definition 2.3 Let Problem P and Problems P_n verify Assumptions 2.6 and 2.7, respectively. If both (2.38) and (2.39) hold, then $\{\gamma_n^\circ\}_{n=1}^\infty$ is defined as a “P-optimizing sequence.” The FSP functions belonging to the sequence $\{\gamma_n^\circ\}_{n=1}^\infty$ are called “P-optimizing FSP functions.” \triangleleft

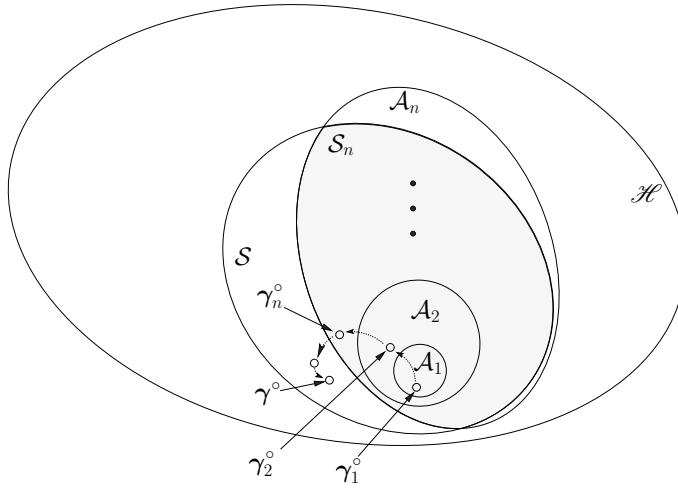


Fig. 2.6 The nested structure of the sequences $\{A_n\}_{n=1}^{\infty}$ and $\{S_n\}_{n=1}^{\infty} = \{A_n \cap \mathcal{S}\}_{n=1}^{\infty}$, with a symbolic example of an optimizing sequence $\{\gamma_n^{\circ}\}_{n=1}^{\infty}$. The dimension d of x is fixed

When the functional F is continuous (of course, in the norm $\|\cdot\|_{\mathcal{H}}$), Convergence (2.38) implies Convergence (2.39). For simplicity, in the whole book we shall suppose that (2.38) implies (2.39). Let us point out explicitly this hypothesis.

Assumption 2.8 Convergence (2.38) implies Convergence (2.39). \triangleleft

Whenever there is no risk of ambiguity, we shall write merely “optimizing sequence” and “optimizing FSP functions,” without including explicitly in the notation the Problem P to which they refer. An optimizing sequence is shown symbolically in Fig. 2.6.

It is important to note that all the aforesaid assumptions and properties are strictly related to the particular instance of Problem P that we are addressing. Indeed, we recall that an instance of Problem P is specified by the normed space \mathcal{H} , the set \mathcal{S} of admissible solutions, and the cost functional F , that is, by the triple $(\mathcal{H}, \mathcal{S}, F)$.

2.5 Polynomially Complex Optimizing FSP Functions

In describing the ERIM, the dependence on the dimension d of the argument vector x (see (2.16)) on which the decision functions depend, i.e., their number d of variables, has to be considered explicitly. In this section, we emphasize such a dependence by specifying “ d ” as a superscript and thus by writing

$$F^d : \mathcal{S}^d \rightarrow \mathbb{R} \quad (2.40)$$

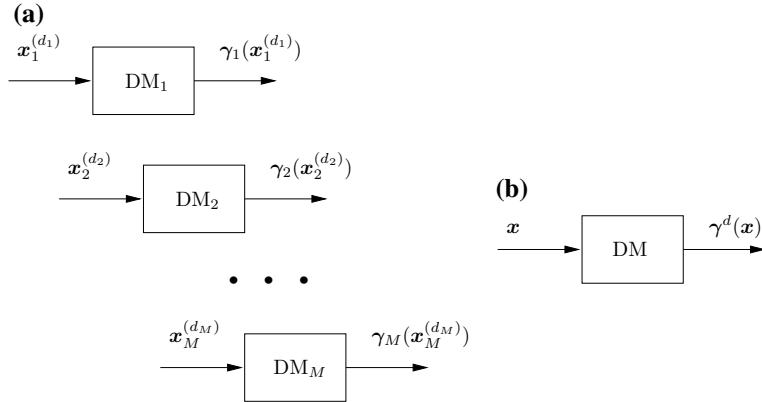


Fig. 2.7 **a** The M decision-makers of Problem PM are shown separately. **b** The global input vector \boldsymbol{x} and the decision function $\gamma^d(\boldsymbol{x})$ are reported

and

$$\gamma^d : D^d \rightarrow C \quad (2.41)$$

instead of (1.3) and (1.4), respectively.

For the sake of notational simplicity and without loss of generality, we assume that d simply increases as follows: $d = 1, 2, \dots$. There is no difficulty in considering problems in which the dimension d takes on nonconsecutive integer values d_1, d_2, \dots . Indeed, as we shall see in the next section, nonconsecutive integer values are more the rule than the exception. As, in general, we have to deal mostly with Problems PM, it is important to further clarify the meaning of the dimension d , besides what already said in Sects. 1.3.1 and 2.2.2. To this end, in Fig. 2.7 we show both the M decision-makers DM_1, \dots, DM_M considered separately (Fig. 2.7a) and in aggregate form (Fig. 2.7b). In Fig. 2.7a, the input vector $\boldsymbol{x}_i^{(d_i)}$ and the decision function $\gamma_i(\boldsymbol{x}_i^{(d_i)})$ of each DM_i are pointed out; the dimension d_i of the vector $\boldsymbol{x}_i^{(d_i)}$ is also highlighted. In Fig. 2.7b, the global input vector \boldsymbol{x} (see (2.17)) and the decision function $\gamma^d(\boldsymbol{x})$ (see (1.8)) are reported. Clearly, $d = d_1 + \dots + d_M$. This dimension will be considered from now on.

Therefore, we reduce the sequence of Problems PM to a sequence of Problems P, where d now appears in superscript. Such a sequence is given by

$$\{P^d : d = 1, 2, \dots\}. \quad (2.42)$$

Each Problem P^d can be stated as follows.

Problem \mathbf{P}^d . Given Functional (2.40) and Decision Functions (2.41), find the optimal decision function

$$\gamma^{d\circ} = \arg \min_{\gamma^d \in \mathcal{S}^d} F^d(\gamma^d), \quad d = 1, 2, \dots . \quad (2.43)$$

△

In the next two sections, we consider contexts in which the dimension d increases. As we are interested in solving such problems approximately by the ERIM, we are interested in the rate at which the model complexity n of the FSP functions has to increase in order to keep the approximation error below a given threshold ε .

2.5.1 The Growth of the Dimension d

For any given dimension $d = 1, 2, \dots$, let us consider the sequence

$$\{\mathbf{P}_n^d\}_{n=1}^{\infty} \quad (2.44)$$

of NLP problems that approximates the original IDO Problem \mathbf{P}^d by FSP functions with increasing model complexities. Let us also suppose that the sequences

$$\{\gamma_n^{d\circ}\}_{n=1}^{\infty}, \quad d = 1, 2, \dots . \quad (2.45)$$

verify the conditions required by Definition 2.3, so, for any given d , they are \mathbf{P}^d -optimizing sequences. Now, let us introduce the maximum acceptable approximation error $\varepsilon > 0$ and, in the set of all optimal FSP functions belonging to Sequences (2.45), for any d let us consider the indices n and the corresponding functions $\gamma_n^{d\circ}$ that verify the inequality

$$\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d} \leq \varepsilon . \quad (2.46)$$

It is worth noting that most IDO problems in the book are characterized by large values of d . It goes without saying that, for a given sequence of IDO problems, establishing a relationship among the following three quantities is a fundamental issue: (i) the model complexity n , (ii) the dimension d , and (iii) the maximum allowed approximation error ε . More specifically, it is important to understand the rate at which the model complexities of the FSP functions have to grow when d increases in order to keep on verifying Inequality (2.46).

From a practical point of view, there are several problems – coming from the physical, economic, and other areas – that are characterized by a possible increase of the dimension d . Consider, for example, the optimal water reservoir management problems described in Sect. 7.7. In Figs. 7.8 and 7.9, 7-reservoir and 10-reservoir configurations are addressed, respectively. Of course, other configurations made of the same number of reservoirs may occur. The two management problems (one

for each configuration) have been modeled as T -stage stochastic optimal control problems of the form of Problem 1.4. In Chap. 7, we shall present the solutions of such problems by the ERIM and by dynamic programming (DP). As FSP functions, we have chosen neural networks for approximating the control functions (1.23). Properly defined integers describe the global model complexities of the chains of networks. The input vectors to the networks are given by the state vectors, whose number of components is $s = 10$ in the 7-reservoir configuration and $s = 30$ in the 10-reservoir configuration. Of course, the rate at which the global model complexities of the chains of networks have to increase with the dimension $d = s \cdot T$ of Problem P^d in order to keep the approximation errors below a desired threshold ε is an important design information.

Similar considerations can be repeated for the freeway traffic optimal control problem addressed in Sect. 8.6.2 (see Fig. 8.8). In this configuration, the freeway stretch is divided into $D = 30$ sections. The traffic controller, implemented by a chain of T neural networks (each acting at the temporal stage $t = 0, 1, \dots, T - 1$), receives two measurements from each freeway section, i.e., the mean traffic speed and the traffic density, which are components of the state vector. The controller also receives the measurements of queue lengths that may be present on the on-ramps. In the example presented in Chap. 8, the measurements are affected by disturbances. Here our purpose is to clarify the meaning of the dimension d in that context. So, we consider the state as perfectly measurable. Since the number of sections in which on/off ramps are present is usually small with respect to the number D of sections, we disregard the queue lengths in computing the dimension of the state vector. Then, the dimension of this vector is $s \simeq 2D$. Here again, we are interested in the rate at which the global model complexity n of the chain of FSP functions increases with $d \simeq 2D \cdot T$.

Among other optimal control problems characterized by the possibility that d increases to arbitrarily large values, we mention the optimal reheating problem of a slab [3]. This is a typical example of control of distributed-parameter systems whose models are infinite-dimensional dynamic systems. In the example dealt with in [3], the state vector has an infinite number of components given by the temperatures at any point of the slab. The heat supplied to a side of the slab by an external source plays the role of the control variable. The preliminary step to solve the problem consists in reducing the partial differential equation model to an ordinary differential equation one. Such a reduction is obtained by discretizing the slab's geometrical domain via the use of some suitable mesh giving a number D of nodes. The temperatures of the slab at these nodes become the D components of the state of a finite-dimensional differential dynamic system. A further discretization with respect to time yields a finite-dimensional discrete-time dynamic system. The system is controlled by a chain of T FSP functions (like in the previous two examples) with a global model complexity n . Of course, practical reasons prevent the controller from measuring the temperatures of all the nodes of the mesh. Suppose that only a subset of $s < D$ nodes is accessible for measurements and let s be proportional to D . Then, a vector with s components constitutes the control input of each FSP function. Once again, the rate at which n increases with d is an important issue (so, a suboptimal controller

is obtained). Note that the growth of D (hence of s) may be caused by an increase of the dimension of the slab (the accuracy induced by the meshes being equal) or by the use of more refined meshes (the slab's dimension being equal). In both cases, $d = s \cdot T$.

The growth of the dimension d of the input vector is also an important issue in image processing, pattern recognition, etc., since d is strictly related, for example, with the accuracy by which the image is presented to the vision computing device.

Of course, there are problems in which an increase in the state dimension s is meaningless. Consider, for example, the manoeuvre of the robot in the plane addressed in Sect. 7.8.1 (see Fig. 7.14). Its state dimension is $s = 6$. If we want to study the problem of the motion of the robot in the three-dimensional space, the state dimension increases to $s = 12$. After that, for the same robot structure, no other values of s are obviously allowed. In the example shown in Fig. 7.14, the dimension d is then given by $d = 6T$, where T is the number of decision stages.

2.5.2 Polynomial and Exponential Growths of the Model Complexity n with the Dimension d

For each Problem P^d in Sequence (2.42), let us assume the sequence $\{\gamma_n^{d_o}\}_{n=1}^\infty$ to be P^d -optimizing. In investigating the rate of growth of the model complexity n with the dimension d , we follow the typical approach adopted by *complexity theory*, which is based on a clean cut between *polynomial* and *exponential rates*. More specifically, we contrast a *polynomial upper bound* on the rate of growth of the error $\|\gamma_n^{d_o} - \gamma^{d_o}\|_{\mathcal{H}^d}$ with respect to d with an *exponential lower bound* on the same quantity.

It has to be taken into account that the important issue is the moderate growth not merely of the model complexity n but of the number of parameters in the FSP functions, i.e., the number $\mathcal{N}(n)$ of components of the vector \mathbf{w}_n . Thus, the distinction described above and detailed in the following is meaningful if the function $\mathcal{N} : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$ (see (2.1)) grows at most polynomially with the model complexity n . In the following, to avoid pathological cases (such as no increase in the number of parameters with n), we require also that it grows at least linearly with n (otherwise n itself would not be very meaningful as model complexity). We emphasize this as follows.

Assumption 2.9 The function $\mathcal{N} : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$ defined in (2.1) grows at least linearly and at most polynomially. \triangleleft

In the sequel and also in other chapters of the book, we shall use the “big-O” and other similar notations. It may be useful to recall their meanings formally [32].

Remark 2.2 For $f, g : \mathbb{Z}^+ \rightarrow \mathbb{R}$, we write $f = O(g)$ (“big O”-notation) when there exist two constants $c > 0$ and $M > 0$ such that, for every $n \geq M$, one has $|f(n)| \leq$

$c |g(n)|$. We write $f = \Omega(g)$ if and only if $g = O(f)$, and $f = \Theta(g)$ if and only if both $f = O(g)$ and $g = O(f)$ hold. Some authors use the notation $f \asymp g$ instead of $f = \Theta(g)$. \triangleleft

In the book, we will often consider functions f depending on the ratio $1/n$, for which the requirement $n \geq M$ can be more naturally formulated as $1/n \leq 1/M$. Moreover, the “big-O” and related notations will be applied sometimes to functions depending also on additional variables, such as the “dimension” of a problem. In such situations, the notations above have to be interpreted by fixing all the variables apart from one (typically, the model complexity n). It is important to stress the fact that, in this interpretation, in the above-reported definitions the “constants” $c > 0$ and $M > 0$ may depend on “hidden” variables (i.e., the ones held fixed). Sometimes, we will additionally require that they are *absolute constants*, i.e., constant with respect to all quantities involved, each time, in the analysis. In this connection, it has to be emphasized that in the literature on the approximation of functions and on IDO the dependence on the number d of variables may be cryptic. Focusing, e.g., on the “constant” c , this means that the approximation errors are bounded from above by quantities that are constant with respect to n but *do* depend on d and the manner of dependence is not specified (see, e.g., [5–7, 9, 10, 14, 24, 25, 36]). Most available upper bounds take on the factorized form

$$\xi(d)\kappa(n).$$

In part of the literature (see, e.g., [6]), the terms depending on d are referred to as “constants” since these papers focus on the model complexity and assume a fixed value for the dimension d . Often, such estimates are formulated as bounds of the form $O(\kappa(n))$, so the dependence on d is hidden in the “big-O” notation. However, it has been shown that such “constants” may actually grow at an exponential rate in d [33, 35].

As remarked in [40], in general, the dependence of the approximation errors on the input dimension d (i.e., the function $\xi(d)$) is much harder to estimate than the dependence on the model complexity n . Not many such estimates are available. The role of d is considered explicitly in the so-called *information-based complexity* (see [39–41]) and it has been only quite recently investigated in the context of function approximation and IDO by neural networks [2, 28, 37].

Case 1. Polynomial growth of n with d

First, suppose that the error $\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d}$ is bounded from above as follows.

Assumption 2.10 There exist constants $p, q, c \in \mathbb{R}$, $p \geq 0, q, c > 0$ such that

$$\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d} \leq c \frac{d^p}{n^q}, \quad d = 1, 2, \dots, n = 1, 2, \dots. \quad (2.47)$$

\triangleleft

Assumption 2.10 is related to (and stronger than) the following one: there exist constants $p, q \in \mathbb{R}$, $p \geq 0, q > 0$ such that

$$\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d} = O(d^p/n^q), \quad (2.48)$$

since (2.47) is expressed in terms of an absolute constant.

In order to have an error $\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d}$ not larger than ε , what is required by the following assumption is essential.

Assumption 2.11 For every $\varepsilon > 0$ there exist $p, q, c \in \mathbb{R}$, $p \geq 0, q, c > 0$ such that the model complexities n of the optimal FSP functions $\gamma_n^{d\circ}$ verify the inequalities

$$c \frac{d^p}{n^q} \leq \varepsilon, \quad d = 1, 2, \dots, n = 1, 2, \dots. \quad (2.49)$$

△

Equivalently, one can say that

$$n \geq \left(\frac{c}{\varepsilon}\right)^{1/q} d^{p/q}. \quad (2.50)$$

Assumptions 2.10 and 2.11 are summarized by the two inequalities (see (2.47) and (2.49))

$$\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d} \leq c \frac{d^p}{n^q} \leq \varepsilon. \quad (2.51)$$

Now, we denote by $n^\circ(d, \varepsilon)$ the *minimum* model complexity of $\gamma_n^{d\circ}$ that verifies (2.51) for given values of the dimension d and the maximum allowed error ε . This minimum model complexity is given by

$$n^\circ(d, \varepsilon) \triangleq \left\lceil \left(\frac{c}{\varepsilon}\right)^{1/q} d^{p/q} \right\rceil, \quad (2.52)$$

where, for every $a \in \mathbb{R}$, $\lceil a \rceil$ denotes the smallest integer that is not smaller than a . This means that, when the dimension d increases, the condition

$$n \geq n^\circ(d, \varepsilon) \quad (2.53)$$

is *sufficient* to guarantee an error not larger than a given ε . By (2.52), the model complexity $n^\circ(d, \varepsilon)$ has to grow at most as a power of d .

From a theoretical point of view, the classical approach of complexity theory (i.e., the dichotomy between polynomial dependence and exponential dependence) leads us to say that (2.52) provides a *favorable rate of approximate optimization*. Of course, from a practical point of view, when the exponent p/q of d is very large, the minimum model complexity defined by (2.52), although being polynomial, may be unacceptably fast growing with d .

All these considerations bring us to define the following class of P^d -optimizing sequences of FSP functions.

Definition 2.4 Let us consider a sequence (2.42) of Problems P^d and a maximum allowed error $\varepsilon > 0$. Let Assumptions 2.6, 2.7, 2.10, and 2.11 be verified. A sequence of P^d -optimizing FSP functions

$$\{\gamma_n^{d\circ}\}_{n=1}^{\infty}, \quad (2.54)$$

where the model complexities are such that $n \geq n^\circ(d, \varepsilon)$, is defined as “ ε -polynomially complex P^d -optimizing sequence” (ε -pc P^d -optimizing sequence). The FSP functions making up the sequence (2.54) are called “ ε -pc P^d -optimizing FSP functions.” \triangleleft

From now on, whenever there will be no risk of ambiguity, we drop ε and simply use the terminology “pc-optimizing sequences” and “pc-optimizing FSP functions.”

Clearly, one may wonder why we do not define as “ ε -pc P^d -optimizing sequence” the sequence (2.54) itself with $n = n^\circ(d, \varepsilon)$ instead of $n \geq n^\circ(d, \varepsilon)$. This way, we might consider the “simplest” of all sequences (2.54). The explanation is given by the extreme difficulty in determining the function $n^\circ(d, \varepsilon)$. The difficulty is twofold. On one hand, it consists in deriving the values of ε for which the inequalities $\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d} \leq \varepsilon$ (see (2.51)) are verified. Indeed, the optimal solution $\gamma^{d\circ}$ is unknown. On the other hand, it consists in calculating (or, at least, in estimating from above, in a sufficiently tight way) the constant c in (2.47).

In practice, all that we can do is to choose, for given values of $d = 1, 2, \dots$, a sequence of integers n such that they are “sufficiently larger” than the corresponding integers $n^\circ(1, \varepsilon), n^\circ(2, \varepsilon), \dots, n^\circ(d, \varepsilon), \dots$. Then, we consider a sequence of related sets $\{\mathcal{A}_n^d\}_{n=1}^{\infty}$, $d = 1, 2, \dots$, and we solve the NLP problems $\{P_n^d\}_{n=1}^{\infty}$, $d = 1, 2, \dots$. Thus, we obtain sequences of FSP functions (2.54) that are “reasonably” expected to verify Inequalities (2.51).

Case 2. Exponential growth of n with d

An opposite situation occurs if, for some $c' > 0$, one has

$$\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d} \geq c' \frac{1}{n^{1/d}}, \quad d = 1, 2, \dots, n = 1, 2, \dots. \quad (2.55)$$

This is related to (and stronger than) having a lower bound of order $\Omega(1/n^{1/d})$ on the errors $\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d}$, because (2.55) is expressed in terms of an absolute constant.

We also need the guarantee that the error is at most ε , i.e.,

$$\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d} \leq \varepsilon. \quad (2.56)$$

We report jointly the Inequalities (2.55) and (2.56). For brevity, we do not impose explicitly assumptions similar to Assumptions 2.10 and 2.11 as required in Case 1. We have (compare with (2.51))

$$c' \frac{1}{n^{1/d}} \leq \|\gamma_n^{d^\circ} - \gamma^{d^\circ}\|_{\mathcal{H}^d} \leq \varepsilon, \quad d = 1, 2, \dots, n = 1, 2, \dots. \quad (2.57)$$

Therefore, from (2.57), it has to be

$$c' \frac{1}{n^{1/d}} \leq \varepsilon,$$

hence

$$n \geq \left(\frac{c'}{\varepsilon} \right)^d.$$

Then, we define

$$\tilde{n}^\circ(d, \varepsilon) \triangleq \left\lceil \left(\frac{c'}{\varepsilon} \right)^d \right\rceil, \quad (2.58)$$

where $\tilde{n}^\circ(d, \varepsilon)$ is the minimum model complexity that satisfies the first inequality in (2.57). Note that, unlike Sufficient Condition (2.53), the inequality

$$n \geq \tilde{n}^\circ(d, \varepsilon) \quad (2.59)$$

is *necessary* to obtain an error not larger than ε .

To sum up, FSP functions, whose rates of approximate optimization are characterized by (2.47) and small values of the exponent p/q (see (2.50)), imply a feasible design, i.e., the use of a moderate number of parameters. Instead, sequences with rates characterized by (2.55) are unfeasible in high-dimensional settings. In order to explain graphically – even though qualitatively – the concepts related to Inequalities (2.53) and (2.59), in Fig. 2.8 we show the (discrete) “surfaces” \mathcal{S}_1 and \mathcal{S}_2 that represent the functions $n^\circ(d, \varepsilon)$ (see (2.52)) and $\tilde{n}^\circ(d, \varepsilon)$ (see (2.58)), respectively. We depict a portion of such surfaces for a certain interval of values for ε , and with d taking its values from 1 to a certain integer. For a given value of ε , the function $\tilde{n}^\circ(d, \varepsilon)$ has an exponential growth with d , whereas $n^\circ(d, \varepsilon)$ has a polynomial growth. In the figure, we assume the latter growth to be linear. Of course, for a fixed value of d , both functions decrease when ε increases.

Let us assume that the properties of the sequence (2.42) of Problems P^d and of the chosen family of the FSP functions are such that Inequality (2.47) is verified. Then, for any pair (d, ε) , the choice of a “moderate” model complexity n “taken above” the surface \mathcal{S}_1 is a guarantee that the desired accuracy in approximating the optimal decision function γ^{d° is achieved. On the contrary, if Inequality (2.55) has to be verified to obtain the desired accuracy, the value of n must be chosen above the surface \mathcal{S}_2 . In this case, the curse of dimensionality is incurred.

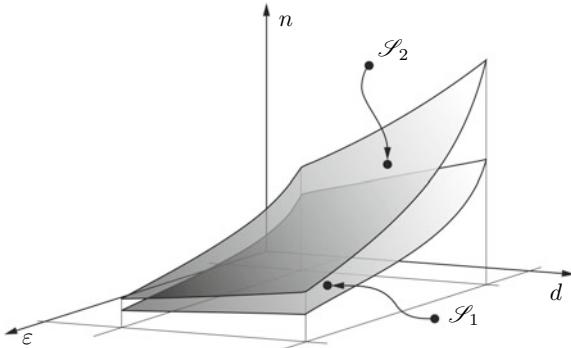


Fig. 2.8 Qualitative interpretation of Functions (2.52) and (2.58), graphically described by the surfaces \mathcal{S}_1 and \mathcal{S}_2 , respectively. In the case described by Function (2.52), in order to achieve an error not larger than ε in approximating optimal solutions γ^{d_0} of Problems P^d , it is *sufficient* to choose optimizing FSP functions with model complexities “above” the surface \mathcal{S}_1 without incurring the curse of dimensionality when the dimension d increases. However, in the case described by (2.58), to avoid an error not larger than ε , it is *necessary* to choose model complexities “above” the surface \mathcal{S}_2 . Then, the curse of dimensionality is incurred

2.6 Approximating Sets

In this and the next sections, we shall address classes of FSP functions aimed at approximating (in some suitable sense) given functions over subsets of \mathbb{R}^d . In other words, we are interested in facing the *function approximation problem*, stated formally later in the book.

Typically, the normed spaces of functions to be approximated refer to the following two cases:

Case (i) The spaces so far considered in the definition of the sequence (2.42) of Problems P^d , that is, the spaces \mathcal{H}^d of the admissible decision functions (or the subsets $\mathcal{S}^d \subset \mathcal{H}^d$ if constraints on such functions are present).

Case (ii) Other normed spaces different from \mathcal{H}^d , e.g., the ones involved in the application of dynamic programming, considered in Chaps. 6, 7, and 8.

In Case (i), we are interested in the properties of the FSP functions when they are called to approximate the admissible solutions of Problem P^d , i.e., the functions

$$\gamma^d \in \mathcal{S}^d \subseteq \mathcal{H}^d. \quad (2.60)$$

Toward this end, we can exploit the results obtained in the area of function approximation theory, a well-established field of mathematics. By contrast, as already said, methods like the ERIM, aimed at solving IDO problems in an approximate way, still rely on a narrower theoretical basis. Moreover, there are other reasons for introducing the function approximation problem in the context of approximate solutions to Problems P^d . Let us assume that all the decision functions γ^d , belonging to the

admissible set \mathcal{S}^d , are characterized by certain regularity properties and that, thanks to such properties, some family of FSP functions exists so that its elements can approximate arbitrarily well any function $\gamma^d \in \mathcal{S}^d$, hence, every optimal decision function $\gamma^{d\circ}$. If this assumption holds true, it represents a very encouraging premise for a successful application of the ERIM. Indeed, consider the sequence $\{\gamma_n^{d\circ}\}_{n=1}^\infty$ of optimal solutions to Problems P_n^d and suppose that this sequence converges to a function γ^{d*} . Then, as we shall see in Sect. 2.8, if suitable assumptions are verified, it follows that $\gamma^{d\circ} = \gamma^{d*}$.

Finally, for a given upper bound on the approximation error, let us suppose that some family of FSP functions can approximate every function $\gamma^d \in \mathcal{S}^d$ (hence also $\gamma^{d\circ}$) with a model complexity that increases “moderately” (in the sense stated in Sect. 2.5, i.e., polynomially) with respect to d . This assumption leads us to assert that the ERIM can potentially be applied in the context of IDO with good chances of mitigating the curse of dimensionality. This issue will be discussed in the next section.

In Case (ii), the reason for addressing the function approximation problem stems from the fact that we do not want to limit ourselves to facing IDO problems only by the ERIM. For instance, one of the most important IDO problems in which we are interested is the T -stage optimal control problem (see the example in Sect. 1.5.3). As is well known, DP is a powerful instrument for solving analytically (when this is possible) or numerically T -stage optimal control problems. DP will be described extensively in Chaps. 6, 7, and 8. More specifically, we shall investigate whether DP can benefit by the use of suitable FSP functions in approximating the optimal cost-to-go and control functions. Another important application of the FSP functions for function approximation problems will be presented in Chap. 10, where we shall address the optimal control problem stated over an infinite time horizon.

In general, in Case (ii), we have to approximate functions belonging to subsets \mathcal{M}^d of normed spaces \mathcal{G}^d different from the ambient spaces \mathcal{H}^d of instances of Problems P^d . (For the reader’s convenience, we point out explicitly that we refer to \mathcal{H}^d when we address Problem P^d , and to \mathcal{G}^d when we address the function approximation problem.) Then, (2.60) is replaced with

$$\gamma^d \in \mathcal{M}^d \subseteq \mathcal{G}^d. \quad (2.61)$$

Once a suitable upper bound for the approximation error and a family of approximating FSP functions have been defined, approximating a function leads to a problem similar to Problem P_n^d . Indeed, the unknowns are given by the “free” parameters of the approximating FSP functions, i.e., by the vector \mathbf{w}_n . Let us assume the *approximation error* to be given by the *distance* (measured according to the norm $\|\cdot\|_{\mathcal{G}^d}$ of \mathcal{G}^d) between a function $\gamma^d \in \mathcal{M}^d$ and an FSP function $\gamma_n^d \in \mathcal{A}_n^d$. Then, for $n = 1, 2, \dots$, the approximation error or the distance between γ^d and γ_n^d can be regarded as a cost function (see (2.11)), that is

$$\tilde{F}^d(\mathbf{w}_n) \triangleq \|\gamma_n^d(\cdot, \mathbf{w}_n) - \gamma^d(\cdot)\|_{\mathcal{G}^d}.$$

Now, we consider the *distance between a function* $\gamma^d \in \mathcal{M}^d$ *and the approximating set* \mathcal{A}_n^d . Roughly speaking, this distance is the one between γ^d and the FSP function $\gamma_n^d \in \mathcal{A}_n^d$ that is “closest” to γ^d (when such an FSP function γ_n^d exists, otherwise the infimum of such distances over \mathcal{A}_n^d is taken). The solution of the following problem enables one to determine such a distance.

Problem PA_n^d (*Function approximation problem*). *Given a function* $\gamma^d \in \mathcal{M}^d$, *find the distance between the function* γ^d *and the approximation set* \mathcal{A}_n^d , *given by*

$$\tilde{d}(\gamma^d, \mathcal{A}_n^d) \triangleq \inf_{\mathbf{w}_n} \tilde{F}^d(\mathbf{w}_n) = \inf_{\mathbf{w}_n} \|\gamma_n^d(\cdot, \mathbf{w}_n) - \gamma^d(\cdot)\|_{\mathcal{G}^d}. \quad (2.62)$$

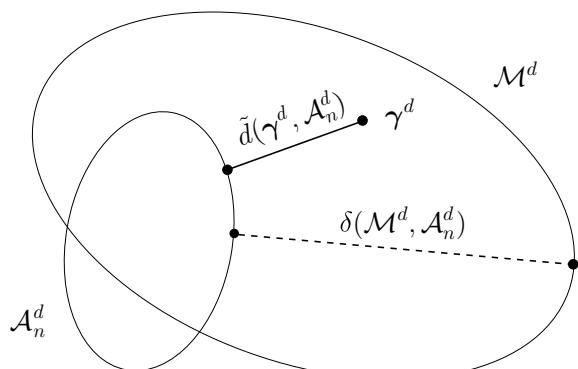
△

The distance $\tilde{d}(\gamma^d, \mathcal{A}_n^d)$ is shown in Fig. 2.9.

Remark 2.3 We have stated Problem PA_n^d in the form (1.10), i.e., searching for infima. Actually, in the case in which the infimum of the distance $\tilde{F}^d(\mathbf{w}_n)$ is attained for some \mathbf{w}_n^* , the FSP function $\gamma^d(\cdot, \mathbf{w}_n^*)$ is called a *best approximation to* γ^d *from the set* \mathcal{A}_n^d . More in general, the set \mathcal{A}_n^d of the FSP functions is said to have the *best approximation property* for the set \mathcal{M}^d if there is a best approximation $\gamma^d(\cdot, \mathbf{w}_n^*) \in \mathcal{A}_n^d$ for each $\gamma^d \in \mathcal{M}^d$. In [13] it is shown that FSP functions corresponding to multilayer sigmoidal networks do not have the best approximation property for approximating the space $\mathcal{G}^d = \mathcal{C}(D, \mathbb{R}^m)$ of continuous functions $\gamma: D \rightarrow \mathbb{R}^m$ ($D \subset \mathbb{R}^d$). Radial basis functions, on the other hand, have such a property [13]. □

Let us now address Problem PA_n^d by considering Case (i). Clearly, it is very important to know the behavior of $\tilde{d}(\gamma^d, \mathcal{A}_n^d)$ when the model complexity n increases. This behavior tells one whether, in Case (i), a function $\gamma^d \in \mathcal{S}^d$ can be approximated to any degree of accuracy by using FSP functions belonging to \mathcal{A}_n^d . If it is so, it means that *there exist FSP functions able to approximate arbitrarily well the optimal solution* $\gamma^{d\circ}$. This is an important requirement that should be verified before searching for P^d-optimizing sequences. Indeed, although we are interested in approximating

Fig. 2.9 The continuous segment represents the distance between the function γ^d and the set \mathcal{A}_n^d (see Definition (2.62)). The dashed segment represents the deviation $\delta(\mathcal{M}^d, \mathcal{A}_n^d)$ of \mathcal{M}^d from \mathcal{A}_n^d , i.e., the worst-case error of approximation of the functions $\gamma^d \in \mathcal{M}^d$ by the FSP functions $\gamma_n^d \in \mathcal{A}_n^d$ (see Definition 2.7)



merely $\gamma^{d\circ}$, such an optimal solution is unknown. Naturally, the rate at which the approximation error decreases is also of utmost importance.

Of course, Case (ii) too leads us to study the approximation properties of the families of the FSP functions in generic sets $\mathcal{M}^d \subseteq \mathcal{G}^d$. Thus, we introduce the following definitions.

Definition 2.5 (Density) A nested sequence $\{\mathcal{A}_n^d\}_{n=1}^\infty$ of sets of FSP functions is dense in $\mathcal{M}^d \subseteq \mathcal{G}^d$ if, for any $\varepsilon > 0$ and any $\gamma^d \in \mathcal{M}^d$, there exist a model complexity \bar{n} and a parameter vector $\bar{\mathbf{w}}_{\bar{n}}$ (hence an FSP function $\tilde{\gamma}_{\bar{n}}^d \triangleq \gamma^d(\cdot, \bar{\mathbf{w}}_{\bar{n}}) \in \mathcal{A}_{\bar{n}}^d$) such that one has $\|\tilde{\gamma}_{\bar{n}}^d - \gamma^d\|_{\mathcal{G}^d} \leq \varepsilon$. \triangleleft

Definition 2.6 A nested sequence $\{\mathcal{A}_n^d\}_{n=1}^\infty$ of sets of FSP functions that is dense in a set $\mathcal{M}^d \subseteq \mathcal{G}^d$ is called “ \mathcal{M}^d -approximating sequence of sets” (we also say that it is provided with the “ \mathcal{M}^d -density property”). \triangleleft

Remark 2.4 From now on, we shall find it convenient to use the term “ \mathcal{M}^d -approximating FSP functions” for the FSP functions belonging to an \mathcal{M}^d -approximating sequence of sets. However, this contains an abuse of terminology as the density property is associated with a sequence $\{\mathcal{A}_n^d\}_{n=1}^\infty$ of sets of FSP functions and not to a sequence of FSP functions. With this specification clearly in mind, there should be no risk of ambiguity. Moreover, when the set \mathcal{M}^d will be clear from the context, sometimes we shall write simply “density property,” “approximating sequence of sets,” and “approximating FSP functions.” \triangleleft

Let us now specify the norms used to measure the distances between functions for typical scenarios that will be addressed in the book, namely, problems like the ones considered in Sect. 1.5. More specifically, two normed spaces \mathcal{G} are particularly important:

- (1) The space $\mathcal{C}(D, \mathbb{R}^m)$ of continuous functions $\gamma : D \rightarrow \mathbb{R}^m$ ($D \subseteq \mathbb{R}^d$) equipped with the supremum norm

$$\|\gamma\|_{\mathcal{C}(D, \mathbb{R}^m)} \triangleq \sup_{\mathbf{x} \in D} |\gamma(\mathbf{x})|. \quad (2.63)$$

- (2) The space $\mathcal{L}_2(D, \mathbb{R}^m)$ of square-integrable functions $\gamma : D \rightarrow \mathbb{R}^m$ equipped with the \mathcal{L}_2 norm

$$\|\gamma\|_{\mathcal{L}_2(D, \mathbb{R}^m)} \triangleq \left[\int_D |\gamma(\mathbf{x})|^2 d\mathbf{x} \right]^{1/2}, \quad (2.64)$$

or, more generally for $1 \leq p < \infty$, the normed space $\mathcal{L}_p(D, \mathbb{R}^m)$ of functions $\gamma : D \rightarrow \mathbb{R}^m$ equipped with the \mathcal{L}_p norm

$$\|\gamma\|_{\mathcal{L}_p(D, \mathbb{R}^m)} \triangleq \left[\int_D |\gamma(\mathbf{x})|^p d\mathbf{x} \right]^{1/p}. \quad (2.65)$$

Let us first refer to the normed space $\mathcal{C}(D, \mathbb{R}^m)$. Accordingly, we have \mathcal{C} -approximating sequences (we put $\mathcal{M} = \mathcal{C}$). On the basis of Definition 2.5, we can say that, given any function $\gamma \in \mathcal{C}(D, \mathbb{R}^m)$ and any $\varepsilon > 0$, there exist an $\bar{n} \in \mathbb{Z}^+$ and a vector $\bar{\mathbf{w}}_{\bar{n}} \in \mathbb{R}^{\mathcal{N}(\bar{n})}$ (then an FSP function $\tilde{\gamma}_{\bar{n}}$) such that one has

$$\|\tilde{\gamma}_{\bar{n}} - \gamma\|_{\mathcal{C}(D, \mathbb{R}^m)} \leq \varepsilon. \quad (2.66)$$

Hence,

$$|\gamma(\mathbf{x}, \bar{\mathbf{w}}_{\bar{n}}) - \gamma(\mathbf{x})| \leq \varepsilon, \quad \forall \mathbf{x} \in D. \quad (2.67)$$

The space $\mathcal{C}(D, \mathbb{R}^m)$ with the norm $\|\cdot\|_{\mathcal{C}(D, \mathbb{R}^m)}$ is of particular interest whenever the error $|\gamma(\mathbf{x}, \bar{\mathbf{w}}_{\bar{n}}) - \gamma(\mathbf{x})|$ must not exceed a given threshold ε for every $\mathbf{x} \in D$. For example (consistently with intuition), Inequality (2.67) is an important “guarantee” that ensures asymptotic properties like stability for approximate controllers or convergence of estimation errors for approximate state estimators. This property is related to the density property of polynomials stated by the classical Weierstrass’ theorem. If such a guarantee is not required, as is often the case with discrete-time optimal control and state estimation problems stated over a finite number of temporal stages, it may be an unnecessarily strong requirement that the functions to be approximated belong to the space $\mathcal{C}(D, \mathbb{R}^m)$.

To allow for suitably weaker conditions, one can consider the normed space $\mathcal{L}_p(D, \mathbb{R}^m)$, $1 \leq p < \infty$. An \mathcal{L}_p -approximating sequence is given by $\{\mathcal{A}_n\}_{n=1}^\infty$ provided with the density property in $\mathcal{L}_p(D, \mathbb{R}^m)$. Given any function $\gamma : D \rightarrow \mathbb{R}^m$, $\gamma \in \mathcal{L}_p(D, \mathbb{R}^m)$, and any $\varepsilon > 0$, such a density property requires that there exist an $\bar{n} \in \mathbb{Z}^+$ and a vector $\bar{\mathbf{w}}_{\bar{n}} \in \mathbb{R}^{\mathcal{N}(\bar{n})}$ such that one has

$$\|\tilde{\gamma}_{\bar{n}} - \gamma\|_{\mathcal{L}_p(D, \mathbb{R}^m)} \leq \varepsilon.$$

To simplify the notation, in the rest of this chapter and in the subsequent ones we often write $\mathcal{C}(D)$ and $\mathcal{L}_p(D)$ instead of $\mathcal{C}(D, \mathbb{R}^m)$ and $\mathcal{L}_p(D, \mathbb{R}^m)$, respectively. For brevity, sometimes we shall simply write \mathcal{C} - or \mathcal{L}_p -density property on compact sets.

2.7 Polynomially Complex Approximating Sequences of Sets

In Sect. 2.5, we have pointed out how essential it is that a P^d -optimizing sequence also enjoys the property of being a polynomially complex P^d -optimizing sequence. Subsequently, in Sect. 2.6, we have stressed the importance of using \mathcal{M}^d -approximating FSP functions for holding good hopes for solving the basic Problem P^d by the ERIM. Clearly, we are still very interested in understanding whether families of \mathcal{M}^d -approximating sets of FSP functions exist that enable one to achieve a desired

accuracy in approximating the optimal solution of Problem P^d by using “moderate” model complexities n when the dimension d increases. To be more specific, we are interested in establishing relationships that show the rate at which the approximation error goes to zero when n tends to infinity. Such relationships must take into account the dimension d .

In this respect, we recall once again that the results so far established in the literature about function approximation in high-dimensional settings are much more extensive than the ones obtained for the optimizing sequences, hence, for the ERIM. The possibility of transferring (at least in part) the results from the area of function approximation to the area of IDO drives us to gain a deep insight into the former area.

In this section, we shall define families of FSP functions that have the fundamental property of approximating the functions $\gamma^d \in \mathcal{M}^d$ (by solving Problems PA_n^d) with a given accuracy provided with a model complexity n growing *only polynomially* with d . As is consistent with intuition, the functions γ^d have to be equipped with “suitable” regularity properties. Let us suppose that \mathcal{M}^d is made of all the functions γ^d having such regularity properties.

2.7.1 The Worst-Case Error of Approximation of Functions

We are now interested in measuring the performances of an \mathcal{M}^d -approximating sequence $\{\mathcal{A}_n^d\}_{n=1}^\infty$ by means of the worst-case error of approximation of elements of \mathcal{M}^d by elements of $\{\mathcal{A}_n^d\}_{n=1}^\infty$. Such a worst-case error is formalized in the following definition.

Definition 2.7 Let \mathcal{M}^d be a subset of a normed space \mathcal{G}^d of d -variable functions. The “deviation of \mathcal{M}^d from \mathcal{A}_n^d ” is defined as³

$$\delta(\mathcal{M}^d, \mathcal{A}_n^d) \triangleq \sup_{\gamma^d \in \mathcal{M}^d} \tilde{d}(\gamma^d, \mathcal{A}_n^d) = \sup_{\gamma^d \in \mathcal{M}^d} \inf_{\gamma_n^d \in \mathcal{A}_n^d} \|\gamma_n^d - \gamma^d\|_{\mathcal{G}^d}, \quad (2.68)$$

where

$$\tilde{d}(\gamma^d, \mathcal{A}_n^d) = \inf_{\gamma_n^d \in \mathcal{A}_n^d} \|\gamma_n^d - \gamma^d\|_{\mathcal{G}^d} \quad (2.69)$$

is the distance between the function γ^d and the set \mathcal{A}_n^d , obtained by solving Problem PA_n^d (see (2.62)). \triangleleft

From now on, we shall assume only for simplicity that maximizing and minimizing functions exist. Then, “sup” and “inf” operators can be replaced by “max” and “min”, respectively. All this is consistent with Assumption 1.1.

³Note that one should specify the space \mathcal{G}^d in the definition, since the same set \mathcal{M}^d might be considered as subset of various normed linear spaces. We refer the reader to Chap. 3 for a more detailed treatment.

Once \mathcal{A}_n^d has been fixed, the deviation (2.68) measures the distance between the function belonging to \mathcal{M}^d that is “hardest to approximate” and the function in \mathcal{A}_n^d that is best suited for such an approximation. The deviation (2.68) is shown in Fig. 2.9.

2.7.2 Polynomial and Exponential Growth of the Model Complexity n with the Dimension d

Let \mathcal{M}^d be a set of functions γ^d to be approximated within a given maximum approximation error $\varepsilon > 0$ and let us consider an \mathcal{M}^d -approximating sequence of sets $\{\mathcal{A}_n^d\}_{n=1}^\infty$. Of course, we would like to avoid that the \mathcal{M}^d -approximating sequences of sets are characterized by too fast a growth of the model complexity n with the dimension d .

Depending on the behavior of the deviation $\delta(\mathcal{M}^d, \mathcal{A}_n^d)$ with respect to d and n , we make an explicit distinction between a polynomial upper bound on $\delta(\mathcal{M}^d, \mathcal{A}_n^d)$ and an exponential lower bound on the same quantity, as done in Sect. 2.5.2 for the behavior of $\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d}$. This distinction is described in the following.

Case 1. Polynomial growth of n with d

We follow the same distinctions as the ones made in Sect. 2.5.2.

Assumption 2.12 There exist constants $p, q, c \in \mathbb{R}$, $p \geq 0, q, c > 0$ such that

$$\delta(\mathcal{M}^d, \mathcal{A}_n^d) \leq c \frac{d^p}{n^q}, \quad d = 1, 2, \dots, n = 1, 2, \dots. \quad (2.70)$$

△

This is related to (and stronger than) requiring

$$\delta(\mathcal{M}^d, \mathcal{A}_n^d) = O(d^p/n^q).$$

Assumption 2.13 For every $\varepsilon > 0$ there exist $p, q, c \in \mathbb{R}$, $p \geq 0, q, c > 0$ such that the model complexities n of the optimal FSP functions $\gamma_n^{d\circ}$ verify the inequalities

$$c \frac{d^p}{n^q} \leq \varepsilon, \quad d = 1, 2, \dots, n = 1, 2, \dots. \quad (2.71)$$

△

Inequalities (2.70) and (2.71) provide (compare with (2.51))

$$\delta(\mathcal{M}^d, \mathcal{A}_n^d) \leq c \frac{d^p}{n^q} \leq \varepsilon. \quad (2.72)$$

Hence

$$n \geq \left(\frac{c}{\varepsilon} \right)^{1/q} d^{p/q}. \quad (2.73)$$

Thus, the minimum model complexity that verifies (2.73) is given by

$$n^*(d, \varepsilon) \triangleq \left\lceil \left(\frac{c}{\varepsilon} \right)^{1/q} d^{p/q} \right\rceil. \quad (2.74)$$

In such a case, the condition

$$n \geq n^*(d, \varepsilon) \quad (2.75)$$

is *sufficient* to guarantee that the deviation $\delta(\mathcal{M}^d, \mathcal{A}_n^d)$ does not exceed ε . Moreover, $n^*(d, \varepsilon)$ grows at most as a power of d . Here we can make similar remarks to those made in Sect. 2.5.2 about the dependence of the error $\|\gamma_n^{d_o} - \gamma^{d_o}\|_{\mathcal{H}^d}$ on d and n . In particular, (2.74) provides *favorable rates of growth* of n if p/q takes sufficiently low values.

On this basis, we give the following definition.

Definition 2.8 Let us consider (i) a sequence of sets \mathcal{M}^d , $d = 1, 2, \dots$, of functions γ^d to be approximated by FSP functions $\gamma_n^d \in \mathcal{A}_n^d$ and (ii) an \mathcal{M}^d -approximating sequence of sets

$$\{\mathcal{A}_n^d\}_{n=1}^{\infty}, \quad d = 1, 2, \dots \quad (2.76)$$

Let a given $\varepsilon > 0$ be the maximum allowed approximation error for any $\gamma^d \in \mathcal{M}^d$, Assumptions 2.12 and 2.13 be verified, and the model complexities of the FSP functions γ_n^d be such that $n \geq n^*(d, \varepsilon)$. For any given d , the Sequence (2.76) is defined as “ ε -polynomially complex \mathcal{M}^d -approximating sequence of sets” (ε -pc \mathcal{M}^d -approximating sequence of sets). The FSP functions belonging to the sets \mathcal{A}_n^d are called “ ε -pc \mathcal{M}^d -approximating FSP functions.”

□

In the book, whenever there will be no risk of ambiguity, we drop the symbol ε and simply use the terminology pc \mathcal{M}^d -approximating sequences of sets. Definition 2.8 states that the rate of increase of the minimum value of n with d – which guarantees a given approximation accuracy – is at most a power of d . From a qualitative point of view, such a polynomially bounded complexity can be interpreted as an indication of “moderate” computational burden in approximating functions dependent on a large number of variables. Roughly speaking, we say that one can associate a set \mathcal{M}^d with a class of approximating FSP functions that makes the problem of approximating the elements of \mathcal{M}^d “computationally feasible.”

Case 2. Exponential growth of n with d

By imposing constraints similar to (2.55) and (2.56), with $\delta(\mathcal{M}^d, \mathcal{A}_n^d)$ in place of $\|\gamma_n^{d_o} - \gamma^{d_o}\|_{\mathcal{H}^d}$, we assume for a suitable positive constant c'

$$\delta(\mathcal{M}^d, \mathcal{A}_n^d) \geq c' \frac{1}{n^{1/d}}, \quad d = 1, 2, \dots, n = 1, 2, \dots. \quad (2.77)$$

and

$$\delta(\mathcal{M}^d, \mathcal{A}_n^d) \leq \varepsilon. \quad (2.78)$$

From (2.77) and (2.78) we have (compare with (2.57))

$$c' \frac{1}{n^{1/d}} \leq \delta(\mathcal{M}^d, \mathcal{A}_n^d) \leq \varepsilon, \quad d = 1, 2, \dots, n = 1, 2, \dots, \quad (2.79)$$

which gives

$$c' \frac{1}{n^{1/d}} \leq \varepsilon$$

and then

$$n \geq \left(\frac{c'}{\varepsilon} \right)^d.$$

In this case, \mathcal{M} -approximating FSP functions with model complexities not smaller than

$$\tilde{n}^*(d, \varepsilon) \triangleq \left\lceil \left(\frac{c'}{\varepsilon} \right)^d \right\rceil$$

are required to obtain an approximation error not larger than ε . Likewise in Sect. 2.5.2, the inequality $n \geq \tilde{n}^*$ is a *necessary* condition to satisfy this request. Such an exponential dependence of \tilde{n}^* on the dimension d gives rise to the curse of dimensionality. Every \mathcal{M}^d -approximating sequence $\{\mathcal{A}_n^d\}_{n=1}^\infty$ affected by it is unsuitable in high-dimensional settings.

2.8 Connections Between Approximating Sequences and Optimizing Sequences

In this section, we consider the role played by \mathcal{M}^d - and pc \mathcal{M}^d -approximating sequences $\{\mathcal{A}_n^d\}_{n=1}^\infty$ within the ERIM, in search of connections with the \mathbb{P}^d -optimizing sequences $\{\gamma_n^{d\circ}\}_{n=1}^\infty$. For the reader's convenience, Table 2.1 sums up the definitions and denominations of the *two types of sequences of FSP functions* defined in Sects. 2.4 and 2.5, whereas Table 2.2 regards the *two types of sequences of sets* of FSP functions defined in Sects. 2.6 and 2.7.

Let us consider again a sequence (2.42) of IDO problems

$$\{\mathbb{P}^d : d = 1, 2, \dots\}$$

Table 2.1 From P^d -optimizing sequences to ε -polynomially complex P^d -optimizing sequences of FSP functions

Starting sequence of FSP functions	Assumptions	Reference of the sequence obtained
Sequence of FSP functions $\{\gamma_n^{d^\circ}\}_{n=1}^\infty$	Assumptions 2.6, 2.7: $\lim_{n \rightarrow \infty} \ \gamma_n^{d^\circ} - \gamma^{d^\circ}\ _{\mathcal{H}^d} = 0 \quad (2.38)$ $\lim_{n \rightarrow \infty} F_n^\circ = F^\circ \quad (2.39)$	Definition 2.3
Sequence of P^d -optimizing FSP functions $\{\gamma_n^{d^\circ}\}_{n=1}^\infty$	Assumptions 2.6, 2.7, 2.10, 2.11: $\exists c, p, q \in \mathbb{R} (p \geq 0, c, q > 0) \text{ s.t.}$ $(i) \ \gamma_n^{d^\circ} - \gamma^{d^\circ}\ _{\mathcal{H}^d} \leq c \frac{d^p}{n^q} \quad (2.47)$ $(ii) c \frac{d^p}{n^q} \leq \varepsilon \quad (2.49)$ \downarrow $n \geq n^\circ(d, \varepsilon)$	Definition 2.4

Table 2.2 From \mathcal{M}^d -approximating sequences to ε -polynomially complex \mathcal{M}^d -approximating sequences of sets of FSP functions

Starting sequence of FSP functions	Assumptions	Reference of the sequence obtained
Nested sequence of sets $\{\mathcal{A}_n^d\}_{n=1}^\infty$	$\{\mathcal{A}_n^d\}_{n=1}^\infty$ is dense in \mathcal{M}^d	Definition 2.6
\mathcal{M}^d -approximating sequence of sets $\{\mathcal{A}_n^d\}_{n=1}^\infty$	Assumptions 2.12, 2.13: $\exists c, p, q \in \mathbb{R} (p \geq 0, c, q > 0) \text{ s.t.}$ $(i) \delta(\mathcal{M}^d, \mathcal{A}_n^d) \leq c \frac{d^p}{n^q} \quad (2.70)$ $(ii) c \frac{d^p}{n^q} \leq \varepsilon \quad (2.71)$ \downarrow $n \geq n^\star(d, \varepsilon)$	Definition 2.8

and recall that, for each Problem P^d , $d = 1, 2, \dots$, searching for optimizing sequences $\{\gamma_n^{d^\circ}\}_{n=1}^\infty$ means solving the sequence of NLP problems

$$\{P_n^d : n = 1, 2, \dots\}.$$

In Fig. 2.10, we expand and specify the conceptual steps shown in Fig. 1.2 to solve IDO problems approximately through the ERIM. In addition to Fig. 2.10, Tables 2.1

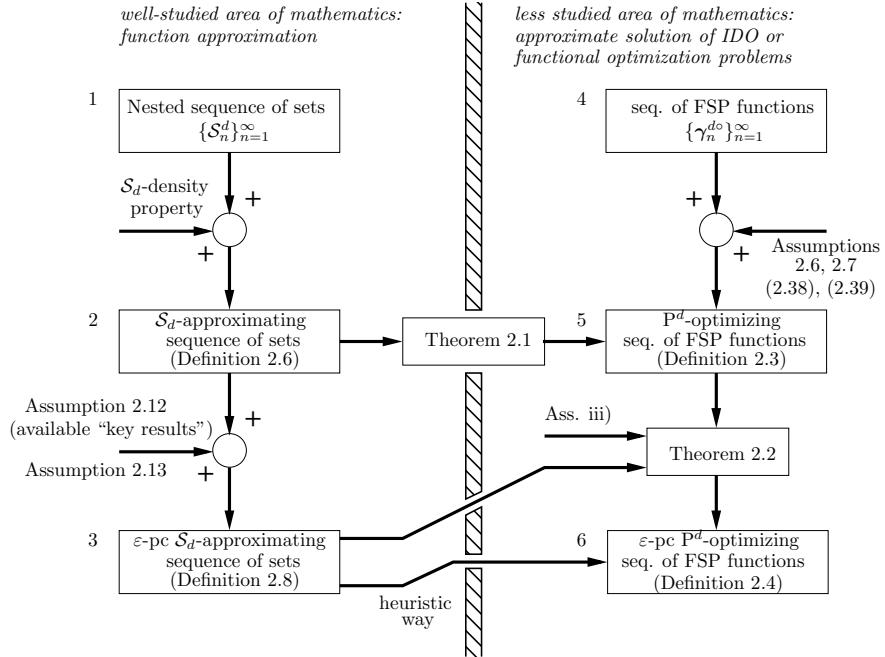


Fig. 2.10 From the area of function approximation to the area of IDO or functional optimization

and 2.2 should provide a complete picture concerning the two types of sequences of functions $\{\gamma_n^{do}\}_{n=1}^\infty$ and the two types of sequences of sets $\{\mathcal{A}_n^d\}_{n=1}^\infty$ that have been considered in the previous sections. In particular, the “transfer tools” mentioned in Fig. 1.2 are expanded in Theorems 2.1 and 2.2 later in this section. Moreover, Fig. 2.10 shows the connections among the various sequences, enhancing “paths” that, from the “initial” blocks 1 and 3, get to the “final” block 6. Such a block is the target that we would like to reach. In fact, block 6 contains the FSP functions which should allow us to solve approximately the original IDO problem via the solution of an “adequate” number of FDO problems, i.e., Problems P_n^d . This way one can obtain approximate solutions to any desired degree of accuracy, hopefully not incurring the curse of dimensionality.

2.8.1 From \mathcal{S}^d -Approximating Sequences of Sets to P^d -Optimizing Sequences of FSP Functions

For every d , let now $\mathcal{M}^d = \mathcal{S}^d$ and consider pc \mathcal{S}^d -approximating sequences $\{\mathcal{A}_n^d\}_{n=1}^\infty$. Let us address at first the basic question whether the sequences $\{\gamma_n^{do}\}_{n=1}^\infty$ may become P^d -optimizing sequences. Theorem 2.1 adapts the results stated in [12, p. 196] to our framework.

Theorem 2.1 For every d , let us consider a given instance of Problem P^d belonging to the sequence (2.42) of IDO problems and let Assumptions 1.1 and 2.7 hold. Let the following further assumptions also be verified:

- (i) The cost functional $F^d(\gamma^d)$ is continuous in the norm of \mathcal{H}^d .
- (ii) The optimal function $\gamma^{d\circ}$ is unique.
- (iii) $\{\mathcal{S}_n^d\}_{n=1}^\infty$ is an \mathcal{S}^d -approximating sequence.
- (iv) The sequence $\{\gamma_n^{d\circ}\}_{n=1}^\infty$ is such that

$$\lim_{n \rightarrow \infty} \gamma_n^{d\circ} = \gamma^{d*} \in \mathcal{S}^d. \quad (2.80)$$

Then, $\{\gamma_n^{d\circ}\}_{n=1}^\infty$ is a P^d -optimizing sequence, that is,

$$\lim_{n \rightarrow \infty} F(\gamma_n^{d\circ}) = F(\gamma^{d\circ}) \quad (2.81)$$

and

$$\gamma^{d\circ} = \gamma^{d*}. \quad (2.82)$$

□

Proof As $F^{d\circ} = F^d(\gamma^{d\circ})$ is the minimum of F^d over \mathcal{S}^d and F^d is continuous at $\gamma^{d\circ}$, for every $\varepsilon > 0$, there exists $\delta > 0$ such that

$$F^{d\circ} \leq F^d(\gamma^d) < F^{d\circ} + \varepsilon \quad (2.83)$$

for all functions $\gamma^d \in \mathcal{S}^d$ for which

$$\|\gamma^{d\circ} - \gamma^d\|_{\mathcal{H}^d} < \delta. \quad (2.84)$$

The existence of an \mathcal{S}^d -approximating sequence implies that there exists $\hat{n} \in \mathbb{Z}^+$ and a vector $\hat{\mathbf{w}}_{\hat{n}} \in \mathbb{R}^{\mathcal{N}(\hat{n})}$ (hence, an FSP function $\hat{\gamma}_{\hat{n}}^d = \gamma^d(\cdot, \hat{\mathbf{w}}_{\hat{n}})$) such that

$$\|\hat{\gamma}_{\hat{n}}^d - \gamma^{d\circ}\|_{\mathcal{H}^d} < \delta. \quad (2.85)$$

Then, by (2.83), Assumption 2.7, and the nestedness of the sequence $\{\mathcal{S}_n^d\}_{n=1}^\infty$, we get

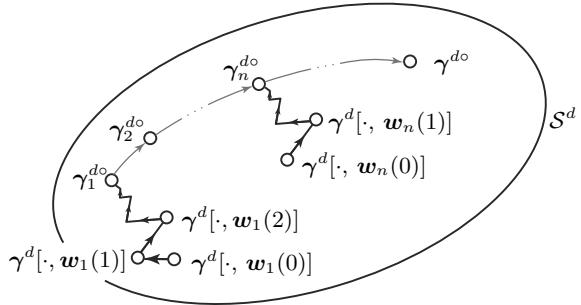
$$F^{d\circ} \leq F_n^{d\circ} = F^d(\gamma_n^{d\circ}) \leq F^d(\hat{\gamma}_{\hat{n}}^d) < F^{d\circ} + \varepsilon, \quad \forall n \geq \hat{n}.$$

Since ε is arbitrary, it follows that

$$\lim_{n \rightarrow \infty} F^d(\gamma_n^{d\circ}) = F^d(\gamma^{d*}) = F(\gamma^{d\circ}),$$

i.e., γ^{d*} is a minimizer of F^d over \mathcal{S}^d . The uniqueness of $\gamma^{d\circ}$ implies (2.82). ■

Fig. 2.11 An NLP algorithm solves Problems $P_1^d, \dots, P_n^d, \dots$ and determines the optimal FSP functions $\gamma_1^{d_o}, \dots, \gamma_n^{d_o}, \dots$. Two trajectories of the parameter vector are shown



Clearly, the availability of an NLP algorithm solving the sequence of Problems $P_1^d, \dots, P_n^d, \dots$ for any given dimension d is of major importance. Such an algorithm determines the optimal functions $\gamma_1^{d_o}, \dots, \gamma_n^{d_o}, \dots$ over the sets $S_1^d, \dots, S_n^d, \dots$. Of course, we have

$$F^d(\gamma_1^o) \geq \dots \geq F^d(\gamma_n^{d_o}) \geq \dots$$

Remark 2.5 The existence of Limits (2.80) (with (2.82)) and (2.81) allows us to say that $\{\gamma_n^{d_o}\}_{n=1}^\infty$ is a P^d -optimizing sequence. This is important not only from a theoretical point of view but also from a practical one. As, in general, one can determine only some functions belonging to the sequence $\{\gamma_n^{d_o}\}_{n=1}^\infty$, it is of primary importance to know that, at least in principle, it is possible anyway to get arbitrarily close to the optimal function γ^{d_o} and to compute the optimal value F^{d_o} of the cost functional arbitrarily well. The sequence $\{\gamma_n^{d_o}\}_{n=1}^\infty$ is shown in Fig. 2.11 together with two symbolic trajectories that determine two optimal FSP functions by some NLP descent algorithm. \triangleleft

Remark 2.6 As shown in Fig. 2.10, Theorem 2.1 establishes a “passage” (see Fig. 1.2) from the well-studied area of function approximation to the less studied area of IDO. However, once the existence of P^d -optimizing sequences of FSP functions has been ascertained, the transition from block 5 to block 6 (our final objective) is problematic, owing to the difficulty of satisfying Assumption 2.10, that is, the inequalities (see (2.47))

$$\|\gamma_n^{d_o} - \gamma^{d_o}\|_{\mathcal{H}_d} \leq c \frac{d^p}{n^q}, \quad d = 1, 2, \dots, n = 1, 2, \dots$$

As we have noted in Sect. 1.1, for historical and applicational reasons very important theoretical results have been achieved in the area of function approximation (see the Assumption 2.12 and the “key results” in Fig. 2.10). This holds true typically for FSP functions taking the structure of OHL networks – like sigmoidal neural networks, radial basis functions with variable centers and widths, trigonometric polynomials with free frequencies, spline functions with free knots, etc. – already mentioned

in Chap. 1 and described in more detail in Chap. 3. Briefly speaking, such “key results” consist in satisfying, under suitable regularity properties of the functions to be approximated, Assumption 2.12, that is (see (2.70) with $\mathcal{M}^d = \mathcal{S}^d$), the inequalities

$$\delta(\mathcal{M}^d, \mathcal{A}_n^d) \leq c \frac{d^p}{n^q}, \quad d = 1, 2, \dots, n = 1, 2, \dots.$$

△

Of course, the assumptions necessary to apply both Theorems 2.1 and 2.2 to reach block 6 of Fig. 2.10 (and then to demonstrate the existence of pc optimizing sequences) may be difficult or even impossible to verify. Then, we have left the link termed “heuristic way” in Fig. 2.10. The link is justified by the large number of successful numerical results appeared in the literature, some of which reported in the applicative chapters of the book (see Fig. 1.3).

2.8.2 *From P^d -Optimizing Sequences of FSP Functions to Polynomially Complex P^d -Optimizing Sequences of FSP Functions*

By Theorem 2.1 we obtained the possibility of constructing P^d -optimizing sequences of FSP functions, that is, of “passing from the left to the right side of the wall” in Fig. 2.10, provided that some suitable assumptions are verified. Moreover, the availability of “key results,” achieved in the area of function approximation and pointed out in Remark 2.6, constitutes a basic premise for adding the property of being polynomially complex to the optimizing sequences. Finally, a third assumption is required by Theorem 2.2 (see the arrow labeled Ass. (iii) in Fig. 2.10).

All this leads us to state the following theorem that generalizes a result given in [38].

Theorem 2.2 *For every positive integer d , let us consider a given instance of Problem P^d belonging to the sequence (2.42) of IDO problems. Let the following assumptions be verified:*

- (i) $\{\gamma_n^{d\circ}\}_{n=1}^\infty$ is a P^d -optimizing sequence.
- (ii) $\{\mathcal{S}_n^d\}_{n=1}^\infty$ is a pc \mathcal{S}^d -approximating sequence of sets.
- (iii) There exists $\tau \in \mathbb{R}$, $\tau > 0$, such that, for any function $\gamma^d \in \mathcal{S}^d$, the cost functional $F(\gamma^d)$ verifies the inequalities

$$\begin{aligned} \kappa_1^- (\|\gamma^d - \gamma^{d\circ}\|_{\mathcal{H}^d})^{\alpha_1} &\leq F(\gamma^d) - F(\gamma^{d\circ}) \\ &\leq \kappa_1^+ (\|\gamma^d - \gamma^{d\circ}\|_{\mathcal{H}^d})^{\beta_1}, \quad \text{if } \|\gamma^d - \gamma^{d\circ}\|_{\mathcal{H}^d} \leq \tau, \end{aligned} \tag{2.86}$$

and

$$\begin{aligned} \kappa_2^- (\|\gamma^d - \gamma^{d\circ}\|_{\mathcal{H}^d})^{\alpha_2} &\leq F(\gamma^d) - F(\gamma^{d\circ}) \\ &\leq \kappa_2^+ (\|\gamma^d - \gamma^{d\circ}\|_{\mathcal{H}^d})^{\beta_2}, \quad \text{if } \|\gamma^d - \gamma^{d\circ}\|_{\mathcal{H}^d} > \tau, \end{aligned} \quad (2.87)$$

where $\alpha_1, \alpha_2, \beta_1, \beta_2$ are positive scalars such that $\alpha_1 \geq \beta_1$ and $\alpha_2 \leq \beta_2$, and $\kappa_1^-, \kappa_1^+, \kappa_2^-, \kappa_2^+$ are suitable positive constants⁴ such that

$$\kappa_1^- \tau^{\alpha_1} = \kappa_2^- \tau^{\alpha_2} \leq \kappa_1^+ \tau^{\beta_1} = \kappa_2^+ \tau^{\beta_2} \quad (2.88)$$

and κ_1^+ grows at most polynomially with respect to d , and τ and κ_1^- are bounded from 0 by a constant that does not depend on d .

Then, there exists $n^\circ(d, \varepsilon)$ such that, for $n \geq n^\circ(d, \varepsilon)$, the sequence $\{\gamma_n^{d\circ}\}_{n=1}^\infty$ is a pc P^d -optimizing sequence. \square

Proof Assumption (i) gives us the starting point of the proof. What we have to do is to equip the P^d -optimizing sequence with the property of polynomial complexity. By Assumption (ii), it follows that Inequality (2.70) is verified, i.e.,

$$\delta(\mathcal{S}^d, \mathcal{S}_n^d) = \sup_{\gamma^d \in \mathcal{S}^d} \inf_{\gamma_n^d \in \mathcal{S}_n^d} \|\gamma_n^d - \gamma^d\|_{\mathcal{H}^d} \leq c \frac{d^p}{n^q}.$$

By replacing the generic function $\gamma^d \in \mathcal{S}^d$ with $\gamma^{d\circ}$, we have

$$\inf_{\gamma_n^d \in \mathcal{S}_n^d} \|\gamma_n^d - \gamma^{d\circ}\|_{\mathcal{H}^d} \leq c \frac{d^p}{n^q}.$$

Let $\bar{\gamma}_n^d$ be an FSP function minimizing the distance between $\gamma^{d\circ}$ and \mathcal{S}_n^d (simple changes in the proof can be implemented if $\bar{\gamma}_n^d$ does not exist and this distance is expressed as an infimum). Then,

$$\|\bar{\gamma}_n^d - \gamma^{d\circ}\|_{\mathcal{H}^d} \leq c \frac{d^p}{n^q}. \quad (2.89)$$

Let us now introduce the following strictly increasing functions, parametrized by the parameter τ introduced in Assumption (iii):

$$L_\tau(s) \triangleq \begin{cases} \kappa_1^- s^{\alpha_1}, & \text{for } s \leq \tau \\ \kappa_2^- s^{\alpha_2}, & \text{for } s > \tau \end{cases} \quad \text{and} \quad U_\tau(s) \triangleq \begin{cases} \kappa_1^+ s^{\beta_1}, & \text{for } s \leq \tau \\ \kappa_2^+ s^{\beta_2}, & \text{for } s > \tau \end{cases}, \quad (2.90)$$

⁴In Remark 2.2, we stressed the meaning of the term “absolute constant,” i.e., a constant that does not depend on any other quantity involved in the context at issue. As regards $\tau, \kappa_1^-, \kappa_1^+, \kappa_2^-, \kappa_2^+$, we have to point out that such “constants” are not absolute. Indeed, they may depend on d .

where $\alpha_1 \geq \beta_1 > 0$, $0 < \alpha_2 \leq \beta_2$, and $s \in \mathbb{R}$, $s \geq 0$. Clearly, owing to Constraints (2.88), the functions $L_\tau(s)$ and $U_\tau(s)$ are continuous and $L_\tau(s) \leq U_\tau(s)$, $\forall s \geq 0$. Then, from (2.86) and (2.87) it follows that

$$L_\tau(\|\gamma^d - \gamma^{d\circ}\|_{\mathcal{H}^d}) \leq F(\gamma^d) - F(\gamma^{d\circ}) \leq U_\tau(\|\gamma^d - \gamma^{d\circ}\|_{\mathcal{H}^d}). \quad (2.91)$$

By Assumption (iii), we have

$$\begin{aligned} U_\tau(\|\bar{\gamma}_n^d - \gamma^{d\circ}\|_{\mathcal{H}^d}) &\geq F(\bar{\gamma}_n^d) - F(\gamma^{d\circ}) \\ &\geq F(\gamma_n^{d\circ}) - F(\gamma^{d\circ}) \geq L_\tau(\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d}) \end{aligned} \quad (2.92)$$

and hence

$$U_\tau(\|\bar{\gamma}_n^d - \gamma^{d\circ}\|_{\mathcal{H}^d}) \geq L_\tau(\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d}). \quad (2.93)$$

By construction, the functions L_τ and U_τ are strictly increasing (hence invertible) and their inverses L_τ^{-1} and U_τ^{-1} are strictly increasing as well. Then, from (2.89) and (2.93) we obtain

$$\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d} \leq L_\tau^{-1}\left[U_\tau\left(c \frac{d^p}{n^q}\right)\right]. \quad (2.94)$$

Now, for the generic τ introduced in Assumption (iii), suppose that n takes on a value such that

$$c \frac{d^p}{n^q} \leq \tau \quad \text{and} \quad \kappa_1^+ \left(c \frac{d^p}{n^q}\right)^{\beta_1} \leq L_\tau(\tau) = \kappa_1^- \tau^{\alpha_1}. \quad (2.95)$$

Then, Inequalities (2.90) and (2.94) provide

$$\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d} \leq \left[\frac{\kappa_1^+}{\kappa_1^-} \left(c \frac{d^p}{n^q}\right)^{\beta_1} \right]^{1/\alpha_1}. \quad (2.96)$$

Furthermore, since $\left[\frac{\kappa_1^+}{\kappa_1^-} \left(c \frac{d^p}{n^q}\right)^{\beta_1}\right]^{1/\alpha_1} \leq \tau$ by construction, from (2.91) and (2.96) we obtain

$$\begin{aligned} F(\gamma_n^{d\circ}) - F(\gamma^{d\circ}) &\leq U_\tau\left(\left[\frac{\kappa_1^+}{\kappa_1^-} \left(c \frac{d^p}{n^q}\right)^{\beta_1}\right]^{1/\alpha_1}\right) \\ &= \frac{(\kappa_1^+)^{1+\beta_1/\alpha_1}}{(\kappa_1^-)^{\beta_1/\alpha_1}} \left(c \frac{d^p}{n^q}\right)^{\beta_1^2/\alpha_1}. \end{aligned} \quad (2.97)$$

By (2.96), (2.97), and the conditions (2.95), some simple algebra shows that the upper bounds

$$\|\gamma_n^{d\circ} - \gamma^{d\circ}\|_{\mathcal{H}^d} \leq \varepsilon \quad (2.98)$$

and

$$F(\gamma_n^{d^o}) - F(\gamma^{d^o}) \leq \varepsilon \quad (2.99)$$

are satisfied for every $n \geq n^\circ(d, \varepsilon)$, where

$$\begin{aligned} n^\circ(d, \varepsilon) \triangleq & \left\lceil \max \left[\left(\frac{c}{\tau} \right)^{1/q} d^{p/q}, \right. \right. \\ & \left(\frac{k_1^+ c^{\beta_1}}{k_1^- \tau^{\alpha_1}} \right)^{1/(q\beta_1)} d^{p/q}, \\ & \left(\frac{k_1^+ c^{\beta_1}}{k_1^-} \right)^{1/(q\beta_1)} \frac{1}{\varepsilon^{\alpha_1/(q\beta_1)}} d^{p/q}, \\ & \left. \left. \left(\frac{(k_1^+)^{1+\beta_1/\alpha_1} c^{\beta_1^2/\alpha_1}}{(k_1^-)^{\beta_1/\alpha_1}} \right)^{\alpha_1/(q\beta_1^2)} \frac{1}{\varepsilon^{\alpha_1/(q\beta_1^2)}} d^{p/q} \right] \right\rceil, \end{aligned}$$

and where the expressions above have been obtained by imposing the constraints

$$c \frac{d^p}{n^q} \leq \tau,$$

$$k_1^+ \left(c \frac{d^p}{n^q} \right)^{\beta_1} \leq k_1^- \tau^{\alpha_1},$$

$$\left[\frac{k_1^+}{k_1^-} \left(c \frac{d^p}{n^q} \right)^{\beta_1} \right]^{1/\alpha_1} \leq \varepsilon,$$

$$\frac{(k_1^+)^{1+\beta_1/\alpha_1}}{(k_1^-)^{\beta_1/\alpha_1}} \left(c \frac{d^p}{n^q} \right)^{\beta_1^2/\alpha_1} \leq \varepsilon,$$

which derive from the previous analysis. Finally, as κ_1^+ is assumed to grow at most polynomially with respect to d and τ and κ_1^- are bounded from 0 by a constant that does not depend on d , we conclude that $n^\circ(d, \varepsilon)$ grows at most as a power of d (compare with (2.52)). Therefore,

$$\{\gamma_n^{d^o}\}_{n=1}^\infty, \quad n \geq n^\circ(d, \varepsilon), \quad (2.100)$$

is a pc P^d -optimizing sequence, thus ending the proof. ■

Example 2.3 From the proof we just gave, it may be interesting to note that, given a generic “threshold” τ such that $\|\gamma^d - \gamma^{d^o}\|_{\mathcal{H}^d} \leq \tau$, $\forall \gamma^d \in \mathcal{S}^d$, the lower and upper bounds (2.86) and (2.87) allow the “piecewise construction” of two suitable

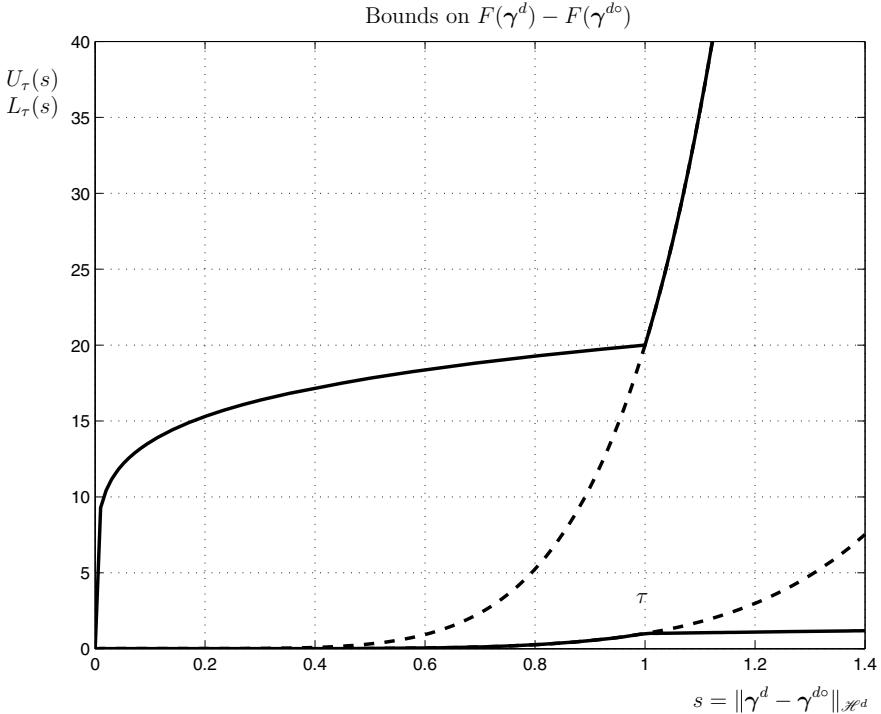


Fig. 2.12 Two examples of functions L_τ and U_τ with $\tau = 1$. Continuous lines: $L_\tau(s) = s^6$ for $s \leq \tau$ and $L_\tau(s) = s^{1/2}$ for $s > \tau$ and $U_\tau(s) = 20s^{1/6}$ for $s \leq \tau$ and $U_\tau(s) = 20s^6$ for $s > \tau$. Dashed lines: $y(s) = s^6$, $\forall s \in [0, \tau)$ and $y(s) = 20s^6$, $\forall s \in [0, \tau)$

continuous strictly increasing functions⁵ $L_\tau(s)$ and $U_\tau(s)$ that constrain the cost functional so that Inequalities (2.91) may hold. In this connection, in Fig. 2.12 a simple example is provided showing that this constraint on F can be made relatively “weaker” than a constraint made of two similar lower and upper bounds defined without taking advantage of the abovementioned “piecewise construction” parametrized by τ . \triangleleft

In order to explain the importance of the “piecewise construction,” let us suppose Assumption (iii) in Theorem 2.2 to be replaced by a single pair of inequalities, that is,

$$\kappa^- (\|\gamma^d - \gamma^{do}\|_{\mathcal{H}^d})^\alpha \leq F(\gamma^d) - F(\gamma^{do}) \leq \kappa^+ (\|\gamma^d - \gamma^{do}\|_{\mathcal{H}^d})^\beta. \quad (2.101)$$

⁵The functions L_τ and U_τ take on the form of the “comparison functions” of class \mathcal{K}_∞ typically used in Lyapunov stability analysis of nonlinear dynamic systems (see, e.g., [30]).

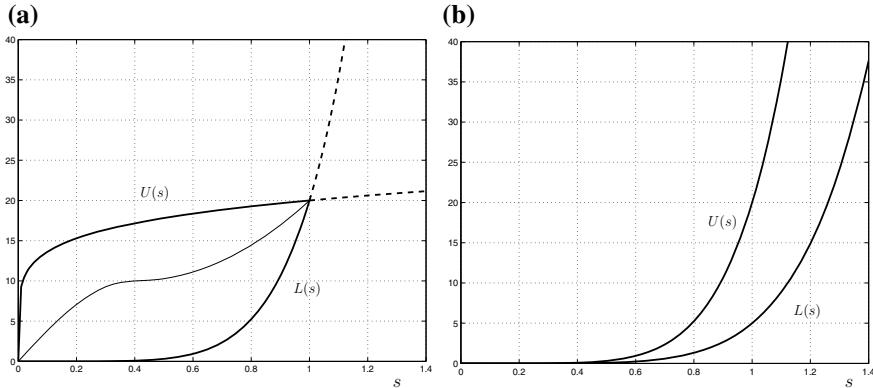


Fig. 2.13 Two examples of lower and upper bound functions without the “piecewise construction.”
a Case (1). **b** Case (2)

We consider two cases, namely, Case (1) $\alpha > 1$ and $0 < \beta < 1$, and Case (2) $\alpha = \beta > 1$, with $\kappa^+ > \kappa^-$. Then, Functions (2.90) simply become

- Case (1): $L(s) = \kappa^- s^\alpha$, $U(s) = \kappa^+ s^\beta$ (see Fig. 2.13a).
- Case (2): $L(s) = \kappa^- s^\alpha$, $U(s) = \kappa^+ s^\beta$ (see Fig. 2.13b).

From Fig. 2.13a, it can be seen that the bounds behave nicely in a neighborhood of the origin, likewise the bounds of Fig. 2.12. In particular, in the origin, the tangents of $L(s)$ and $U(s)$ coincide with the horizontal and vertical axes, respectively. Then, the cost functional F enjoys a large degree of freedom, which enables one to verify Inequalities (2.101) for small values of $s = \|\gamma^d - \gamma^{d\circ}\|_{\mathcal{H}^d}$, without having to impose the functional F to behave like a convex functional. However, for increasing values of s , the constraints become unfeasible (for $s > 1$ in the example shown in Fig. 2.13a, where we have chosen $\kappa^- = \kappa^+ = 20$ and $\alpha = 6$ and $\beta = 1/6$). The opposite occurs for Case (2), shown in Fig. 2.13b, where we have chosen $\kappa^- = 1$, $\kappa^+ = 20$ and $\alpha = \beta = 6$. The bounds $L(s)$ and $U(s)$ allow for unbounded values of s but, for small values of s , the functionals F satisfying Inequalities (2.101) get close to convex ones. Therefore, it can be concluded that the “piecewise construction” enjoys the advantages of Case (1) and Case (2) while avoiding their drawbacks.

Clearly, the construction of the bounds is quite arbitrary, as is the choice of the various constraints in (2.90). Most probably, other piecewise constructions may provide better results as the wider and “better shaped” the area delimited by the bounds, the more likely the possibility that Inequalities (2.86) and (2.87) are verified. This verification is an extremely difficult task even if performed in qualitative terms by trial and error.

2.8.3 Final Remarks

A large part of this chapter has been devoted to the description of two kinds of optimizing sequences of FSP functions (see Table 2.1) and two kinds of approximating sequences of sets of FSP functions (see Table 2.2). Various concepts presented in the next chapters are based on such sequences. They enable us to upgrade sequences of FSP functions belonging to nested sequences of sets (see block 1 in Fig. 2.10) to pc P^d -optimizing sequences (see block 6). This upgrade is made possible by connecting block 1 to block 6 by paths that go from one block to another, thanks to the exploitation of some theorems specifically stated to this end and to the fulfillment of particular hypotheses. Therefore, we have a set of hypotheses made of the ones on which the theorems are based (referring, for instance, to Fig. 2.10, such hypotheses are displayed by the arrows entering the blocks of Theorems 2.1 and 2.2) and of hypotheses provided by available information (see Assumption 2.12 in Fig. 2.10) or hypotheses imposed by ourselves (see Assumption 2.13). Let us denote by \mathcal{I} the complex of all such hypotheses and let us call it *hypothesis set*.

Before describing the path constructed in Sect. 2.8 and reported in Fig. 2.10, it is important to note that the foregoing is quite a general consideration. Clearly, constructing a path connecting block 1 to block 6 is mandatory to give the ERIM the possibility of solving IDO problems by sequences of FDO problems (to be solved via NLP algorithms) hopefully mitigating the curse of dimensionality in high-dimensional settings. In the approach considered in this section, a construction of this type can be performed provided that the hypothesis set \mathcal{I} is verified.

In general, it should be kept in mind that the choice of the hypothesis set \mathcal{I} depends on the analyst's ability to derive efficient theorems for the transition from one block to another. Of course, the “weaker” the hypotheses of a set \mathcal{I} , the greater the possibility that they are verified.

Coming back to the approach addressed in Sect. 2.8, we think it interesting to focus on two hypotheses of the set \mathcal{I} . Referring to Fig. 2.10, the two hypotheses are located in the path that leads us from block 2 to the “target block” 6. In describing this path, we also take the opportunity to sum up and highlight the salient concepts of the chosen approach. The first interesting hypothesis is located in the link from block 2 to block 3, i.e., in the “area of function approximation.” The hypothesis to be verified consists in understanding whether the chosen S^d -approximating sequence of sets $\{\mathcal{S}_n^d\}_{n=1}^\infty$ can become a pc S^d -approximating sequence of sets. As we said earlier, several families of FSP functions benefit from this possibility, called “key results” in Fig. 2.10. Then, verifying the first hypothesis is equivalent to proving a theorem in the area of function approximation. As we shall see in Chap. 3, many results of this kind were demonstrated in the early 90s of the last century by various authors. Note that the choice of the family of FSP functions should be guided by the knowledge of the regularity properties of the functions to be optimized. This knowledge is rarely available.

By proceeding in the description of the path and referring to the assumptions of Theorem 2.2, we see that a link connects block 3 to the block containing Theorem 2.2,

which means that Assumption (ii) is verified. Similarly, the link that goes from the block 5 to the block of Theorem 2.2 means that Assumption (i) is verified. The step through block 5 is due to Theorem 2.1. This theorem “pierces the wall” connecting block 2 to block 5.

The second interesting hypothesis consists in the search of Inequalities (2.86) and (2.87) such that Assumption (iii) is verified. These inequalities are represented symbolically in Fig. 2.10 by the third arrow entering the block of Theorem 2.2. Finally, this theorem leads us in the “target block” 6.

Despite its apparent complexity, Assumption (iii) can be interpreted easily. Once it is sure that the sequence $\{\gamma_n^{d_0}\}_{n=1}^{\infty}$ converges to γ^* with an at-most-polynomial rate and a maximum allowed error not greater than ε , the constants to the left and to the right of Inequalities (2.86) and (2.87) force the sequence $\{F_n^{d_0}\}_{n=1}^{\infty}$ to also converge to F^{d_0} with a maximum error not greater than ε . All this must take place with a model complexity n that grows at most polynomially when the dimension d increases. The condition that must be met is $n \geq n^*(d, \varepsilon)$.

2.9 Notes on the Practical Application of the ERIM

As already remarked in Sect. 1.5, Problem P is characterized by three elements, namely, the infinite-dimensional linear space \mathcal{H} , the set $\mathcal{S} \subseteq \mathcal{H}$ of admissible functions, and the objective functional $F: \mathcal{S} \rightarrow \mathbb{R}$. A similar remark holds for Problem PM. The procedure of approximate solution that we have outlined in this chapter is based on the definitions of the infinite-dimensional linear space \mathcal{H} and the set $\mathcal{S} \subseteq \mathcal{H}$ of admissible functions. Examples of \mathcal{H} are the space of square-integrable functions and the space of continuous functions, endowed with the respective norms. Examples of set \mathcal{S} of admissible functions are balls in such spaces (i.e., sets of functions defined by upper bounds on their norms).

However, the IDO problems that we shall face in Chaps. 6–10 will be stated in very general terms and no specific instances of \mathcal{H} and \mathcal{S} will be provided in those chapters. So, in a strict and formal sense, they are not instances of Problems P or PM. Nevertheless, it will be clear that they are IDO problems and we shall formally apply to them the procedure of reduction to NLP problems detailed in this chapter. Hence, with some formal abuse, we shall refer to the IDO problems arising in Chaps. 6–10 as “instances of Problem P” or “instances of Problem PM.” The accuracy of approximate solutions obtained via FSP functions will be investigated *supposing that* the optimal ones belong to suitable subsets of certain normed linear spaces. In doing so, we shall adopt an “engineering-oriented” point of view, in which some rigor and technicalities are sacrificed without substantial consequences on applications, thus benefitting the less “mathematically oriented” readers.

Instead, the approach keeping a full mathematical coherence with the way of proceeding described in this chapter requires (i) the definition of the normed linear space \mathcal{H} and the set $\mathcal{S} \subseteq \mathcal{H}$ of admissible functions *before* searching for suboptimal solutions and (ii) the use of approximation schemes whose properties are tailored

to such space and set. This can be done by introducing specific hypotheses in the statements of the IDO problems (e.g., on the functionals to be maximized/minimized and on the system dynamics), which allow one to prove that the (unknown) optimal solutions enjoy certain smoothness properties. Such an approach is pursued in a series of papers co-authored by some of the authors of this book. In such works, several kinds of IDO problems are faced: abstract IDO [15, 18, 26, 27, 29, 34], N -stage nonlinear optimal control [11, 20], team optimization [21–23], principal component analysis [16, 17], learning from data [19], and optimal estimation [1, 2].⁶ For instance, in [2] an optimal state estimation problem is addressed, where the ambient space \mathcal{H} is made of continuous functions and the set $\mathcal{S} \subseteq \mathcal{H}$ is defined by imposing on such functions constraints such as uniform boundedness and equi-Lipschitz properties.

To sum up, in Chaps. 6–10 no restrictive hypotheses will be specified on the statements of the IDO problems. We shall suppose that optimal solutions exist with certain smoothness properties and we shall search for suboptimal ones in the form of FSP functions with increasing numbers of parameters. Instead, in the abovementioned papers hypotheses are made (e.g., as already said, on the functional or the system dynamics), which guarantee the existence of optimal solutions with smoothness properties allowing to approximate them via polynomially complex FSP functions.

The two approaches are sort of complementary. In the latter, the emphasis is on the analysis of the errors of approximate optimization, which requires to introduce very specific mathematical regularity conditions. This point of view is of interest mainly to “mathematically oriented designers” or readers focusing on a rigorous mathematical analysis. In the second part of this book, instead, such an analysis is not dealt with because of smaller interest and little impact on the considered challenging engineering applications. More specifically, a typical approach of “engineering-oriented designers” is followed in which the IDO problems are solved by an empirical procedure, where the errors of approximate optimization are estimated by progressively incrementing the model complexity.

References

1. Alessandri A, Cervellera C, Sanguineti M (2007) Functional optimal estimation problems and their solution by nonlinear approximation schemes. *J Optim Theory Appl* 134:445–466
2. Alessandri A, Gnecco G, Sanguineti M (2010) Minimizing sequences for a family of functional optimal estimation problems. *J Optim Theory Appl* 147:243–262
3. Alessandri A, Gaggero M, Zoppoli R (2012) Feedback optimal control of distributed parameter systems by using finite-dimensional approximation schemes. *IEEE Trans Neural Netw Learn Syst* 23:984–996

⁶It is worth mentioning also the work [31], where suboptimal feedback control laws are searched for dynamic systems under LQ assumptions via the Ritz method. However, in [31] the point of view is different: the authors aim at deriving, for an optimal control problem whose solution is known in closed form, an approximate control law taking on a simple structure.

4. Alt W (1984) On the approximation of infinite optimization problems with an application to optimal control problems. *Appl Math Optim* 12:15–27
5. Barron AR (1992) Neural net approximation. In: Narendra KS (ed) *Proceedings of the 7th Yale workshop on adaptive and learning systems*. Yale University Press, pp 69–72
6. Barron AR (1993) Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Trans Inf Theory* 39:930–945
7. Breiman L (1993) Hinging hyperplanes for regression, classification, and function approximation. *IEEE Trans Inf Theory* 39:993–1013
8. Dal Maso G (1993) *Introduction to gamma-convergence*. Springer
9. Darken C, Donahue M, Gurvits L, Sontag E (1993) Rate of approximation results motivated by robust neural network learning. In: *Proceedings of the sixth annual ACM conference on computational learning theory*. ACM, pp 303–309
10. Donahue M, Gurvits L, Darken C, Sontag E (1997) Rates of convex approximation in non-Hilbert spaces. *Constr Approx* 13:187–220
11. Gaggero M, Gnecco G, Sanguineti M (2014) Suboptimal policies for stochastic N -stage optimization problems: accuracy analysis and a case study from optimal consumption. In: El Ouardighi F, Kogan K (eds) *Models and methods in economics and management*. Springer, pp 27–50
12. Gelfand IM, Fomin SV (1963) *Calculus of variations*. Prentice Hall
13. Girosi F, Poggio T (1990) Networks and the best approximation property. *Biol Cybern* 63:169–176
14. Girosi F, Anzellotti G (1993) Rates of convergence for Radial Basis Functions and neural networks. In: Mammone RJ (ed) *Artificial neural networks for speech and vision*. Chapman & Hall, pp 97–113
15. Giulini S, Sanguineti M (2009) Approximation schemes for functional optimization problems. *J Optim Theory Appl* 140:33–54
16. Gnecco G, Sanguineti M (2009) Accuracy of suboptimal solutions to kernel principal component analysis. *Comput Optim Appl* 42:265–287
17. Gnecco G, Sanguineti M (2010) Error bounds for suboptimal solutions to kernel principal component analysis. *Optim Lett* 4:197–210
18. Gnecco G, Sanguineti M (2010) Estimates of variation with respect to a set and applications to optimization problems. *J Optim Theory Appl* 145:53–75
19. Gnecco G, Sanguineti M (2010) Regularization techniques and suboptimal solutions to optimization problems in learning from data. *Neural Comput* 22:793–829
20. Gnecco G, Sanguineti M (2010) Suboptimal solutions to dynamic optimization problems via approximations of the policy functions. *J Optim Theory Appl* 146:764–794
21. Gnecco G, Sanguineti M (2011) Team optimization problems with Lipschitz continuous strategies. *Optim Lett* 5:333–346
22. Gnecco G, Sanguineti M (2012) New insights into Witsenhausen’s counterexample. *Optim Lett* 6:1425–1446
23. Gnecco G, Sanguineti M, Gaggero M (2012) Suboptimal solutions to team optimization problems with stochastic information structure. *SIAM J Optim* 22:212–243
24. Gurvits L, Koiran P (1997) Approximation and learning of convex superpositions. *J Comput Syst Sci* 55:161–170
25. Jones LK (1992) A simple lemma on greedy approximation in Hilbert space and convergence rates for projection pursuit regression and neural network training. *Ann Stat* 20:608–613
26. Kainen P, Kůrková V, Sanguineti M (2003) Minimization of error functionals over variable-basis functions. *SIAM J Optim* 14:732–742
27. Kainen PC, Kůrková V (2005) Rates of minimization of error functionals over Boolean variable-basis functions. *J Math Model Algorithms* 4:355–368
28. Kainen PC, Kůrková V (2009) Complexity of Gaussian radial basis networks approximating smooth functions. *J Complex* 25:63–74
29. Kainen PC, Kůrková V (2012) Dependence of computational models on input dimension: tractability of approximation and optimization tasks. *IEEE Trans Inf Theory* 58:1203–1214

30. Khalil HK (1996) Nonlinear systems. Prentice-Hall
31. Kleinman DL, Athans M (1968) The design of suboptimal linear time-varying systems. IEEE Trans Autom Control 13:150–159
32. Knuth DE (1976) Big omicron and big omega and big theta. *SIGACT News*
33. Kůrková V (2008) Minimization of error functionals over perceptron networks. Neural Comput 20:252–270
34. Kůrková V, Sanguineti M (2005) Error estimates for approximate optimization by the Extended Ritz Method. SIAM J Optim 15:461–487
35. Kůrková V, Savický P, Hlavácková K (1998) Representations and rates of approximation of real-valued Boolean functions by neural networks. Neural Netw 11:651–659
36. Makovoz Y (1996) Random approximants and neural networks. J Approx Theory 85:98–109
37. Mhaskar HN (2004) On the tractability of multivariate integration and approximation by neural networks. J Complex 20:561–590
38. Parisini T, Zoppoli R (2009) Connections between approximating sequences and optimizing sequences in the Extended Ritz Method. Technical Report 2009.1, DIPTEM, Università di Genova
39. Traub JF, Werschulz AG (1999) Complexity and information. Cambridge University Press
40. Wasilkowski GW, Woźniakowski H (2001) Complexity of weighted approximation over \mathbb{R}^d . J Complex 17:722–740
41. Woźniakowski H (1994) Tractability and strong tractability of linear multivariate problems. J Complex 10:96–128

Chapter 3

Some Families of FSP Functions and Their Properties



In this chapter, we present some properties of fixed-structure parametrized (FSP) functions that give insights into the effectiveness of the methodology of approximate solution for the infinite-dimensional optimization (IDO) problems discussed in this book. The chapter contains five major parts, devoted to the structure of some widespread FSP functions (Sects. 3.1–3.5); the density property (Sect. 3.6); rates of approximation (Sects. 3.7 and 3.8); a comparison, from the point of view of the curse of dimensionality, between the classical Ritz method and the extended Ritz method (ERIM) (Sect. 3.9); and rates of approximate optimization by the ERIM (Sect. 3.10).

In Sects. 3.1 and 3.2, the structures of linear combinations of fixed-basis functions (LCFBFs) and one-hidden-layer networks (OHL networks) are discussed, respectively. As far as the latter is concerned, the tensor-product construction, the ridge construction, and the radial construction are described. Section 3.3 considers multi-hidden-layer (MHL) networks and provides some historical notes and clarifications on the terminologies “ridge OHL networks,” “multilayer networks,” and the neural network parlance. Finally, Sect. 3.5 is devoted to kernel smoothing models (KSMs).

The density property of OHL networks (i.e., the capability, using a sufficiently high number of basis functions, of approximating to every desired degree of accuracy all “reasonable” functions encountered in applications) is investigated in Sect. 3.6. We first recall the relationships between \mathcal{C} - and \mathcal{L}_p -density properties (Sect. 3.6.1) and briefly discuss the case of multi-hidden-layers and multiple outputs (Sect. 3.6.4). Section 3.6.2 shows that, by imposing mild conditions on the basis functions, ridge OHL networks turn out to be \mathcal{C} - and \mathcal{L}_2 -approximating networks. Similar density results for OHL networks with radial basis functions are presented in Sect. 3.6.3.

“Approximation rates” measure the speed at which approximation error goes to zero as n tends to infinity and may depend also on the dimension d . Proving

density may involve existential or constructive methods, while the latter type makes it possible to provide upper bounds on the minimum number of basis functions needed to guarantee a desired approximation accuracy. Unfortunately, however, such bounds are typically discouragingly large. In this respect, we recall from Sect. 2.6 that, for a given sequence of function approximation Problems PA_n^d , we wish to understand whether families of \mathcal{G}^d -approximating FSP functions (see Remark 2.4) exist that enable us to achieve a desired accuracy in approximating functions γ^d in Problems PA_n^d by using “moderate” model complexities n when the dimension d increases.

Estimating approximation rates of OHL networks is the objective of Sects. 3.7 and 3.8. In Sect. 3.7.1, a tool that plays a basic role in deriving an upper bound of order $O(n^{-1/2})$ on the worst-case approximation error for OHL networks with model complexity n is presented, namely, the Maurey–Jones–Barron theorem (MJB theorem, for short). Section 3.7.2 details a case study within the widespread class of ridge OHL networks with sigmoidal basis functions. This is done by applying the MJB theorem to such a kind of OHL networks. A hint on the approximation accuracy of MHL networks is contained in Sect. 3.7.4. Finally, in Sect. 3.7.3, we discuss limitations on approximation capabilities of OHL networks in terms of lower bounds on approximation rates.

Extensions of the MJB theorem are investigated in Sect. 3.8. Section 3.8.1 addresses the possibility of achieving faster approximation rates, whereas Sect. 3.8.2 is devoted to a more abstract extension, in terms of a norm tailored to approximation by computational units from a set \mathcal{G} , called \mathcal{G} -variation norm. In Sect. 3.8.3, the possibility of estimating \mathcal{G} -variation in terms of classical and spectral norms is discussed. Considering sets \mathcal{G} made up of different kinds of parametrized computational units and exploiting the above-mentioned estimates of \mathcal{G} -variation, various families of OHL networks and the corresponding sets of functions that can be approximated by OHL networks at a rate of order $O(n^{-1/2})$ are presented in Sect. 3.8.4.

Section 3.9 is devoted to a comparison between the classical Ritz method and the ERIM, from the point of view of the accuracy of the suboptimal solutions that they provide and the curse of dimensionality. After recalling some concepts on Hilbert spaces in Sects. 3.9.1 and 3.9.2 we address a family of IDO problems, in which the functional to be optimized is quadratic (by the way, this is just the class of problems for which the Ritz method was originally conceived). The application of the Ritz method to such problems is described in Sects. 3.9.3 and 3.9.4 we show that it incurs the curse of dimensionality, whereas the ERIM may not suffer from this drawback.

Finally, Sect. 3.10 further investigates P^d -optimizing sequences (see Sect. 2.8) within the ERIM. In that section, we give upper bounds on the error rate for approximate solution of convex IDO problems, which mirror those provided by the MJB Theorem.

3.1 Linear Combinations of Fixed-Basis Functions

The choice of the FSP functions $\gamma(\mathbf{x}, \mathbf{w}_n)$ defined in Sect. 2.1 cannot but be arbitrary. A simple and natural choice is to use, for each component $\gamma_j(\mathbf{x}, \mathbf{w}_n)$, a linear combination of preassigned basis functions of the form

$$\gamma_j(\mathbf{x}, \mathbf{w}_n) = \sum_{i=1}^n c_{ij} \varphi_i(\mathbf{x}), \quad c_{ij} \in \mathbb{R}, \quad j = 1, \dots, m, \quad (3.1)$$

where $\varphi_1, \dots, \varphi_n$ are the first n functions of a sequence of fixed-basis functions, and the components of \mathbf{w}_n are all the coefficients c_{ij} , that is,

$$\mathbf{w}_n \triangleq \text{col}(c_{ij} : i = 1, \dots, n; j = 1, \dots, m).$$

The functions φ_i in (3.1) are called *fixed-basis functions*. Examples of them are the terms of the classical polynomial expansions (e.g., the Legendre, Laguerre, and Hermite polynomials; see, for instance, [74]). $\gamma(\mathbf{x}, \mathbf{w}_n)$ takes on the form

$$\gamma(\mathbf{x}, \mathbf{w}_n) \triangleq \text{col} \left[\sum_{i=1}^n c_{ij} \varphi_i(\mathbf{x}) : j = 1, \dots, m \right] \quad (3.2)$$

and the number of “free” parameters (i.e., the dimension of \mathbf{w}_n) is simply given by (see (2.1))

$$\mathcal{N}(n) = nm.$$

Definition 3.1 Functions (2.3) that take on the structure (3.2) are called “linear combinations of fixed-basis functions” (LCFBFs). \triangleleft

For any positive integer n , the sequence of sets $\{\mathcal{A}_n\}_{n=1}^\infty$ composed of all the functions $\gamma(\mathbf{x}, \mathbf{w}_n)$ in (3.2) when the parameter vector \mathbf{w}_n varies in $\mathbb{R}^{\mathcal{N}(n)}$ has the nested structure (2.4). The model complexity of the LCFBFs is given by n . Clearly, they are strictly related to the Ritz method. The LCFBFs are linear approximators, since they depend linearly on their parameters (i.e., the coefficients of the linear combinations). Classical linear approximators such as algebraic and trigonometric polynomials are examples of linear \mathcal{C} - and \mathcal{L}_p -approximating FSP functions (see Remark 2.4). If the basis functions φ_i are linearly independent, then n is the dimension of the subspace that they span; otherwise, the subspace generated by the basis functions has dimension less than n . For example, in approximating one-variable real-valued functions by algebraic polynomials of order at most $n - 1$, an n -dimensional space is generated by the first n elements of the set $\{x^i : i \in \mathbb{Z}^+\}$. So, an approximation by LCFBFs corresponds to a classical approximation by elements of a subspace. Given n basis functions $\varphi_1, \dots, \varphi_n$, the space spanned by these functions is denoted by

$$\text{span}\{\varphi_1, \dots, \varphi_n\}.$$

3.2 One-Hidden-Layer Networks

3.2.1 The Structure of OHL Networks

Rather a natural extension of the FSP functions (3.2) consists in inserting some “free” parameters to be optimized in the basis functions. If we denote the vector of these “free” parameters by κ_i , then the new basis functions take on the form $\varphi_i(\mathbf{x}, \kappa_i)$ instead of $\varphi_i(\mathbf{x})$. As the vector κ_i increases the “flexibility” of the basis functions quite a lot, we avoid using different basis functions $\varphi_1(\mathbf{x}, \kappa_1), \dots, \varphi_n(\mathbf{x}, \kappa_n)$. Then, in the following paragraphs we shall refer only to one fixed-structure function $\varphi(\mathbf{x}, \kappa_i)$, hence, dropping the subscript i in φ_i . We choose the function φ in such a way that $\varphi(\cdot, \kappa_i) \in \mathcal{H}$ for any $\kappa_i \in \mathbb{R}^k$, $i = 1, \dots, n$, and we call the functions $\varphi(\cdot, \kappa_i)$ *parametrized basis functions* (in contrast to the fixed-basis functions φ_i in (3.1)). Typically, an additional constant basis function is also used. Thus,

$$\gamma_j(\mathbf{x}, \mathbf{w}_n) = \sum_{i=1}^n c_{ij} \varphi(\mathbf{x}, \kappa_i) + b_j, \quad c_{ij}, b_j \in \mathbb{R}, \quad \kappa_i \in \mathbb{R}^k, \quad j = 1, \dots, m, \quad (3.3)$$

where the vector of parameters to be optimized is given by

$$\mathbf{w}_n \triangleq \text{col}(c_{ij}, b_j, \kappa_i : i = 1, \dots, n, j = 1, \dots, m),$$

and $b_j \in \mathbb{R}$ is called *bias*.¹ Then, we can define

$$\gamma(\mathbf{x}, \mathbf{w}_n) \triangleq \text{col} \left[\sum_{i=1}^n c_{ij} \varphi(\mathbf{x}, \kappa_i) + b_j : j = 1, \dots, m \right]. \quad (3.4)$$

For any positive integer n , the sequence of sets $\{\mathcal{A}_n\}_{n=1}^\infty$ – which is composed of all the functions $\gamma(\cdot, \mathbf{w}_n)$ given by (3.4) when the parameter vector \mathbf{w}_n varies in $\mathbb{R}^{N(n)}$ – has the nested structure (2.4). As linear combinations of the form (3.4) will play an important role throughout the book, we give the following definition.

Definition 3.2 Functions (2.3) that take on the structure (3.4) are called “one-hidden-layer networks” (OHL networks). \triangleleft

We borrow the term “one-hidden-layer network” from the parlance of neural networks, as Functions (3.4) take on the same structure as feedforward neural networks with only one hidden layer and linear output activation units (see Sect. 3.2.3). The structure of OHL networks is depicted in Fig. 3.1.

¹Sometimes, in (3.3) and (3.4), the bias is omitted, see Remark 3.2.

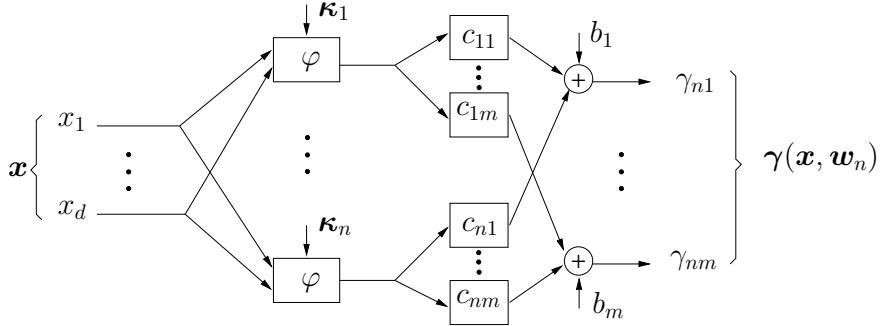


Fig. 3.1 Structure of OHL networks; x_1, \dots, x_d denote the components of the vector \mathbf{x}

For OHL networks, the dimension of the parameter vector \mathbf{w}_n , i.e., the number of “free” parameters, is given by

$$\mathcal{N}(n) = n(k + m) + m. \quad (3.5)$$

This motivates the choice of the number n of parametrized basis functions for quantifying the model complexity of OHL networks. Note that $\mathcal{N}(n)$ is an affine function of n . Indeed, in most applications, $n(k + m) \gg m$.

The difference between LCFBFs and OHL networks is pointed out in the following example, which refers to the case $m = 1$.

Example 3.1 We recall that a trigonometric polynomial of degree at most n is a linear combination of the functions $\{\cos(2\pi i x) : i = 0, 1, \dots, n\}$ and $\{\sin(2\pi i x) : i = 1, \dots, n\}$ (the index $i = 0$ is used – only in the cosines – to represent the unit constant function). Such $2n + 1$ elements form a fixed basis, whose linear combinations represent LCFBFs where the basis functions are cosines and sines with frequencies multiple of 2π plus the constant function. Instead of considering only frequencies that are multiples of 2π , let us now take all possible frequencies into account, i.e., let us define the sets of functions $\{\cos(2\pi i x) : i \in \mathbb{R}\}$ and $\{\sin(2\pi i x) : i \in \mathbb{R} \setminus \{0\}\}$. In correspondence with each choice for $i \in \mathbb{R}$ (or $i \in \mathbb{R} \setminus \{0\}$), a cosine (or sine) with a different frequency is generated. Such a function can be considered as an element of a set of parametrized basis functions obtained by varying the free parameter $i \in \mathbb{R}$ (or $i \in \mathbb{R} \setminus \{0\}$) to which the frequency $2\pi i$ corresponds. Taking $2n$ such functions plus a bias, the total number of parameters is not equal to $2n + 1$ – as in the case of linear approximation by cosines and sines discussed above – but is given by $4n + 1$ ($2n$ coefficients of the linear combination, plus $2n$ frequencies, plus the bias). An obvious advantage of this parametrization is that, this way, it is possible to represent every cosine (or sine) with a single element. If one would use a fixed basis, either a possibly large model complexity would be needed to represent such a function, or this would be not possible at all (depending on i being multiple or not of 2π). In the extension of the parametrization to the case of functions of d variables, there

are $2n(d + 1) + 1$ parameters, d being the number of free parameters in the inner product $\mathbf{i}^\top \mathbf{x}$, with $\mathbf{x} \in \mathbb{R}^d$, $\mathbf{i} \in \mathbb{R}^d$ (or, $\mathbf{i} \in \mathbb{R}^d \setminus \{\mathbf{0}\}$), for each cosine (or, sine) with frequency vector $2\pi \mathbf{i}$. \triangleleft

In general, the presence of the “inner” parameter vectors κ_i in the basis functions “destroys” the linearity of the representation (3.3). This not only occurs when each variable-basis function $\varphi(\cdot, \kappa_i)$ depends nonlinearly on the parameter vector κ_i but may happen even when the dependence is linear. In conclusion, unlike LCFBFs, OHL networks are nonlinear approximators.

Remark 3.1 It is worth observing that, in a sense, LCFBFs including a constant basis function may also be called “linear” OHL networks – since they are derived from (3.3) after “restoring” the linearity of the representation, once the inner parameters κ_i have been fixed, while the OHL networks defined by (3.3) may be called “nonlinear” OHL networks. However, in order to distinguish the two situations, in this book we always use the term “LCFBFs” for the first case, and the expression “OHL networks” for the second one (i.e., we always consider an OHL network as nonlinear). \triangleleft

Remark 3.2 In the book, we shall use sometimes the expression “OHL network” also for the situation in which the bias terms b_j in (3.3) and (3.4) are absent. In this case, the number of free parameters is reduced to $n(k + m)$. \triangleleft

To the more than reasonable question as to why one should pass from the simple LCFBFs to the more complex OHL networks, we answer that this complication (which materializes the change from the classical Ritz method to the ERIM) is introduced in the hope of being able to derive (equally accurate) approximate solutions to Problem P by using a much smaller number of parameters to be optimized when addressing IDO problems in high-dimensional contexts and, in particular, to mitigate the curse of dimensionality. We shall address this very important issue later in this chapter.

3.2.2 A More Abstract View

Let us now refer to the structure (3.3) of OHL networks (for simplicity, without including the bias terms). Since a function $\gamma : \mathbb{R}^d \rightarrow \mathbb{R}^m$ can be implemented by m mappings $\gamma_j : \mathbb{R}^d \rightarrow \mathbb{R}$, $j = 1, \dots, m$ (see (3.4)), we now focus on single-output OHL networks, i.e., we let $m = 1$.

We have seen that Networks (3.3) are made of n basis functions having a fixed structure given by $\varphi(\mathbf{x}, \kappa_i) : \mathbb{R}^d \times W \rightarrow \mathbb{R}$, $W \subseteq \mathbb{R}^k$ and a single linear output unit. Then, these OHL networks generate all functions that can be written as $\sum_{i=1}^n c_i \varphi(\cdot, \kappa_i)$, where $\kappa_i \in W \subseteq \mathbb{R}^k$, i.e., all functions belonging to the set

$$\mathcal{A}_n \triangleq \left\{ \sum_{i=1}^n c_i \varphi(\cdot, \kappa_i) : c_i \in \mathbb{R}, \kappa_i \in W \subseteq \mathbb{R}^k \right\}.$$

To gain some further insights into the structure of these sets, we first introduce some notation. Let \mathcal{G} be a linear space of functions and $\mathcal{G} \subset \mathcal{G}$. The *linear span* of \mathcal{G} , denoted by $\text{span } \mathcal{G}$, is the set of all linear combinations of functions of \mathcal{G} , i.e.,

$$\text{span } \mathcal{G} \triangleq \left\{ \sum_{i=1}^n c_i g_i : c_i \in \mathbb{R}, g_i \in \mathcal{G}, n \in \mathbb{Z}^+ \right\}.$$

For each $n \in \mathbb{Z}^+$, $\text{span}_n \mathcal{G}$ denotes the set of all linear combinations of at most n functions of \mathcal{G} , i.e.,

$$\text{span}_n \mathcal{G} \triangleq \left\{ \sum_{i=1}^n c_i g_i : c_i \in \mathbb{R}, g_i \in \mathcal{G} \right\}.$$

Note that, typically, $\text{span}_n \mathcal{G}$ is not a subspace. Indeed, it is equal to the union of all at most n -dimensional subspaces spanned by n -tuples of elements of \mathcal{G} . In this book (see Sect. 3.7.1 in particular), we shall also need to consider the following sets, which are obtained, respectively, from $\text{span } \mathcal{G}$ and $\text{span}_n \mathcal{G}$ by considering only convex combinations of elements of \mathcal{G} :

$$\text{conv } \mathcal{G} \triangleq \left\{ \sum_{i=1}^n c_i g_i : c_i \in [0, 1], \sum_{i=1}^n c_i = 1, g_i \in \mathcal{G}, n \in \mathbb{Z}^+ \right\},$$

$$\text{conv}_n \mathcal{G} \triangleq \left\{ \sum_{i=1}^n c_i g_i : c_i \in [0, 1], \sum_{i=1}^n c_i = 1, g_i \in \mathcal{G} \right\}.$$

The set $\text{conv } \mathcal{G}$ is called *convex hull* of \mathcal{G} . Moreover, let

$$\mathcal{G}_\varphi \triangleq \left\{ \varphi(\cdot, \kappa) : \kappa \in W \subseteq \mathbb{R}^k \right\}$$

denote the set of functions obtained by varying a vector $\kappa \in W \subseteq \mathbb{R}^k$ in the fixed-structure function φ . For example,

$$\mathcal{G}_{\sin} \triangleq \left\{ f(\mathbf{x}) = \sin(\mathbf{x}^\top \mathbf{w} + \theta), \mathbf{w} \in \mathbb{R}^d, \theta \in \mathbb{R} \right\}$$

is the set of all sines with arbitrary frequencies and phases. Then, the sets of the sequence $\{\mathcal{A}_n\}_{n=1}^\infty$, corresponding to OHL networks with a parametrized basis function φ , can also be written as

$$\mathcal{A}_n = \text{span}_n \mathcal{G}_\varphi = \left\{ \sum_{i=1}^n c_i \varphi(\cdot, \kappa_i) : c_i \in \mathbb{R}, \kappa_i \in W \subseteq \mathbb{R}^k \right\}.$$

The total number of free parameters in $\text{span}_n \mathcal{G}_\varphi$ depends on the model complexity n and on the number of parameters of the functions in \mathcal{G}_φ . In general, if n parametrized basis functions are used, then there are $n(k + 1)$ free parameters, as each of them is obtained by varying a k -dimensional parameter vector (see Sect. 3.2.3, where several types of OHL networks are presented). For example, $k = d + 1$ in the approximation by OHL networks corresponding to trigonometric polynomials with free frequencies and phases.

3.2.3 Tensor-Product, Ridge, and Radial Constructions

There are many possibilities to choose the parametrized basis functions $\varphi(\mathbf{x}, \kappa_i)$. Following [139], below we describe three widely used methods for such a construction. They obtain the multivariable parametrized basis functions $\varphi(\cdot, \kappa_i)$ from a single one-dimensional function $h : \mathbb{R} \rightarrow \mathbb{R}$ or from d one-dimensional functions h_1, \dots, h_d .

- (i) *Tensor-product construction.* Here one builds $\varphi(\mathbf{x}, \kappa_i)$ as the product of d one-dimensional functions, be they identical or not, i.e., for any $\mathbf{x} \triangleq \text{col}(x_1, \dots, x_d) \in \mathbb{R}^d$

$$\varphi(\mathbf{x}, \kappa_i) = h_1(x_1, \tau_{i1}) \times \cdots \times h_d(x_d, \tau_{id}),$$

where $\tau_{is} \in \mathbb{R}^{l_{is}}$ is a parameter vector and $\kappa_i \triangleq \text{col}(\tau_{is} : s = 1, \dots, d)$. This is the usual procedure to build polynomial and spline bases for grid-based approximations.

- (ii) *Ridge construction.* One “shrinks” the d -dimensional vector \mathbf{x} into a one-dimensional variable by taking the inner product, i.e.,

$$\varphi(\mathbf{x}, \kappa_i) = h(\mathbf{x}^\top \boldsymbol{\alpha}_i + \beta_i), \quad (3.6)$$

where $\kappa_i \triangleq \text{col}(\boldsymbol{\alpha}_i, \beta_i) \in \mathbb{R}^{d+1}$. Ridge functions are defined as the mappings that are constant for all $\mathbf{x} \in \mathbb{R}^d$ such that $\mathbf{x}^\top \boldsymbol{\alpha}$ is constant, for some $\boldsymbol{\alpha} \in \mathbb{R}^d$. The vectors $\boldsymbol{\alpha}_i \in \mathbb{R}^d \setminus \{\mathbf{0}\}$ are called *directions*. Thus, each ridge function results from the composition of a multivariable function having a particularly simple form, i.e., the inner product $\mathbf{x}^\top \boldsymbol{\alpha}$, with a function h depending on a unique variable. Simple examples of widespread ridge functions are $e^{\mathbf{x}^\top \boldsymbol{\alpha}}$ and $\mathbf{x}^\top \boldsymbol{\alpha}$. As noted in [126], the name “ridge function” was coined by Logan and Shepp in 1975 [108].² However, ridge functions have been studied for a long time under

²The rationale for introducing such functions in [108], which is a seminal paper in computerized tomography, was the reconstruction of a multivariable function from the values of its integrals

the name of *plane waves* (see, for instance, [25, 75]), coming from various problems in physics. It should also be noted that ridge functions are studied in statistics, where they are often used in the context of *projection pursuit*.³

As regards the function h used to construct Mapping (3.6), a typical choice is a *sigmoidal function*, i.e., a bounded measurable function on the real line such that $\lim_{t \rightarrow +\infty} h(t) = 1$ and $\lim_{t \rightarrow -\infty} h(t) = 0$.⁴ Typical examples of sigmoidal functions are:

- The *Heaviside function* (in the neural network literature also known as the *threshold function*) $h_{Heav}(t) \triangleq \begin{cases} 0, & \text{if } t < 0 \\ 1, & \text{if } t \geq 0 \end{cases}$.
- The *logistic sigmoid* $h_{log}(t) \triangleq \frac{1}{1 + e^{-t}}$.
- The *hyperbolic tangent* $h_{hyp}(t) \triangleq \tanh(t)$.
- The piecewise-linear function $h_{piece}(t) \triangleq \begin{cases} 0, & \text{if } t \leq -1 \\ \frac{t+1}{2}, & \text{if } -1 \leq t \leq 1 \\ 1, & \text{if } t \geq 1 \end{cases}$.
- The *Gaussian sigmoid* $h_{Gauss}(t) \triangleq \frac{1}{(2\pi)^{1/2}} \int_{-\infty}^t e^{-s^2/2} ds$.
- The arctan sigmoid $h_{atan}(t) \triangleq \frac{1}{\pi} \arctan(t) + \frac{1}{2}$.

Feedforward neural networks with one hidden layer and linear output activation functions are examples of OHL networks based on the ridge construction. Their j -th components are expressed as

$$\gamma_j(\mathbf{x}, \mathbf{w}_n) = \sum_{i=1}^n c_{ij} h(\mathbf{x}^\top \boldsymbol{\alpha}_i + \beta_i) + b_j, \quad (3.7)$$

where h is a sigmoidal function, as defined above, and $\boldsymbol{\kappa}_i \triangleq \text{col}(\boldsymbol{\alpha}_i, \beta_i)$. Linear combinations of sinusoidal functions with variable frequencies and phases [77]

along certain planes or lines. If such planes or lines are parallel to each other, then each of the abovementioned integrals can be considered as a ridge function with an appropriate direction.

³Projection pursuit algorithms investigate approximations of a d -variable function by functions having the form $\sum_{i=1}^n g_i(\mathbf{x}^\top \boldsymbol{\alpha}_i)$, where $\boldsymbol{\alpha}_i \in \mathbb{R}^d$ and $g_i : \mathbb{R} \rightarrow \mathbb{R}$ have to be suitably chosen (see e.g., [34, 39]).

⁴We have reported here the most widespread definition of sigmoidal function (see, for instance, [6, 127] and the references therein). However, in the literature there is a certain lack of consistency in the terminology. For example, some authors also require continuity and/or monotonicity (or even strict monotonicity) of h on \mathbb{R} . Others consider finite values for the limits to $+\infty$ and $-\infty$ (e.g., 1 and -1 , respectively). As the algorithms that we shall present in the second part of the book to optimize the parameters of OHL networks are gradient-based, whenever the derivatives of the basis functions are involved we shall implicitly suppose that the sigmoidal functions are continuously differentiable.

are another example of ridge construction OHL networks. Their components are given by

$$\gamma_j(\mathbf{x}, \mathbf{w}_n) = \sum_{i=1}^n c_{ij} \sin(\boldsymbol{\omega}_i^\top \mathbf{x} + \theta_i) + b_j, \quad (3.8)$$

where $\boldsymbol{\kappa}_i \triangleq \text{col}(\boldsymbol{\omega}_i, \theta_i)$.

- (iii) *Radial construction.* The vector \mathbf{x} is shrunk into a scalar variable by means of a norm on \mathbb{R}^d . A typical choice is a weighted l_2 -norm, i.e.,

$$\varphi(\mathbf{x}, \boldsymbol{\kappa}_i) = h(|\mathbf{x} - \boldsymbol{\tau}_i|_{Q_i}^2), \quad (3.9)$$

where $Q_i \in \mathbb{R}^{d \times d}$, $Q_i = Q_i^\top$, $Q_i > 0$ (i.e., Q_i is symmetric and positive-definite), and $\boldsymbol{\kappa}_i \triangleq \text{col}(\boldsymbol{\tau}_i, \text{nonredundant elements of } Q_i)$; each $\boldsymbol{\tau}_i$ is called *center* or *centroid*. Typical examples of basis functions derived from the radial construction are given by Gaussian basis functions, i.e.,

$$\gamma_j(\mathbf{x}, \mathbf{w}_n) = \sum_{i=1}^n c_{ij} e^{-|\mathbf{x} - \boldsymbol{\tau}_i|_{Q_i}^2} + b_j. \quad (3.10)$$

The above-described examples of ridge and radial OHL networks are both \mathcal{C} - and \mathcal{L}_p -approximating FSP functions. The \mathcal{C} -density property has been proved in [26, 64, 71, 106] and elsewhere for OHL neural networks (see (3.7)), and in [41, 124] for radial basis function networks (see (3.10)). In the parlance of neurocomputing, the \mathcal{C} -approximating networks are termed (in rather a high-sounding way) as “universal approximators” (see Sect. 3.6). It is also worth noting that the choice of Examples (3.7), (3.8), and (3.10) is driven by the fact that, among the numberless OHL networks, the chosen ones (and several others) benefit from powerful capabilities useful in finding approximate solutions for Problem P. This fundamental aspect will be addressed later in this chapter.

For OHL networks obtained by ridge or radial constructions, where in the latter case we let $Q_i = \sigma^2 I$, (3.5) becomes

$$\mathcal{N}(n) = n(d + 1 + m) + m. \quad (3.11)$$

If one compares the three above-described ways of constructing OHL networks, one notices that the basis functions constructed by tensor-products contain d factor functions that must be evaluated separately. Then, roughly speaking, the computational cost to evaluate the related OHL networks is proportional to the dimension d . For the ridge and radial construction, the dimension-dependent computational cost consists only in the evaluation of an inner product and a norm of a difference between two vectors, respectively. OHL networks based on tensor-product construction are seldom used in high-dimensional settings.

Note that radial and ridge OHL networks exhibit opposite geometrical properties. Radial basis functions compute a distance to a center; such a distance becomes the argument of a radial function, which typically tends to 0 as that distance tends to ∞ . Hence, they respond to “localized” regions. On the contrary, the arguments of the ridge basis functions are given by a weighted sum of inputs plus a bias and so they respond to “nonlocalized” regions of the input space.

3.3 Multi-Hidden-Layer Networks

An MHL network implementing the function $\gamma(\mathbf{x}, \mathbf{w}_n)$ is composed of L layers. We denote each layer by the integer $s = 1, \dots, L$ (see Fig. 3.2).

The generic layer s contains n_s units. The input/output mapping of the q th unit of the s th layer is given by (see Fig. 3.3)

$$y_q(s) = h[z_q(s)], \quad s = 1, \dots, L, \quad q = 1, \dots, n_s, \quad (3.12)$$

$$z_q(s) = \sum_{p=1}^{n_{s-1}} w_{pq}(s)y_p(s-1) + w_{0q}(s), \quad (3.13)$$

where $y_q(s)$ is the output variable of the unit, $h : \mathbb{R} \rightarrow \mathbb{R}$, and $w_{pq}(s)$ and $w_{0q}(s)$ are the weight and bias parameters, respectively.

Of course, (3.12) and (3.13) include OHL networks (3.4) with output components given by (3.3) (see also Fig. 3.1). More specifically, Networks (3.4) can be obtained from (3.12) and (3.13) by letting $L = 2$ and $h[z_q(2)] = z_q(2)$, $q = 1, \dots, n_2$, in the second layer. By implementing the function $\gamma(\mathbf{x}, \mathbf{w}_n)$ through an MHL network, for every fixed values of \mathbf{x} and \mathbf{w}_n , we have

$$\gamma(\mathbf{x}, \mathbf{w}_n) = \mathbf{y}(L) \triangleq \text{col} [y_1(L), \dots, y_m(L)], \quad (3.14)$$

$$\mathbf{x} = \mathbf{y}(0) \triangleq \text{col} [y_1(0), \dots, y_d(0)]. \quad (3.15)$$

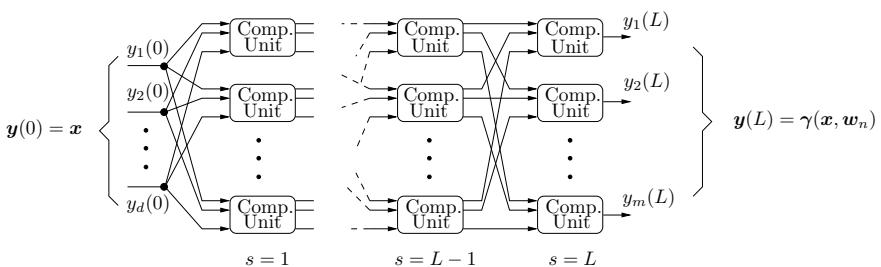


Fig. 3.2 Scheme of an MHL network

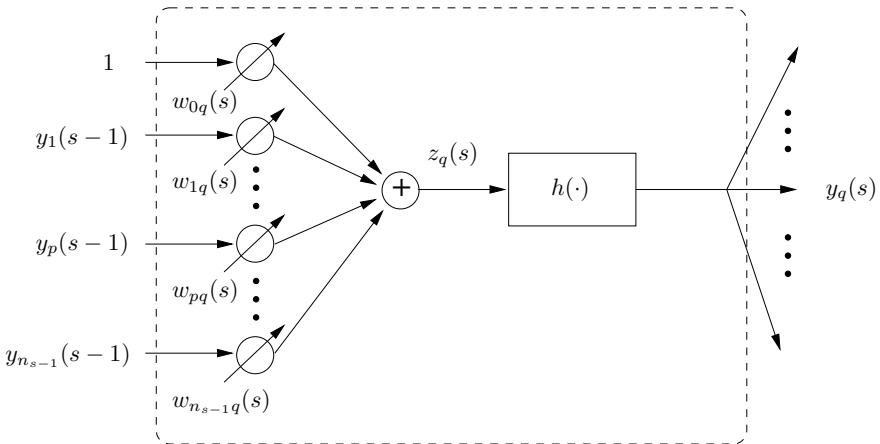


Fig. 3.3 A computational unit showing variable and weight notation

Clearly, many generalizations of the MHL networks described above are possible, for example, the activation functions might be different in each layer (although using the same unit is a common choice) and the architecture may be changed to allow different links between the various units.

3.4 Terminology

We now give some clarifications on the terminology by discussing the connections between “ridge OHL networks,” “multilayer networks,” and the neural network parlance. The reader is referred to [63, 134], and the references therein for a more general treatment.

The model known as the “multilayer feedforward perceptron” in the neural network community consists of a finite sequence of layers, each of them containing a finite number of ridge computational units. In each layer, any unit is connected to each unit of the subsequent layer. The computational units h are also called *activation functions* or *neurons*, and the connections are known as *synapses*. The components of the parameters vector α_i (see (3.6)) are called *weights*; the parameters β_i are called *thresholds* or *biases*. The term “feedforward” is motivated by the fact that in such a model the information flows from each layer to the subsequent one. The first layer, called the *input layer*, consists of the inputs to the network and the last layer, providing the output signals, is called the *output layer*. In between there are the so-called *hidden layers*, whose computational units are the *hidden neurons*.

Multilayer feedforward perceptrons are usually classified on the basis of the number of hidden layers. Unfortunately, sometimes the neural network terminology is not consistent. For example, the term “multilayer perceptron” was introduced by

Rosenblatt [131] to indicate the “no-hidden-layer model” with the Heaviside activation function, inspired by analogy with biological models. However, sometimes such a model is confusingly called a “single layer feedforward network.”

From a mathematical perspective, applying an activation function at the output layer, especially if such an activation function is bounded, might be unnecessarily restrictive. Indeed, a widespread choice in neural network literature and applications is not to use an activation function at the output layer but to merely perform a weighted sum of the outputs of the upstream hidden layer. This choice leads naturally to those that we call “OHL networks.” Thus, when the ridge construction is used, such networks correspond to perceptrons with one hidden layer, with which the reader coming from a neural network background is familiar.

Moreover, OHL networks based on the ridge construction are sometimes called “one-hidden-layer feedforward perceptrons with linear output units,” as the output is obtained by a linear combination of the outputs of the hidden layer (i.e., the activation functions of the output layer are linear). Some authors call OHL networks based on the ridge construction “three-layer feedforward networks with linear output units” (e.g., [135]) or “three-layer perceptrons” (e.g., [72]), as they also account for the input and output layers. Sometimes, the term “two-layer feedforward networks with linear output units” can be found too, accounting for the hidden layer and the output one. For other variations of the terminology, see [146, footnote 6, p. 1421].

3.5 Kernel Smoothing Models

Methods based on the use of kernel functions have been used extensively in the literature, typically in function estimation and classification problems (see, for example, [62]). Probably, the most popular application of kernels in function estimation in recent years is the method of support vector machines (SVMs) (see, for instance, [136, 143]), recently extended to support constraint machines [50] (see also the related works [49, 51, 52]). However, the specific form of their training procedure does not make the SVM paradigm suitable for the purposes of this book, since the number and locations of kernel centers cannot be chosen in a flexible way, and the widths of the kernels are the same for all the kernel units. Thus, we consider here another popular estimator that still employs kernel functions but in a more straightforward way.

The central idea consists in choosing a set X_K of K vectors $\mathbf{x}^{(k)}$, $k = 1, \dots, K$, belonging to the domain $X \subset \mathbb{R}^d$ of the function $\gamma: X \rightarrow \mathbb{R}$ (for the moment, we assume γ to be a one-dimensional function). Let

$$X_K \triangleq \{\mathbf{x}^{(k)} \in X : k = 1, \dots, K\}.$$

For each $\mathbf{x}^{(k)}$, let the values $\gamma(\mathbf{x}^{(k)})$ be known. Thus, we define

$$\Sigma_K \triangleq \{(\mathbf{x}^{(k)}, \gamma(\mathbf{x}^{(k)})) : k = 1, \dots, K\}. \quad (3.16)$$

Now, for any $\mathbf{x} \in X$, we define an approximating FSP function $\tilde{\gamma}(\mathbf{x}, \alpha, \Sigma_K)$ parametrized by a positive scalar α and dependent on the set Σ_K . Then, the function $\tilde{\gamma}$ is obtained by taking the average of the K values $\gamma(\mathbf{x}^{(k)})$, $\mathbf{x}^{(k)} \in X_K$, weighted by a measure of the distance of \mathbf{x} from each point $\mathbf{x}^{(k)}$. Therefore, we define the following FSP function, which is often called *Nadaraya–Watson kernel-weighted average* in the literature:

$$\tilde{\gamma}(\mathbf{x}, \alpha, \Sigma_K) \triangleq \frac{\sum_{k=1}^K \gamma(\mathbf{x}^{(k)}) \mathcal{K}_\alpha(\mathbf{x}, \mathbf{x}^{(k)})}{\sum_{k=1}^K \mathcal{K}_\alpha(\mathbf{x}, \mathbf{x}^{(k)})}, \quad (3.17)$$

where the measure of the distance from each point $\mathbf{x}^{(k)}$ is given by a nonnegative function $\mathcal{K}_\alpha(\mathbf{x}, \mathbf{x}^{(k)})$, called a *kernel function*, decreasing with the distance between \mathbf{x} and $\mathbf{x}^{(k)}$. For a given set Σ_K , Function (3.17) has a fixed structure containing the “free” parameter α .

A typical family of kernel functions takes on the form

$$\mathcal{K}_\alpha(\mathbf{x}, \mathbf{x}^{(k)}) \triangleq \psi\left(\frac{|\mathbf{x} - \mathbf{x}^{(k)}|}{\alpha}\right),$$

where $|\cdot|$ is the Euclidean norm and $\psi(t)$ is a nonincreasing function for $t > 0$ and has its maximum at $t = 0$. Examples of functions ψ are the following:

- The *Gaussian function*

$$\psi_{Gauss}(t) \triangleq e^{-t^2/2}.$$

- The *Epanechnikov quadratic kernel*

$$\psi_{Ep}(t) \triangleq \begin{cases} \frac{3}{4}(1-t^2), & \text{if } |t| \leq 1 \\ 0, & \text{otherwise} \end{cases}.$$

- The *tri-cube function*

$$\psi_{tc}(t) \triangleq \begin{cases} (1-|t|^3)^3, & \text{if } |t| \leq 1 \\ 0, & \text{otherwise} \end{cases}.$$

Figure 3.4 compares these functions after each of them has been multiplied by a suitable constant, in such a way that its integral is equal to 1. Note that both the tri-cube and the Epanechnikov kernels have a compact support whose radius is given by α . The derivative of the former is continuous at the boundary of its support, while the derivative of the latter is not. The Gaussian kernel is continuously differentiable but has unbounded support; α is the standard deviation. Then, in all cases, α is a parameter that controls the “width” of the kernel.

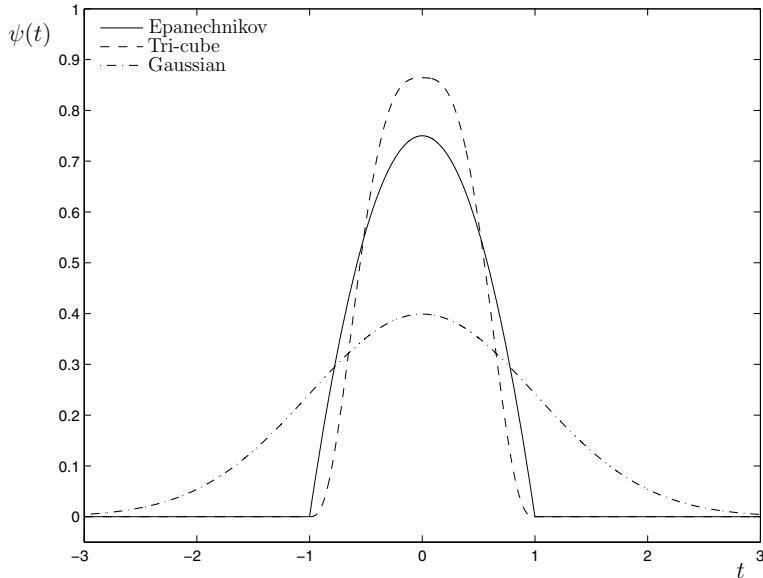


Fig. 3.4 Comparison of three popular kernels for local smoothing. ©2009 Adapted by permission from Springer Nature: [62]

The main motivation for the normalizing denominator is the following: when the kernel functions take on small values or vanish, the numerator in (3.17) may leave “holes” in the domain of \mathbf{x} thus showing poor approximation capabilities. Such a drawback can be avoided by introducing the normalization

$$\tilde{\mathcal{K}}_\alpha(\mathbf{x}, \mathbf{x}^{(k)}) = \frac{\mathcal{K}_\alpha(\mathbf{x}, \mathbf{x}^{(k)})}{\sum_{h=1}^K \mathcal{K}_\alpha(\mathbf{x}, \mathbf{x}^{(h)})}, \quad k = 1, \dots, K.$$

This drawback is explained in Figs. 3.5 and 3.6 for Gaussian kernels and suitable values of α , $\mathbf{x}^{(k)}$, and $\gamma(\mathbf{x}^{(k)})$, $k = 1, \dots, K$. Of course, in the case of kernels with compact support, the radius α must be large enough so that the intersection of the supports covers the whole domain X , in order that the denominator in (3.17) never vanishes at any point.

The extension of the scalar case to vector-valued functions $\gamma: X \rightarrow \mathbb{R}^m$ is immediate. We simply consider m FSP functions of the form (3.17), each parametrized by a width α_j , $j = 1, \dots, m$, and we give them the role of components of the vector-valued function (compare with (3.4))

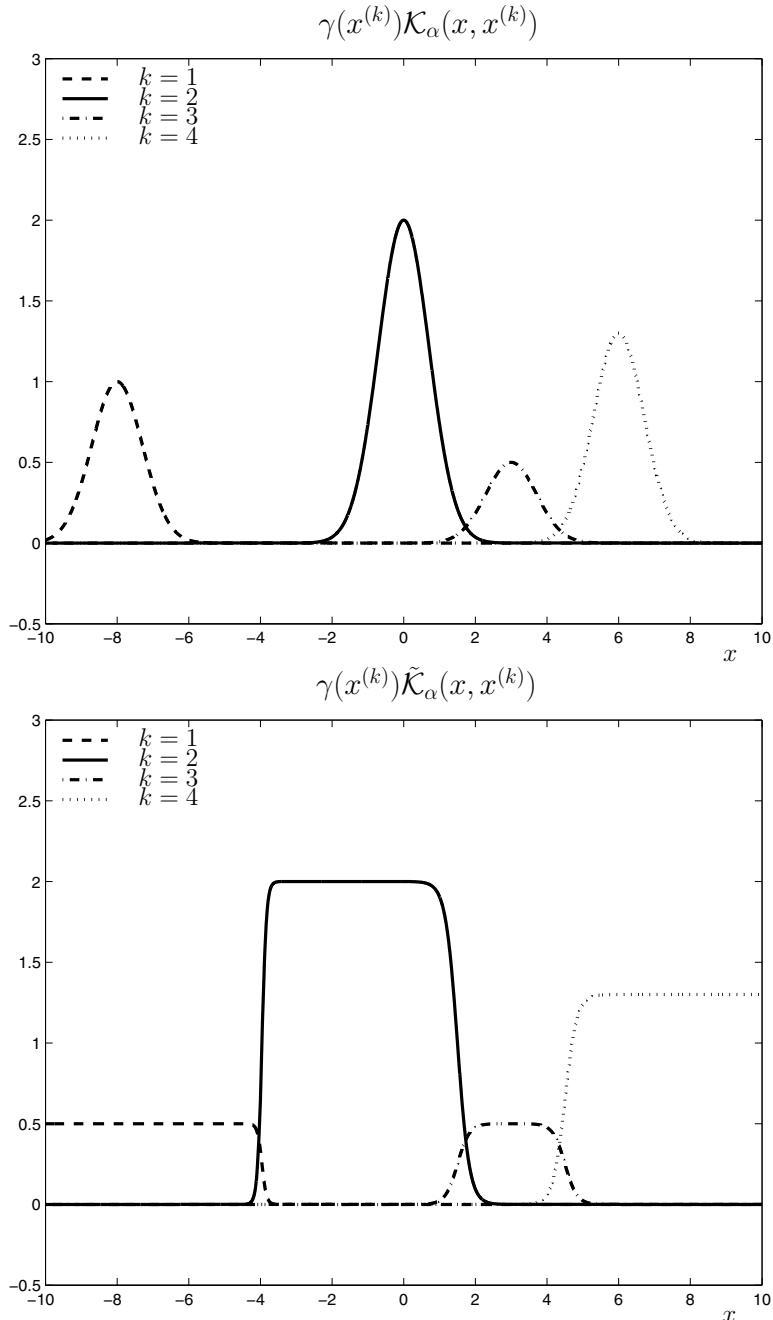


Fig. 3.5 (Top) Examples of $K = 4$ Gaussian kernels \mathcal{K}_α and (bottom) their normalized versions $\tilde{\mathcal{K}}_\alpha$. The Gaussians have standard deviations $\alpha = \sigma = 1$ and centers $x^{(1)} = -8, x^{(2)} = 0, x^{(3)} = 3, x^{(4)} = 6$

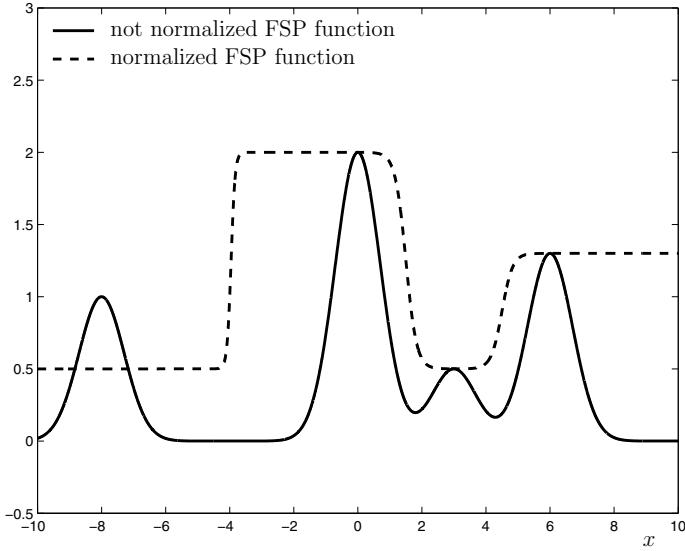


Fig. 3.6 Function (3.17) and its unnormalized version $\sum_{k=1}^K \gamma(\mathbf{x}^{(k)}) \mathcal{K}_\alpha(\mathbf{x}, \mathbf{x}^{(k)})$

$$\tilde{\gamma}(\mathbf{x}, \boldsymbol{\alpha}, \Sigma_K) = \text{col} \left[\frac{\sum_{k=1}^K \gamma_j(\mathbf{x}^{(k)}) \mathcal{K}_{\alpha_j}(\mathbf{x}, \mathbf{x}^{(k)})}{\sum_{k=1}^K \mathcal{K}_{\alpha_j}(\mathbf{x}, \mathbf{x}^{(k)})}, j = 1, \dots, m \right], \quad (3.18)$$

where $\boldsymbol{\alpha} \triangleq \text{col}(\alpha_1, \dots, \alpha_m)$.

FSP functions of the form (3.18) are widely used to approximate functions γ that are known only at points $\mathbf{x}^{(k)} \in X_K$. For example, this makes such structures good candidates for direct use in approximate dynamic programming algorithms, as will be detailed in Chap. 6.

An inspection of (3.17) shows that, in general, this kind of FSP functions is not necessarily endowed with the nestedness property. Yet, the property may be satisfied if the kernels have compact supports and the centers $\mathbf{x}^{(k)}$ can be freely chosen. However, since their optimization would generally require an unfeasible computational burden, the most convenient approach is to generate the centers in such a way that their distribution satisfies suitable hypotheses. For instance, in [19] it is shown that, under appropriate conditions on the kernel functions, if the centers in the set Σ_K uniformly cover the input space, then the absolute error between the model outputs and continuous functions to be approximated can be made arbitrarily small by increasing K and optimizing the width parameter α . In this case the number

K of points in the set Σ_K is conceptually related to the model complexity, as is the number that needs to grow to improve the approximation capability.

Finally, we recall that this simple paradigm of employing kernel functions can be extended to more complex (and more performing) structures, like local linear regression models, in which we model a linear local dependence in a neighborhood of the input point, instead of a piecewise constant effect (see, for instance, [20] for a discussion of the advantages and for a comparison with the standard kernel smoothing models).

3.6 Density Properties

The first theoretical question arising on a given type of OHL network is whether a sufficiently high number of basis functions allows one to approximate all “reasonable” functions encountered in applications up to every desired degree of accuracy. This issue has been addressed for the first time in this book in Sect. 2.6, leading to the definition of density properties of families of FSP functions in a set $\mathcal{M} \subseteq \mathcal{G}$, for a normed space \mathcal{G} , and to the definition of \mathcal{M} -approximating sequences of sets and \mathcal{M} -approximating FSP functions (see Definitions 2.5 and 2.6). When the normed space \mathcal{G} is the space of continuous functions with the supremum norm, in neural network terminology the density property is also called the *universal approximation property* (see [71]). Although density does not imply efficiency of an approximation scheme (e.g., a small number of parameters required for a desired accuracy in the approximation), its lack with respect to function spaces commonly used in applications is an indication of limited capabilities.

3.6.1 \mathcal{C} - and \mathcal{L}_p -Density Properties

The choice of the space \mathcal{G} and, as a consequence, the choice of the type of basis functions yielding to the construction of OHL networks with the corresponding density property, depends on the instance of Problem P^d at hand. As stated in Sect. 2.6, we are interested in IDO problems for which a basic role is played by the \mathcal{C} - and \mathcal{L}_2 -density properties on a compact subset D of \mathbb{R}^d .

Now, one may wonder which kind of relationship, if any, between \mathcal{C} - and \mathcal{L}_p -density properties does exist.⁵ In this respect, let us first recall from standard results in functional analysis (see, e.g., [1, pp. 28–31] and [127, p. 151]) that, for every compact set $D \subset \mathbb{R}^d$ and every $1 \leq p < \infty$, $\mathcal{C}(D)$ is dense in $\mathcal{L}_p(D)$. Thus, by the following two-step density argument one concludes that density of an OHL network in $\mathcal{C}(D)$ implies its density in $\mathcal{L}_p(D)$.

⁵We resort to the simpler notation introduced at the end of Sect. 2.6.

Let $\{\mathcal{A}_n\}_{n=1}^\infty$ be a sequence of sets of OHL networks with a given type of basis functions and suppose that the sequence $\{\mathcal{A}_n\}_{n=1}^\infty$ is a $\mathcal{C}(D)$ -approximating sequence of sets, that is, it is dense in $\mathcal{C}(D)$. Fix $p \in [1, \infty)$ and $D \subset \mathbb{R}^d$ compact. Since $\mathcal{C}(D)$ is dense in $\mathcal{L}_p(D)$, as remarked above, in correspondence with any function $\gamma \in \mathcal{L}_p(D)$ and any constant $\varepsilon > 0$ there exists a function $\zeta \in \mathcal{C}(D)$ such that $\|\gamma - \zeta\|_{\mathcal{L}_p(D)} \leq \varepsilon/2$. For such a ζ , by the assumed density of $\{\mathcal{A}_n\}_{n=1}^\infty$ in $\mathcal{C}(D)$ there exist $n \in \mathbb{Z}^+$ and an OHL network $\gamma_n \in \mathcal{A}_n$ with n basis functions, such that $\|\zeta - \gamma_n\|_{\mathcal{C}(D)} \leq \varepsilon/(2\mu(D)^{1/p})$, where $\mu(D)$ denotes the Lebesgue measure of the set D , and so $\|\zeta - \gamma_n\|_{\mathcal{L}_p(D)} \leq \|\zeta - \gamma_n\|_{\mathcal{C}(D)} \mu(D)^{1/p} \leq \varepsilon/2$. Thus, $\|\gamma - \gamma_n\|_{\mathcal{L}_p(D)} \leq \|\gamma - \zeta\|_{\mathcal{L}_p(D)} + \|\zeta - \gamma_n\|_{\mathcal{L}_p(D)} \leq \varepsilon/2 + \varepsilon/2 = \varepsilon$. Hence, $\{\mathcal{A}_n\}_{n=1}^\infty$ is dense in $\mathcal{L}_p(D)$.

According to the discussion above, it remains understood that conditions guaranteeing the \mathcal{C} -density property also guarantee the \mathcal{L}_p -one. However, without resorting to the two-step density argument outlined above, proof techniques developed ad hoc might guarantee the \mathcal{L}_p -density property (but possibly not the \mathcal{C} -density property) under weaker assumptions on the basis functions. At this point it is worth recalling that an equivalent way to state density of a set \mathcal{Y} in a normed linear space \mathcal{G} is to write that $\text{cl } \mathcal{Y} = \mathcal{G}$, where “cl” denotes closure with respect to the norm $\|\cdot\|_{\mathcal{G}}$. That is to say, \mathcal{Y} is dense in \mathcal{G} if and only if the closure of \mathcal{Y} is the whole space \mathcal{G} . Hence, a set \mathcal{Y} of functions has the \mathcal{C} - or \mathcal{L}_p -density property if and only if $\text{cl } \mathcal{Y} = \mathcal{C}(D)$ and $\text{cl } \mathcal{Y} = \mathcal{L}_p(D)$, respectively, where in the first case the closure is with respect to the supremum norm, whereas in the second it is with respect to the \mathcal{L}_p -norm. The closure in $\mathcal{C}(D)$, i.e., with respect to the supremum norm, is also called *uniform closure* [133, p. 149].

3.6.2 The Case of Ridge OHL Networks

Focusing on the case of ridge OHL networks, let us go back in the history of function approximation in the neural network context (see [134] for a survey). In 1969, it was proved that the simple perceptron with no hidden layer can represent or approximate only functions belonging to quite a narrow class (see the book [121] by Minsky and Papert). However, this left open the possibility that network architectures containing one or more hidden layers could achieve better performances. Only some 20 years later, the first results in this direction appeared: at the end of the 1980s, almost 30 years after the publication of the two early rules for training adaptive elements in network architectures in 1960 (the perceptron learning rule, by Rosenblatt [132], and the least mean square algorithm, by Widrow and Hoff [145]), there was a sort of renaissance in neural network theory (in addition to [63], a nice review of the developments in feedforward neural networks during the period 1960–1990 is given in [146]). Many researchers started to investigate the density property and the above-discussed limits of the no-hidden-layer perceptron being known, investigations focused on networks with at least one hidden layer. As we have seen, the model with one hidden layer corresponds to what we have called OHL networks. In the remainder of this section,

we shall show that, under mild conditions on the basis functions, such networks are able to approximate “large” classes of functions arbitrarily well, more precisely, every continuous or \mathcal{L}_p function on every compact subset of \mathbb{R}^d .

From the last decades of the twentieth century, plenty of works appeared, proving that ridge OHL networks are capable of approximating arbitrarily well wide classes of functions commonly used in applications, such as continuous and square-integrable ones. In our context, we are interested in proving that OHL networks become approximating FSP functions, in the sense specified by Definition 2.6 and Remark 2.4, on a variety of normed spaces \mathcal{G} , such as $\mathcal{C}(D, \mathbb{R})$ and $\mathcal{L}_2(D, \mathbb{R})$. These works answered the following question raised in [71]:

The apparent ability of sufficiently elaborate feedforward networks to approximate quite well nearly any function encountered in applications leads one to wonder about the ultimate capabilities of such networks. Are the successes observed to date reflective of some deep and fundamental approximation capability, or are they merely flukes, resulting from selective reporting and a fortuitous choice of problems?

Recall from Sect. 3.2.3 that the ridge construction (see (3.6)) “shrinks” the d -dimensional vector \mathbf{x} into a one-dimensional variable by the inner product, i.e.,

$$\varphi(\mathbf{x}, \boldsymbol{\kappa}_i) = h(\mathbf{x}^\top \boldsymbol{\alpha}_i + \beta_i),$$

where $\boldsymbol{\kappa}_i \triangleq \text{col}(\boldsymbol{\alpha}_i, \beta_i) \in \mathbb{R}^{d+1}$ and $h : \mathbb{R} \rightarrow \mathbb{R}$ is fixed. Each function with such a form is constant along the parallel hyperplanes $\mathbf{x}^\top \boldsymbol{\alpha}_i + \beta_i = k_i$, $k_i \in \mathbb{R}$. Now, let

$$\begin{aligned} \text{Ridge}(h) &\triangleq \text{span} \{h(\mathbf{x}^\top \boldsymbol{\alpha}_i + \beta_i), \boldsymbol{\alpha}_i \in \mathbb{R}^d, \beta_i \in \mathbb{R}\} \\ &= \{\gamma : \mathbb{R}^d \rightarrow \mathbb{R}; \gamma(\mathbf{x}) = \sum_{i=1}^n c_i h(\mathbf{x}^\top \boldsymbol{\alpha}_i + \beta_i), \boldsymbol{\alpha}_i \in \mathbb{R}^d, c_i, \beta_i \in \mathbb{R}\} \end{aligned}$$

denote the set of d -variable single-output OHL networks functions defined on \mathbb{R}^d and based on the ridge construction with a basis function h .

An intuitive way of proving the $\mathcal{C}(D)$ - and $\mathcal{L}_2(D)$ -density of $\text{Ridge}(h)$ for a compact set $D \subset \mathbb{R}^d$ and suitable basis functions h is the following, qualitatively described in [14, p. 511]. First, one can think to approximate any given function in $\mathcal{C}(D)$ or $\mathcal{L}_2(D)$ by a classical multivariable finite trigonometric sum, with an arbitrarily small error (in the supremum or $\mathcal{L}_2(D)$ norm, respectively). For illustration purposes, let us focus on the case $d = 2$ with $D = [0, 1]^2$ being the unit square. In such a case the terms of the sum are of the form $a_{mn} \cos(mx_1) \cos(nx_2)$, $b_{mn} \sin(mx_1) \sin(nx_2)$, $c_{mn} \sin(mx_1) \cos(nx_2)$, $d_{mn} \cos(mx_1) \sin(nx_2)$, where $m, n \in \mathbb{Z}^+$, and a_{mn} , b_{mn} , c_{mn} , and d_{mn} are suitable coefficients. By substituting the classical Werner’s trigonometrical formulas in the sum, one obtains a linear combination of terms of the form $\cos(z_i)$ and $\sin(z_i)$, where each z_i is a linear function of x_1 and x_2 . Now, it is easy to show (see [14] for a proof) that every function f of a single real variable on a closed and bounded interval can be approximated arbitrarily well by a ridge OHL network in the supremum norm if f is continuous and in the \mathcal{L}_2 norm

if it is square-integrable. Hence, approximating the functions $\cos(z_i)$ and $\sin(z_i)$ in such a way, the original trigonometric sum is also approximated by a ridge OHL network.

As regards the three works [26, 40, 71] in which the \mathcal{C} -density property on compact sets was first proved independently and almost simultaneously for wide classes of sigmoidal functions, it should be noted that three different proof techniques were used. Still another technique was used in [18], where the first proof of the \mathcal{L}_2 -density property of ridge OHL networks on compact sets was given.

The next theorem provides a slightly simplified form of the \mathcal{C} - and \mathcal{L}_p -density results obtained in [106], which guarantee density under quite mild hypotheses on the basis functions.

Theorem 3.1 *Let $h : \mathbb{R} \rightarrow \mathbb{R}$ be locally bounded and piecewise continuous and d be a positive integer. Then, for every compact set $D \subset \mathbb{R}^d$, $\text{Ridge}(h)$ is dense in $\mathcal{C}(D)$ and in $\mathcal{L}_p(D)$, $p \in [1, \infty)$, if and only if h is not an algebraic polynomial. \square*

The necessity of non-polynomiality for density in $\mathcal{C}(D)$ can be easily seen. Indeed, if h is a polynomial of degree k , then, for every α and β , $h(\mathbf{x}^\top \boldsymbol{\alpha} + \beta)$ is a polynomial of degree at most k . Hence, $\text{Ridge}(h)$ is a subset of the set of algebraic polynomials of degree at most k , and thus it cannot be dense in $\mathcal{C}(D)$. Similarly, for the density in $\mathcal{L}_p(D)$, $p \in [1, \infty)$, the necessity of non-polynomiality immediately follows by observing that, if h is a polynomial of degree k , then $\text{Ridge}(h)$ is a subset of the set of polynomials of degree at most k ; hence, it cannot be dense in $\mathcal{L}_p(D)$. As regards the sufficiency of non-polynomiality, from Theorem 3.1, $\text{Ridge}(h)$ is dense in $\mathcal{C}(D)$ and, as already recalled, $\mathcal{C}(D)$ is dense in $\mathcal{L}_p(D)$, for $p \in [1, \infty)$, thus immediately ending the proof for the case of $\mathcal{L}_p(D)$, once sufficiency of non-polynomiality has been proved for $\mathcal{C}(D)$.

Theorem 3.1 deserves some further comments. According to it, the \mathcal{C} - and \mathcal{L}_p -density properties are not restricted to “biologically motivated” sigmoids but, with the exception of polynomials, they are satisfied by any “reasonable” basis function. As a second comment, it should be noted that the basis functions in ridge OHL networks need not be continuous or smooth to guarantee \mathcal{C} - and \mathcal{L}_p -density: the only requirement is non-polynomiality. In such a way, Theorem 3.1 answers the basic question raised in [68, Discussion, p. 253]:

Whether or not the continuity assumption can entirely be dropped is still an open (and quite challenging) problem.

Finally, it is worth investigating the role played by the thresholds, i.e., by the parameters β_i in the basis functions $h(\mathbf{x}^\top \boldsymbol{\alpha}_i + \beta_i)$ (see (3.6)). To this end, following [106], let us consider the continuous, bounded, nonconstant, and non-polynomial function $h(x) = \sin(x)$ and the compact interval $[-1, 1]$. As the family $\{\sin(wx), w \in \mathbb{R}\}$ consists only of odd functions, functions like $\cos(x)$ cannot be

Table 3.1 For $D \subset \mathbb{R}^d$ compact, some density properties in $\mathcal{C}(D)$ and $\mathcal{L}_p(D)$ for ridge OHL networks

Space \mathcal{G}	Basis function h	References
$\mathcal{L}_2(D)$	Continuous sigmoidal	[18]
$\mathcal{C}(D)$	Continuous sigmoidal	[26]
$\mathcal{L}_1(D)$	Bounded sigmoidal	[26]
$\mathcal{L}_1(D)$	$h \in \mathcal{L}_1(\mathbb{R})$, $\int h(t) dt \neq 0$	[26]
$\mathcal{C}(D)$	Nondecreasing sigmoidal	[71]
$\mathcal{L}_p(D)$, $p \in [1, \infty)$	Nondecreasing sigmoidal	[71]
$\mathcal{C}(D)$	$h \in \mathcal{L}_1(\mathbb{R})$, $\int h(t) dt \neq 0$, and h continuous	[141]
$\mathcal{C}(D)$	Continuous, bounded, and nonconstant	[68]
$\mathcal{L}_p(D)$, $p \in [1, \infty)$	Bounded and nonconstant	[68]
$\mathcal{C}(D)$	Increasing sigmoidal	[73]
$\mathcal{C}(D)$	Continuous and h/p bounded for some polynomial p but h not a polynomial itself	[118]
$\mathcal{C}(D)$	k th degree sigmoidal	[118]
$\mathcal{C}(D)$	Locally bounded, piecewise continuous, and not an algebraic polynomial	[106]
$\mathcal{C}(D)$	Locally Riemann integrable and not an algebraic polynomial	[70]
$\mathcal{L}_p(D)$, $p \in [1, \infty)$	Locally bounded and not an algebraic polynomial	[70]
$\mathcal{C}(D)$	Bounded and such that there exist $\lim_{t \rightarrow \pm\infty} h(t)$	[106]
$\mathcal{C}(D)$	Bounded sigmoidal	[21, 22, 76]

approximated by OHL networks using such a family of basis functions. Hence, the set $\text{span}\{\sin(wx), w \in \mathbb{R}\}$ cannot be dense in $\mathcal{C}([-1, 1])$. However, density can be recovered by introducing a threshold element (corresponding, in this case, to a phase shift) in such a set of functions, since this makes it possible to also include cosines in the basis (e.g., $\sin(x + \pi/2) = \cos(x)$). On the other hand, there exist ridge basis functions for which the threshold has no influence. For example, this is the case with the exponential function since, for every $t \in \mathbb{R}$, $e^{x+t} = e^x e^t$, i.e., the effect of the threshold t on the function e^x is simply a multiplication by a positive constant. In [70], conditions on the basis functions are given, such that a single threshold suffices for \mathcal{C} - and \mathcal{L}_p -density (i.e., the density property can also be guaranteed when all the thresholds of the basis functions have the same value).

Table 3.1, which is by no means exhaustive, summarizes a variety of density results for many types of basis functions. An extended version of the table can be found in [134, Table 1].

3.6.3 The Case of Radial OHL Networks

OHL networks with suitable radial basis functions are also universal approximators.

Recall from Sect. 3.2.3 that the radial construction “shrinks” the d -dimensional vector \mathbf{x} into a scalar variable by means of some norm $|\cdot|_{Q_i}$ chosen on \mathbb{R}^d (see (3.9)). A particularly interesting example is the case in which $Q_i = b_i I$, where $b_i > 0$ and I is the identity matrix, so one has $\kappa_i = \text{col}(\boldsymbol{\tau}_i, b_i)$, and

$$\varphi(\mathbf{x}, \kappa_i) = h(b_i |\mathbf{x} - \boldsymbol{\tau}_i|^2). \quad (3.19)$$

Now, let

$$\begin{aligned} \text{Radial}(h) &\triangleq \text{span}\{h(b_i |\mathbf{x} - \boldsymbol{\tau}_i|^2), \boldsymbol{\tau}_i \in \mathbb{R}^d, b_i > 0\} \\ &= \{\gamma : \mathbb{R}^d \rightarrow \mathbb{R}; \gamma(\mathbf{x}) = \sum_{i=1}^n c_i h(b_i |\mathbf{x} - \boldsymbol{\tau}_i|^2), \boldsymbol{\tau}_i \in \mathbb{R}^d, b_i > 0, c_i \in \mathbb{R}\} \end{aligned}$$

denote the set of d -variable single-output OHL networks defined on \mathbb{R}^d and based on the radial construction with a basis function h .

The following theorem is a specialization to the set $\text{Radial}(h)$ of a more general density result by Park and Sandberg [124]. It shows that under mild conditions on h , the set $\text{Radial}(h)$ is dense in $\mathcal{L}_p(\mathbb{R}^d)$ for every $p \in [1, \infty)$. This holds even in the case in which one assumes that the parameters b_i are the same in all the terms of the expansion $\sum_{i=1}^n c_i h(b_i |\mathbf{x} - \boldsymbol{\tau}_i|^2)$.

Theorem 3.2 *Let $h : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ be integrable, bounded, continuous almost everywhere, and such that $\int_0^\infty h(x)dx \neq 0$, and let d be any positive integer. Then, $\text{Radial}(h)$ is dense in $\mathcal{L}_p(\mathbb{R}^d)$ for every $p \in [1, \infty)$. \square*

The proof of Theorem 3.2 is based on the fact that, if one defines the functions

$$u(x) \triangleq \frac{h(x)}{\int_{\mathbb{R}^d} h(\alpha)d\alpha},$$

where h satisfies the hypotheses of the theorem, and, for $b > 0$,

$$u_b(x) \triangleq b^d u(bx),$$

then, for each function $f \in \mathcal{L}_p(\mathbb{R}^d)$, $p \in [1, \infty)$, one has $\lim_{b \rightarrow 0^+} \|u_b * f - f\|_{\mathcal{L}_p(\mathbb{R}^d)} = 0$, where $*$ denotes the convolution operator. Hence, if b is sufficiently small, one can show that the convolution integral $u_b * f$ can be approximated with small error by a finite summation, which is a function in $\text{Radial}(h)$. After some technical steps, this proves the density of $\text{Radial}(h)$ in $\mathcal{L}_p(\mathbb{R}^d)$, for $p \in [1, \infty)$.

The paper [124] also provides density results of $\text{Radial}(h)$ in $\mathcal{C}(\mathbb{R}^d)$, endowed with a suitable metric.

3.6.4 Multiple Hidden Layers and Multiple Outputs

As we have seen, one hidden layer is sufficient to achieve \mathcal{C} - and \mathcal{L}_p -density. However, this may not be the same as regards the number of parameters required to achieve a desired accuracy. We shall address this issue in Sect. 3.7.4. Although OHL networks exhibit sufficiently powerful capabilities for our purposes and they have been very widely investigated in the literature, several times the use of multilayer networks may be useful from a numerical point of view. Hence, in the application examples considered in the second part of the book. In some specific cases, multilayer networks will be used.

Let us briefly consider the case in which the mapping to be approximated has $m > 1$ components. Although the extension from the single-output case is formally straightforward, as it simply requires putting together m OHL networks, one for each component of the output, this way of proceeding takes into account only the point of view of density, i.e., arbitrarily accurate approximation. As, in proceeding like this, each component of the output is treated independently, one gives up considering how the same basis functions can contribute in approximating different components, how the minimum number of parameters that guarantees a desired approximation accuracy can be diminished by taking into account the possible mutual dependence of the various components of the output, and how a practical learning algorithm for finding the optimal values of the parameters may take advantage of this.⁶ For an alternative approach to approximation of vector-valued functions, see, e.g., [55, 122, 123].

3.7 From Universality of Approximation to Moderate Model Complexity

In Sect. 3.6.2, we have seen that quite mild assumptions on the basis functions allow one to prove that ridge OHL networks enjoy the density property in $\mathcal{C}(D)$ and in $\mathcal{L}_p(D)$. Similar density results for OHL networks with suitable radial basis functions have been presented in Sect. 3.6.3. It is worth noting that some density proofs are merely existential: they neither provide an algorithm to construct the network, nor give an estimate of the minimum number of basis functions that guarantees a desired approximation accuracy. Other proofs are constructive; among these, some provide upper bounds on such a minimum number of basis functions. Typically, these upper bounds grow very fast (e.g., exponentially) with the number d of variables of the functions to be approximated. This is clearly a significant drawback when addressing sequences of approximation Problems PA_n^d for increasing values of the dimension d and of the model complexity n , as such bounds are not able to show that the

⁶Note that such a mutual dependence may arise only when the set of vector-valued functions to be approximated is not the Cartesian product of sets of scalar-valued functions.

curse of dimensionality does not arise (see Sect. 2.7, where the meaning of curse of dimensionality in the context of approximating sequences of sets has been defined).

In [69], it is conjectured that the fast growth with d in such density proofs is mainly due to the fact that typical constructive arguments are “artificial”, in the sense that the OHL network approximating a function γ is not obtained directly from γ itself but going through another family (or several other intermediate families) of approximators. To clarify this point, let us consider, as an example, the technique used in [71] to prove density of $\text{Ridge}(h)$ in $\mathcal{C}(D)$, for a nondecreasing sigmoidal h . Given a function $\gamma \in \mathcal{C}(D)$, first this is approximated by $\text{Ridge}(\cos)$ (i.e., by a trigonometric polynomial), then an approximation for the cosine function is searched for in $\text{Ridge}(h)$. Density of $\text{Ridge}(\cos)$ in $\mathcal{C}(D)$ is established in [71] by the Stone–Weierstrass theorem, hence in a “nonconstructive” way (although the same result can be obtained by explicitly constructing a trigonometric polynomial). The use of auxiliary approximators, which allow one to make an intermediate step (or more intermediate steps) between the function to be approximated and the OHL network, is common to almost all constructive proofs of \mathcal{C} - and \mathcal{L}_p -density for OHL networks with sigmoidal computational units.

However, the substantial reason at the basis of the curse of dimensionality arising in density proofs is that they aim at guaranteeing the property in the whole spaces. We anticipate that the recipe to cope with the curse of dimensionality involves giving up on investigating approximation error bounds for the whole space $\mathcal{C}(D)$ or $\mathcal{L}_p(D)$. If the functions to be approximated are restricted to suitable subsets of such spaces, then one can obtain approximating OHL networks with a model complexity that grows “moderately” with d for a fixed approximation accuracy.

To evaluate and compare different approximating sequences $\{\mathcal{A}_n\}_{n=1}^\infty$, we shall estimate the worst-case error of approximation of functions in a set \mathcal{M} by functions in \mathcal{A}_n , represented, for general approximation schemes, by the deviation $\delta(\mathcal{M}, \mathcal{A}_n)$, introduced in Definition 2.7 and, for the particular case of LCFBFs, by the n -width $d_n(\mathcal{M})$, introduced in the following Definition 3.3.

To describe a theoretical lower bound on the approximation error obtained by LCFBFs, Kolmogorov [87] considered the infimum of the deviations of a set \mathcal{M} belonging to a normed linear space \mathcal{G} from all its n -dimensional subspaces \mathcal{G}_n (according to Definition 2.7, we denote such deviations by $\delta(\mathcal{M}, \mathcal{G}_n)$). He introduced the following concept of n -width, which was later called *Kolmogorov n-width*, of a set \mathcal{M} in the space \mathcal{G} (see also [125]).

Definition 3.3 Let \mathcal{M} be a subset of a normed linear space \mathcal{G} . The Kolmogorov n -width of \mathcal{M} in \mathcal{G} is defined as

$$d_n(\mathcal{M}, \mathcal{G}) \triangleq \inf_{\mathcal{G}_n} \delta(\mathcal{M}, \mathcal{G}_n) = \inf_{\mathcal{G}_n} \sup_{\gamma \in \mathcal{M}} \inf_{\gamma_n \in \mathcal{G}_n} \|\gamma - \gamma_n\|_{\mathcal{G}}, \quad (3.20)$$

where the leftmost infimum is taken over all n -dimensional subspaces \mathcal{G}_n of \mathcal{G} . \triangleleft

Note that one should specify the space \mathcal{G} in the definition, since the same set \mathcal{M} might be considered as a subset of various normed linear spaces. However, when

there is no risk of ambiguity we shall write merely $d_n(\mathcal{M})$. Note also that the n -width involves one more infimum than the definition of deviation. This is well-suited to the aim of the comparison of approximation rates between the *best approximation by LCFBFs* and the *OHL approximating sequence of sets* associated with a given choice of the structure of the parametrized basis functions in (3.3).

To sum up, for the case of OHL networks, the comparison between the approximation rates of two different approximating sequences $\{\mathcal{A}_n\}_{n=1}^{\infty}$ and $\{\mathcal{A}'_n\}_{n=1}^{\infty}$ will be in terms of $\delta(\mathcal{M}, \mathcal{A}_n)$ and $\delta(\mathcal{M}, \mathcal{A}'_n)$. The comparison between OHL networks and LCFBFs will be expressed in terms of $\delta(\mathcal{M}, \mathcal{A}_n)$ and $d_n(\mathcal{M})$, instead.

For both the deviation and the n -width, exact expressions for the behavior of the worst-case error as a function of n and d can be obtained (up to constants) only in particular cases (see, for instance, [125, p. 232], where the n -widths of balls in Sobolev spaces [2, 17] are addressed). However, a comparison among different approximation schemes can be based on known upper bounds and lower bounds on the worst-case error. This approach has been followed, e.g., in the works [5, 6, 46, 53, 54, 58, 67, 81, 82, 96, 97, 100]. Families of functions for which LCFBFs suffer from the curse of dimensionality, whereas some OHL networks do not exhibit this drawback, are described, e.g., in [47, 48, 97].

3.7.1 The Maurey–Jones–Barron Theorem

In this section, we present an upper bound on the worst-case approximation error for OHL networks. The main theoretical tool to derive such upper bound is Theorem 3.3, which is based on results by Maurey (see [128]), Jones [77], and Barron [6]. For this reason, it is often called the *Maurey–Jones–Barron theorem* or *Maurey–Jones–Barron lemma*; in the following, we shall refer to it as the *MJB theorem*.

We first recall some basic concepts from the theory of Hilbert spaces. For the reader who is not familiar with them, we simply point out that, informally, Hilbert spaces represent the natural extension to infinite dimension of the concept of a finite-dimensional real vector space with a scalar product (i.e., a finite-dimensional Euclidean space).

Definition 3.4 A Hilbert space \mathcal{H} is a *complete inner product space*, i.e., an inner product space that is complete⁷ with respect to the norm $\|\cdot\|_{\mathcal{H}}$ induced by the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$. \triangleleft

In real Hilbert spaces, the underlying field is \mathbb{R} . A typical example of a real Hilbert space is the space $\mathcal{L}_2(D, \mathbb{R}^m)$ of functions $\gamma : D \rightarrow \mathbb{R}^m$ on a set $D \subseteq \mathbb{R}^d$, whose inner product is defined for any $\gamma_1, \gamma_2 \in \mathcal{L}_2(D, \mathbb{R}^m)$ as

⁷Recall that completeness means that every *Cauchy sequence* in \mathcal{H} converges to an element of \mathcal{H} . A Cauchy sequence in \mathcal{H} is any sequence $\{\gamma_i \in \mathcal{H}\}_{i=1}^{\infty}$ characterized by the property that, for every $\varepsilon > 0$, there exists an index \bar{i} such that for every $i, j \geq \bar{i}$ one has $\|\gamma_i - \gamma_j\|_{\mathcal{H}} \leq \varepsilon$.

$$\langle \gamma_1, \gamma_2 \rangle_{\mathcal{L}_2(D, \mathbb{R}^m)} \triangleq \int_D \gamma_1(\mathbf{x})^\top \gamma_2(\mathbf{x}) d\mathbf{x}, \quad (3.21)$$

which induces the norm

$$\|\gamma\|_{\mathcal{L}_2(D, \mathbb{R}^m)} \triangleq \sqrt{\int_D \gamma(\mathbf{x})^\top \gamma(\mathbf{x}) d\mathbf{x}}. \quad (3.22)$$

The next result is a slightly reformulated version of [6, Lemma 1]. We report its proof from [91] since, as we shall see, it has a nice geometric interpretation. For the definitions of $\text{conv } \mathcal{G}$ and $\text{conv}_n \mathcal{G}$, we refer the reader to Sect. 3.2.2, where they were introduced.

Theorem 3.3 *Let \mathcal{H} be a Hilbert space, \mathcal{G} its subset, and $s_{\mathcal{G}} \triangleq \sup_{g \in \mathcal{G}} \|g\|_{\mathcal{H}}$. Then, for every $\gamma \in \text{cl conv } \mathcal{G}$ and every positive integer n one has*

$$\left\| \gamma - \text{conv}_n \mathcal{G} \right\|_{\mathcal{H}} \leq \sqrt{\frac{s_{\mathcal{G}}^2 - \|\gamma\|_{\mathcal{H}}^2}{n}}. \quad (3.23)$$

□

Proof. For simplicity of notation, we focus here on the case of a real Hilbert space. As the distance from $\text{conv}_n \mathcal{G}$ is continuous on \mathcal{H} (see, for instance, [137]), it is sufficient to verify the statement for $\gamma \in \text{conv } \mathcal{G}$.

For every sequence $\{\bar{g}_i : i \in \mathbb{Z}^+\} \subseteq \mathcal{G}$, let us consider the sequence $\{\gamma_n\}_{n=1}^\infty$ of barycenters, where γ_n is defined as

$$\gamma_n \triangleq \frac{1}{n} \sum_{i=1}^n \bar{g}_i,$$

initialized by $\gamma_0 \triangleq 0$. Setting $e_n \triangleq \|\gamma - \gamma_n\|_{\mathcal{H}}$, we get

$$\begin{aligned} e_{n+1}^2 &= \left\| \frac{n}{n+1}(\gamma - \gamma_n) + \frac{1}{n+1}(\gamma - \bar{g}_{n+1}) \right\|_{\mathcal{H}}^2 \\ &= \frac{n^2}{(n+1)^2} e_n^2 + \frac{2n}{(n+1)^2} (\gamma - \gamma_n) \cdot (\gamma - \bar{g}_{n+1}) \\ &\quad + \frac{1}{(n+1)^2} \|\gamma - \bar{g}_{n+1}\|_{\mathcal{H}}^2. \end{aligned}$$

Let

$$\gamma = \sum_{j=1}^m a_j g_j$$

be a representation of γ as a convex combination of elements of \mathcal{G} . Then, for every positive integer $n + 1 \leq m$ we can choose $\bar{g}_{n+1} \in \{g_1, \dots, g_m\} \subseteq \mathcal{G}$ such that the last two terms in the above expression of e_{n+1}^2 are bounded from above by

$$\frac{s_{\mathcal{G}}^2 - \|\gamma\|_{\mathcal{H}}^2}{(n+1)^2}.$$

Indeed,

$$\begin{aligned} & \sum_{j=1}^m a_j \left(\frac{2n}{(n+1)^2} (\gamma - \gamma_n) \cdot (\gamma - g_j) + \frac{1}{(n+1)^2} \|\gamma - g_j\|_{\mathcal{H}}^2 \right) \\ &= \frac{2n}{(n+1)^2} (\gamma - \gamma_n) \cdot \left(\gamma - \sum_{j=1}^m a_j g_j \right) \\ & \quad + \frac{1}{(n+1)^2} \left(\|\gamma\|_{\mathcal{H}}^2 - 2\gamma \cdot \left(\sum_{j=1}^m a_j g_j \right) + \sum_{j=1}^m a_j \|g_j\|_{\mathcal{H}}^2 \right) \\ &= \frac{1}{(n+1)^2} \left(\sum_{j=1}^m a_j \|g_j\|_{\mathcal{H}}^2 - \|\gamma\|_{\mathcal{H}}^2 \right) \leq \frac{1}{(n+1)^2} (s_{\mathcal{G}}^2 - \|\gamma\|_{\mathcal{H}}^2). \end{aligned}$$

In particular, for $n = 0$, we conclude from the above inequality about the existence of $\bar{g}_1 \in \{g_1, \dots, g_m\} \subseteq \mathcal{G}$ such that $e_1^2 = \|\gamma - g_1\|_{\mathcal{H}}^2 \leq s_{\mathcal{G}}^2 - \|\gamma\|_{\mathcal{H}}^2$. Thus, setting $c \triangleq s_{\mathcal{G}}^2 - \|\gamma\|_{\mathcal{H}}^2$ we get $e_1^2 \leq c$ and similarly,

$$e_{n+1}^2 \leq \frac{n^2}{(n+1)^2} e_n^2 + \frac{c}{(n+1)^2}. \quad (3.24)$$

It is easy to check by induction that Recursion (3.24) implies $e_n^2 \leq \frac{c}{n}$. Indeed, $e_{n+1}^2 \leq \frac{n^2}{(n+1)^2} \frac{c}{n} + \frac{c}{(n+1)^2} = \frac{c}{n+1}$. ■

Note that the proof of Theorem 3.3 provides as a byproduct an upper bound on the rate of convergence of incremental learning algorithms (i.e., algorithms that add a new hidden unit at each step; see, for instance, [90]).

It is worth noting that the functions \bar{g}_i in the proof of Theorem 3.3 are chosen one at a time, in such a way that, for every positive integer $n + 1 \leq m$, the bound

$$\frac{2n}{(n+1)^2} (\gamma - \gamma_n) \cdot (\gamma - \bar{g}_{n+1}) + \frac{1}{(n+1)^2} \|\gamma - \bar{g}_{n+1}\|_{\mathcal{H}}^2 \leq \frac{s_{\mathcal{G}}^2 - \|\gamma\|_{\mathcal{H}}^2}{(n+1)^2}$$

holds. An alternative *greedy* construction, still providing an upper bound of order $O(1/\sqrt{n})$ on the worst-case approximation error, is considered in [6], where, at each step, functions belonging to the subset \mathcal{G} are used to improve the approximation of the target function γ so as to obtain a fast convergence. More specifically, let us

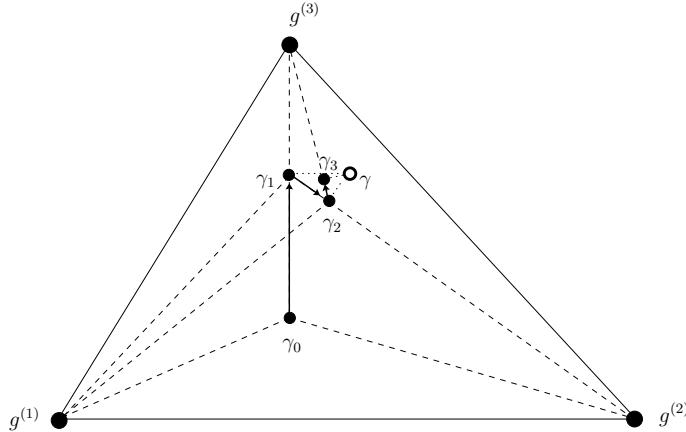


Fig. 3.7 Geometric interpretation of the iterative approximation scheme used in the greedy proof of Theorem 3.3

consider a given target function $\gamma \in \text{cl conv } \mathcal{G}$ to be approximated. Starting from an arbitrary function $\gamma_0 \in \mathcal{G}$, such an alternative construction builds a sequence of approximating functions $\{\gamma_n : n = 1, 2, \dots\}$ with the structure

$$\gamma_n = \bar{\alpha}_n \gamma_{n-1} + (1 - \bar{\alpha}_n) \bar{g}_n, \quad (3.25)$$

where $\bar{\alpha}_n \in [0, 1]$ and $\bar{g}_n \in \mathcal{G}$ are given by the solutions of the problems

$$\inf_{\alpha_n \in [0, 1], g_n \in \mathcal{G}} \|\gamma - \alpha_n \gamma_{n-1} - (1 - \alpha_n) g_n\|_{\mathcal{H}}^2. \quad (3.26)$$

It is worth noting that, to obtain an upper bound of order $O(1/\sqrt{n})$ on the worst-case approximation error, similar to that of Theorem 3.3, Problems (3.26) are not required to be solved exactly: at each iteration, it is sufficient to reach a distance from the infimum bounded from above by $O(1/n^2)$ (see [6, p. 938]).

A geometric interpretation of the iterative procedure described above is shown in Fig. 3.7, where, to clarify ideas, we focus on a very simple case in which $\mathcal{G} = \{g^{(1)}, g^{(2)}, g^{(3)}\}$. The functions $g^{(1)}, g^{(2)}, g^{(3)}$ are enhanced as the three vertices of the triangular shape representing the convex hull $\text{conv } \mathcal{G}$, which in this case coincides with its closure $\text{cl conv } \mathcal{G}$. Three steps of the iterative procedure (3.25) are reported; the initial function $\gamma_0 \in \text{cl conv } \mathcal{G}$ is arbitrarily chosen. Then, for $n = 0$, the geometric evidence suggests that $\bar{g}_1 = g^{(3)}$, thus obtaining γ_1 . Analogously, in the next two iterations, we immediately obtain $\bar{g}_2 = g^{(2)}$ and $\bar{g}_3 = g^{(3)}$. The convergence behavior of the functions γ_i to the target function γ turns out to be clear.

The same upper bound as in Theorem 3.3 was derived by Maurey [128] using a probabilistic argument, in which for any representation

$$\gamma = \sum_{i=1}^m a_i g_i$$

of a function γ as an element of the set $\text{conv } \mathcal{G}$, a finite probability distribution on the subset $\{g_1, \dots, g_m\}$ of the set \mathcal{G} is defined as $P(g = g_i) \triangleq a_i$. Then, an n -tuple h_1, \dots, h_n of elements of $\{g_1, \dots, g_m\} \subseteq \mathcal{G}$ is chosen randomly and independently with respect to the probability distribution P , and the random variable

$$\gamma_n \triangleq \frac{1}{n} \sum_{j=1}^n h_j,$$

which is the barycenter of the n -tuple, is considered. At this point, one can verify that the mean value $\|\gamma - \gamma_n\|_{\mathcal{H}}^2$ is bounded from above by $\frac{s_{\mathcal{G}}^2 - \|\gamma\|_{\mathcal{H}}^2}{n}$. Hence, there exists (at least) one n -tuple $h_1^*, \dots, h_n^* \in \{g_1, \dots, g_m\} \subseteq \mathcal{G}$ for which

$$\left\| \gamma - \frac{1}{n} \sum_{j=1}^n h_j^* \right\|_{\mathcal{H}} \leq \sqrt{\frac{s_{\mathcal{G}}^2 - \|\gamma\|_{\mathcal{H}}^2}{n^2}}.$$

The extension of this estimate to $\text{cl conv } \mathcal{G}$ gives the same upper bound as in Theorem 3.3.

Inspection of both the incremental proof and the probabilistic one shows that the approximation rate $\frac{s_{\mathcal{G}}^2 - \|\gamma\|_{\mathcal{H}}^2}{n}$ is achieved even when barycenters of n -tuples of elements of \mathcal{G} (with possible repetitions) are used as approximators, i.e.,

$$\|\gamma - \text{bary}_n \mathcal{G}\|_{\mathcal{H}} \leq \sqrt{\frac{s_{\mathcal{G}}^2 - \|\gamma\|_{\mathcal{H}}^2}{n}},$$

where

$$\begin{aligned} \text{bary}_n \mathcal{G} \triangleq & \left\{ \gamma \in \mathcal{H} : \gamma = \frac{1}{n} \sum_{i=1}^k j_i g_i, k \in \mathbb{Z}^+, j_i \in \mathbb{Z}^+, k \leq n, \right. \\ & \left. \sum_{i=1}^n j_i = n, g_i \in \mathcal{G}, i = 1, \dots, n \right\}. \end{aligned}$$

Let us now cast the upper bound of the order $O(1/\sqrt{n})$ provided by Theorem 3.3 into the framework of Sects. 2.6 and 2.7. Accordingly, let us refer again to the sequences of approximation Problems PA_n^d , for increasing values of the dimension d and of the model complexity n . As stated in Sect. 2.7, we are interested in understanding whether families of \mathcal{G}^d -approximating FSP functions exist that enable us

to achieve a desired accuracy in approximating the functions γ^d in Problems PA_n^d by using “moderate” basis cardinalities n when the dimension d increases.

Let $\varepsilon > 0$ be a given approximation error and, for every instance of Problem PA_n^d , let us consider an \mathcal{M}^d -approximating sequence $\{\mathcal{A}_n^d\}_{n=1}^\infty$. In general, the factor $\sqrt{s_G^2 - \|\gamma\|_{\mathcal{H}}^2}$ in the upper bound provided by Theorem 3.3 may depend on d . Let us assume that it grows at most polynomially with d , that is, $\sqrt{s_G^2 - \|\gamma\|_{\mathcal{H}}^2} \leq \bar{c} d^p$, for some suitable positive constant \bar{c} and some $p \in \mathbb{R}$, $p \geq 0$. Then, it is possible to determine a large enough model complexity n such that (compare with (2.71), with $q = 1/2$)

$$\bar{c} \frac{d^p}{\sqrt{n}} \leq \varepsilon. \quad (3.27)$$

Thus, we can denote by $n^*(d, \varepsilon)$ the minimum model complexity that verifies Inequality (3.27) and we write

$$n^*(d, \varepsilon) \triangleq \left\lceil \left(\frac{\bar{c}}{\varepsilon} \right)^2 d^{2p} \right\rceil. \quad (3.28)$$

Hence, the approximating sequence $\{\mathcal{A}_n^d\}_{n=1}^\infty$ exhibits a *favorable* behavior with respect to d , and, according to Definition 2.8, it is an ε -pc \mathcal{M}^d -approximating sequence of sets. Unfortunately, however, in the case of Hilbert spaces of d -variable functions, the factor $\sqrt{s_G^2 - \|\gamma\|_{\mathcal{H}}^2}$ in Theorem 3.3 may grow with d , even exponentially fast. This was noted in [6, Sect. IX], although its consequences seem to have been underestimated as regards the curse of dimensionality. The dependence of such a scalar on d and its repercussions on the model complexity have been investigated in [81, 82] in the context of tractability of approximation and optimization tasks whose functions depend on a large number of variables.

It is worth noting that the estimate of the distance from $\text{conv}_n \mathcal{G}$ provided by Theorem 3.3 is not restricted to Hilbert spaces. In [30], it was extended to \mathcal{L}_p -spaces for $p \in (1, \infty)$, obtaining a slightly worse rate of approximation (of order $O(n^{-\frac{1}{q}})$), where $q \triangleq \max(p, \frac{p}{p-1})$; there also exist extensions for \mathcal{L}_∞ (see, for instance, [5, 42, 60, 104, 113]).

3.7.2 Case Study: Sigmoidal OHL Networks

In [6], Theorem 3.3 was applied to OHL networks with sigmoidal computational units and an associated family of functions to be approximated. A comparison was made therein with the worst-case approximation error of the same families of functions by LCFBFs, too. The one from [6] is among the first investigations aimed at explaining the advantages of OHL networks over LCFBFs theoretically. Similar estimates were obtained in [5], measuring the worst-case approximation error in the

supremum norm. Later on, the comparison between approximation rates of OHL networks and the worst-case approximation by LCFBFs was further investigated; see, e.g., [46, 53, 54, 67, 96, 97] and the references therein. In the following, we provide some details on the results in [5, 6]. For the application of Theorem 3.3 to OHL networks with radial computational units, we refer the reader to [44].

We consider functions $f : \mathbb{R}^d \rightarrow \mathbb{R}$ characterized by a Fourier representation of the form

$$f(\mathbf{x}) = \int_{\mathbb{R}^d} e^{i\omega^\top \mathbf{x}} \tilde{F}(d\omega) \quad (3.29)$$

(here, i denotes the imaginary unit) for a complex-valued measure $\tilde{F}(d\omega) = e^{i\theta(\omega)} F(d\omega)$, where $F(d\omega)$ and $\theta(\omega)$ are the magnitude distribution and the phase at the angular frequency ω , respectively. For a finite $c > 0$ and a bounded subset D of \mathbb{R}^d containing $\mathbf{0}$, we denote by $\Gamma_{D,c}$ the set of functions $f(\mathbf{x})$ for which Representation (3.29) holds for a magnitude distribution F such that

$$\int_{\mathbb{R}^d} \sup_{\mathbf{x} \in D} |\omega^\top \mathbf{x}| F(d\omega) \leq c, \quad (3.30)$$

i.e.,

$$\Gamma_{D,c} \triangleq \left\{ f : \mathbb{R}^d \rightarrow \mathbb{R} : f(\mathbf{x}) = \int_{\mathbb{R}^d} e^{i\omega^\top \mathbf{x}} \tilde{F}(d\omega), \int_{\mathbb{R}^d} \sup_{\mathbf{x} \in D} |\omega^\top \mathbf{x}| F(d\omega) \leq c \right\}. \quad (3.31)$$

When D is the d -dimensional ball of radius r , we get

$$\sup_{\mathbf{x} \in \mathbb{R}^d : |\mathbf{x}| \leq r} |\omega^\top \mathbf{x}| = r|\omega|.$$

For other domains D , see [6, Sect. V]. Note that when d increases, Condition (3.30) becomes *more and more constraining*.

We can now state the following result, which follows from [6, Theorem 1].

Theorem 3.4 *Let D be a bounded subset of \mathbb{R}^d containing $\mathbf{0}$ and $c > 0$. For every $f \in \Gamma_{D,c}$, every sigmoidal function σ , and every positive integer n , there exists an OHL network of the form (3.7), i.e., a function*

$$\tilde{\gamma}(\mathbf{x}, \mathbf{w}_n) = \sum_{i=1}^n c_i \sigma(\mathbf{x}^\top \boldsymbol{\alpha}_i + \beta_i) + c_0,$$

such that

$$\sqrt{\int_D [f(\mathbf{x}) - \tilde{\gamma}(\mathbf{x}, \mathbf{w}_n)]^2 d\mathbf{x}} \leq \frac{2c}{\sqrt{n}} \mu(D), \quad (3.32)$$

where μ denotes the Lebesgue measure. \square

Inequality (3.32) shows that *at most*

$$n_1^* \triangleq \left\lceil \left(\frac{2c}{\varepsilon} \right)^2 \right\rceil$$

sigmoidal units are required to guarantee a worst-case approximation error ε in the \mathcal{L}_2 -norm to approximate functions belonging to $\Gamma_{D,c}$.

A substantial difference characterizes the case in which LCFBFs (3.1) are used to approximate functions belonging to the unit hypercube $[0, 1]^d$ (i.e., D is now given by $[0, 1]^d$) where a uniform probability measure is considered. Let $\text{span}\{\varphi_1, \dots, \varphi_n\}$ be the linear space spanned by the first n linearly independent basis functions of the LCFBFs. The worst-case \mathcal{L}_2 approximation error obtained by LCFBFs (3.1) is given by the Kolmogorov n -width (see Definition 3.3) of $\Gamma_{[0,1]^d,c}$ in the space $\mathcal{L}_2([0, 1]^d)$, for which the following lower bound was derived [6, Theorem 6]. As $\Gamma_{D,c}$ consists of functions defined on the whole \mathbb{R}^d , to consider the n -width of $\Gamma_{D,c}$ in $\mathcal{L}_2(D)$ we have to restrict its elements to D . So, we let $\bar{\Gamma}_{D,c} \triangleq \{f|_D : f \in \Gamma_{D,c}\}$, where $|_D$ denotes the restriction of the domain to the set D .

Theorem 3.5 *For every positive integer n ,*

$$d_n(\bar{\Gamma}_{[0,1]^d,c}, \mathcal{L}_2([0, 1]^d)) \geq \frac{c}{16\pi e^{\pi-1} d} \left(\frac{1}{2n} \right)^{1/d}. \quad (3.33)$$

□

In Theorem 3.5 we followed [47], according to which the factors $1/16$ and $1/(2n)$ should replace $1/8$ and $1/n$ in [6, Theorem 6], respectively. In any case, this does not change the essence of the result. The reader should also note that there is an unfortunate misprint in the Discussion in [6] on the lower bound provided by [6, Theorem 6], here reported as Theorem 3.5. Indeed, the factor $1/d (1/(2n))^{1/d}$ in the right-hand side of the bound is reported therein as $(1/(2n))^{1/d}$, omitting d in the denominator. Note that such a factor is substantial in determining the behavior of the lower bound for $d \rightarrow \infty$.

In contrast to OHL networks with sigmoidal computational units, if LCFBFs (3.1) are used, Inequality (3.33) implies that *at least*

$$n_2^* = \left\lfloor \frac{1}{2} \left(\frac{c}{16\pi e^{\pi-1} d} \right)^d \varepsilon^{-d} \right\rfloor$$

fixed-basis functions are required to guarantee a worst-case approximation error ε in the \mathcal{L}_2 -norm to approximate functions belonging to $\Gamma_{D,c}$ in the hypercube $[0, 1]^d$.

An issue that makes it difficult to exploit the behaviors of the upper bound from Theorem 3.4 and the lower bound from Theorem 3.5 to motivate the outperformance of sigmoidal OHL networks over LCFBFs for the approximation of certain sets of functions is the following. The presence of the factor $\mu(D)$ may have sub-

stantial consequences on the behavior of the right-hand side of Estimate (3.32), as remarked in [97, p. 273]. When, for instance, D is the unit ball B^d in \mathbb{R}^d with the l_2 -norm, $\mu(B^d) = \pi^{d/2}/\Gamma((d+2)/2)$ [24, p. 304], where Γ denotes the gamma function; hence, in such a case $\mu(D) \rightarrow \infty$ when $d \rightarrow \infty$. This strikingly contrasts, for example, with the domain $D = [0, 1]^d$, which is the case in which the n -width is estimated by Theorem 3.5 (see the remarks in [79, Sect. 18.2]). So, the dependence on d of the Lebesgue measure of the d -dimensional domain D has a strong impact on the usefulness of the comparison between the upper bound from Theorem 3.4 and the lower bound from Theorem 3.5.

To conclude, it is worth mentioning that, for the approximation by OHL networks, a bound similar to (3.32) was also obtained in [5] for the error in the supremum norm. Moreover, Bound (3.33) for the worst-case approximation by LCFBFs can be extended to a bound expressed in the supremum norm. We report both estimates in the forms presented in [47].

Theorem 3.6 *For every $f \in \Gamma_{D,c}$, every sigmoidal function σ , and every positive integer n , there exists a OHL network of the form (3.7), i.e.,*

$$\tilde{\gamma}(\mathbf{x}, \mathbf{w}_n) = \sum_{i=1}^n c_i \sigma(\mathbf{x}^\top \boldsymbol{\alpha}_i + \beta_i) + c_0,$$

such that

$$\sup_{\mathbf{x} \in D} |f(\mathbf{x}) - \tilde{\gamma}(\mathbf{x}, \mathbf{w}_n)| \leq \frac{120c}{\sqrt{n}}. \quad (3.34)$$

□

Theorem 3.7 *For every positive integer n ,*

$$d_n(\bar{\Gamma}_{[0,1]^d,c}, \mathcal{C}([0, 1]^d)) \geq \frac{c}{16\pi e^{\pi-1} d} \left(\frac{1}{2n} \right)^{1/d}. \quad (3.35)$$

□

The interpretation of the upper and lower bounds from Theorems 3.4, 3.5, 3.6, and 3.7 is not an easy task, for the following reason. As noted in [6], although there are many families of functions for which the integral in the left-hand side of (3.30) is bounded (see [6, Sect. IX]), the value of the bound may depend – and generally depends – on d in different ways. Hence, instead of c one has to write c_d . Let $\Gamma_{D,c}^{\text{poly}}$ denote a family of functions defined as in (3.31), for which the constant c_d increases at most polynomially with d (examples of classes of functions with an at most polynomial dependence of c_d on d are given in [6, 104]). In this case, if $\mathcal{M} \subseteq \Gamma_{D,c}^{\text{poly}}$, then OHL networks with sigmoidal computational units are polynomially complex approximating networks in \mathcal{M} . However, when c_d grows exponentially with d (see [6, Sect. IX]), the upper bounds from Theorems 3.4 and 3.6 do not allow the same conclusion to be drawn.

To sum up, the results reported in [6] do not yet fully explain the advantageous behavior of sigmoidal OHL networks in high-dimensional settings, although they represent a first attempt at comparing LCFBFs and OHL networks for the approximation of a wide class of functions.

In [47], the family of functions

$$\Gamma_{[0,1]^d, d', c} \triangleq \{f : \mathbb{R}^d \rightarrow \mathbb{R} : f \in \Gamma_{[0,1]^d, c} \text{ and } f \text{ depends only on } d' < d \text{ variables}\} \quad (3.36)$$

was considered. Note that $\Gamma_{[0,1]^d, d', c}$ is a subset of functions in $\Gamma_{[0,1]^d, c}$ that depend only on $d' < d$ arguments. The indices of such d' variables are not fixed but depend on the particular function $f \in \Gamma_{[0,1]^d, d', c}$. For such a family, in [47] a lower bound was derived on $d_n(\Gamma_{[0,1]^d, d', c}, \mathcal{L}_2([0, 1]^d))$ larger than the lower bound on $d_n(\Gamma_{[0,1]^d, c}, \mathcal{L}_2([0, 1]^d))$ provided by (3.33). In particular, for any fixed n and d' , [47, Proposition 2.4] shows that, for $d \geq 2n - 1$ one has

$$d_n(\Gamma_{[0,1]^d, d', c}, \mathcal{L}_2([0, 1]^d)) \geq \frac{c}{8\pi}. \quad (3.37)$$

Note that, for fixed c , the lower bound on $d_n(\Gamma_{[0,1]^d, d', c}, \mathcal{L}_2([0, 1]^d))$ does not tend to 0 as d tends to ∞ . Consider instead the lower bound on $d_n(\Gamma_{[0,1]^d, c}, \mathcal{L}_2([0, 1]^d))$ given by the right-hand side of (3.35), and assume that c is constant with respect to d . The factor $1/d$ determines the lower bound decreasing to 0 as d tends to ∞ . Then, this lower bound becomes less meaningful for large values of d . Loosely speaking, the Result (3.37) is due to the fact that LCFBFs are “not flexible enough” to identify the d' variables on which a function $f \in \Gamma_{[0,1]^d, d', c}$ actually depends. In contrast to this, sigmoidal OHL networks enjoy such a flexibility. All this is important whenever the functions to be approximated (or learned) depend only on a subset of $d' \ll d$ variables (i.e., *features*, in the machine learning parlance), and such a subset is not known a priori. This arises frequently in classification and regression tasks.

3.7.3 Lower Bounds on Approximation Rates

Whereas, as we have seen, many upper bounds on approximation rates by OHL networks are known, fewer lower bounds are available: limitations of computational capabilities of shallow networks are much less understood. Generally, proofs of lower bounds are much more difficult than arguments deriving upper bounds. Moreover, the available bounds are mostly nonconstructive and hold for types of computational units that are not commonly used [110, 111].

Worst-case errors exhibiting the curse of dimensionality in approximation of functions from Sobolev spaces [2, 17] by ridge OHL networks with logistic sigmoid and piecewise polynomial computational units were derived in [112]. An example of functions that cannot be efficiently computed by an OHL network was presented

in [11], where it was proved that for classification of points in the d -dimensional Boolean cube according to their parities by shallow Gaussian networks with fixed centers, at least 2^{d-1} units are necessary. Recently, in [101, 102] limitations of approximation capabilities of OHL networks have been investigated by deriving probabilistic lower bounds on approximation errors. To give a flavor of this kind of results, in [102] it was proven that unless the model complexity is larger than any polynomial of the logarithm of the size of the domain, a good approximation cannot be achieved for almost any uniformly randomly chosen function on a given domain.

In the sequel, we present and discuss a lower bound on the rates of approximation by ridge OHL networks, for functions in subsets of Sobolev spaces. We first recall some definitions and notations about such spaces (see, for instance, [2, 17] for comprehensive treatments on Sobolev spaces).

Definition 3.5 Let $\Omega \subseteq \mathbb{R}^d$ be an open set, s a nonnegative integer, and $\gamma : \Omega \rightarrow \mathbb{R}$ a function for which all partial derivatives up to order s exist almost everywhere in Ω . For the multi-index

$$\mathbf{k} \triangleq (k_1, \dots, k_d),$$

whose components k_1, \dots, k_d are nonnegative integers, let

$$|\mathbf{k}|_1 \triangleq \sum_{j=1}^d k_j$$

and

$$D^{\mathbf{k}} \gamma(\mathbf{x}) \triangleq \frac{\partial^{|\mathbf{k}|_1}}{\partial x_1^{k_1} \dots \partial x_d^{k_d}} \gamma(x) \quad \text{if } |\mathbf{k}|_1 = 1, \dots, s, \quad D^{\mathbf{0}} \gamma(\mathbf{x}) \triangleq \gamma(\mathbf{x}).$$

△

Definition 3.6 For an open set $\Omega \subseteq \mathbb{R}^d$, $p \in [1, \infty]$, and s a nonnegative integer, the Sobolev space $\mathcal{W}_p^s(\Omega)$ of order s consists of all the functions in $\mathcal{L}_p(\Omega)$ whose partial derivatives up to the order s are in $\mathcal{L}_p(\Omega)$, i.e.,

$$\mathcal{W}_p^s(\Omega) \triangleq \{\gamma : \Omega \rightarrow \mathbb{R} \text{ such that } D^{\mathbf{k}} \gamma \in \mathcal{L}_p(\Omega), |\mathbf{k}|_1 = 0, \dots, s\}.$$

In particular, for $p = 2$, $s = 1$, and $\Omega = (0, 1)^d$, the Sobolev space $\mathcal{W}_2^1((0, 1)^d)$ is the space of all functions in $\mathcal{L}_2((0, 1)^d)$ whose partial derivatives up to the order 1 are square-integrable.

△

Definition 3.7 For any function $\gamma \in \mathcal{W}_p^s(\Omega)$ with $p \in [1, \infty)$, its Sobolev norm is defined as

$$\|\gamma\|_{\mathcal{W}_p^s(\Omega)} \triangleq \left\{ \sum_{|\mathbf{k}|_1=0, \dots, s} \|D^{\mathbf{k}} \gamma\|_{\mathcal{L}_p(\Omega)}^p \right\}^{1/p}.$$

For any function $\gamma \in \mathcal{W}_\infty^s(\Omega)$, its Sobolev norm is defined as

$$\|\gamma\|_{\mathcal{W}_\infty^s(\Omega)} \triangleq \max_{|\mathbf{k}|_1=0,\dots,s} \|D^\mathbf{k}\gamma\|_{\mathcal{L}_\infty(\Omega)}.$$

△

Let

$$B^d \triangleq \{\mathbf{x} \in \mathbb{R}^d : |\mathbf{x}| \triangleq \sqrt{x_1^2 + \dots + x_d^2} \leq 1\}$$

denote the unit ball in \mathbb{R}^d with the Euclidean norm, and

$$\mathcal{B}_p^{s,d} \triangleq \{\gamma \in \mathcal{W}_p^s(B^d) : \|\gamma\|_{\mathcal{W}_p^s} \leq 1\}$$

the unit ball in the Sobolev space $\mathcal{W}_p^s(B^d)$ of order s . We consider the following set, containing all functions $f : \mathbb{R}^d \rightarrow \mathbb{R}$ that can be computed by d -variable single-output OHL networks with n basis functions (not necessarily all with the same structure) obtained via the ridge construction from functions that are integrable on any compact set of \mathbb{R} ; we denote by $\mathcal{L}^{\text{comp}}(\mathbb{R})$ the space of such functions (recall from Sect. 3.2.2 the definition of span_n):

$$\begin{aligned} \text{Ridge}_n^{\text{int}} &\triangleq \text{span}_n\{h(\mathbf{x}^\top \boldsymbol{\alpha}_i + \beta_i) : h \in \mathcal{L}^{\text{comp}}(\mathbb{R}), \boldsymbol{\alpha}_i \in \mathbb{R}^d, \beta_i \in \mathbb{R}\} \\ &= \{\gamma(\mathbf{x}) = \sum_{i=1}^n c_i h_i(\mathbf{x}^\top \boldsymbol{\alpha}_i + \beta_i) : h_i \in \mathcal{L}^{\text{comp}}(\mathbb{R}), \boldsymbol{\alpha}_i \in \mathbb{R}^d, c_i, \beta_i \in \mathbb{R}\}. \end{aligned}$$

The following theorem provides a lower bound on the rates of approximation by ridge OHL networks (see [110, Theorem 1]).

Theorem 3.8 *For all positive integers n and d and all nonnegative integers s , there exist $\gamma^* \in \mathcal{B}_2^{s,d}$ and a positive scalar $c_{d,s}$ (independent of γ^* and n), such that*

$$\inf_{\gamma_n \in \text{Ridge}_n^{\text{int}}} \|\gamma^* - \gamma_n\|_{\mathcal{L}_2(B^d)} \geq c_{d,s} \left(\frac{1}{n}\right)^{s/(d-1)}. \quad (3.38)$$

□

The order $\mathcal{O}\left(\left(\frac{1}{n}\right)^{s/d}\right)$ in (3.38) (recall the meaning of the “big- \mathcal{O} ” notation introduced in Sect. 2.5.2) is known as the *Jackson-type bound* or the *Jackson rate* [32]. Theorem 3.8 means the following: no matter how large the model complexity of an OHL network in $\text{Ridge}_n^{\text{int}}$ is taken to be, there exists a function in $\mathcal{B}_2^{s,d}$ that cannot be approximated with an error smaller than $c_{d,s} n^{-s/(d-1)}$ (in the \mathcal{L}_2 -norm). Estimating the factor $c_{d,s}$ is not an easy task; however, typically it is a nondecreasing function

of d . So, unless the order s of the Sobolev space is made suitably dependent on d , (3.38) may exhibit the curse of dimensionality.⁸

The fact that the approximation of elements of the Sobolev ball $\mathcal{B}_p^{s,d}$ by OHL networks in $\text{Ridge}_n^{\text{int}}$ has a Jackson-type lower bound implies the following: if the functions to be approximated are s -time continuously differentiable and depend on d variables, non-exponential upper estimates of the minimum number of parameters needed for a given approximation accuracy may be obtained via such networks only by letting the smoothness increase suitably with the dimension d .

In terms of deviation, Theorem 3.8 implies the following lower bound on the deviation of the ball $\mathcal{B}_2^{s,d}$ in the Sobolev norm from the set $\text{Ridge}_n^{\text{int}}$ (the function γ^* used below is defined in Theorem 3.8):

$$\begin{aligned} \delta(\mathcal{B}_2^{s,d}, \text{Ridge}_n^{\text{int}}) &= \inf_{\gamma_n \in \text{Ridge}_n^{\text{int}}} \sup_{\gamma \in \mathcal{B}_2^{s,d}} \|\gamma - \gamma_n\|_{\mathcal{L}_2(B^d)} \\ &\geq \inf_{\gamma_n \in \text{Ridge}_n^{\text{int}}} \|\gamma^* - \gamma_n\|_{\mathcal{L}_2(B^d)} \\ &\geq c_{d,s} \left(\frac{1}{n}\right)^{s/(d-1)}. \end{aligned} \quad (3.39)$$

It is worth remarking that in [110] it was proven that the set of functions

$$\{\gamma^* \in \mathcal{B}_p^{s,d} : \text{Lower Bound (3.38) holds}\}$$

is “large”, in the sense that there exist “many” functions in correspondence with which Lower Bound (3.38) is achieved (see [110] for the mathematical formalization of the informal expressions “large” and “many” used above).

As Theorem 3.8 applies to OHL networks obtained via the ridge construction from functions in $\mathcal{L}^{\text{comp}}(\mathbb{R})$, another possibility of avoiding its above-discussed implications is to use variable-basis functions that are not integrable on any compact set of \mathbb{R} : for them, Lower Bound (3.38) might not hold.

A topological approach to deriving lower bounds of Jackson-type for nonlinearly parametrized approximators was proposed in [32]. It is based on the extension of the concept of n -width to a “continuous nonlinear n -width,” defined for linear combinations of parametrized basis functions whose parameters depend continuously on the function to be approximated. However, it was shown in [83–85] that for a variety of OHL networks typically used in applications (such as those with basis functions corresponding to the hyperbolic tangent, the Heaviside function, and the Gaussian function), the hypothesis of continuous dependence of the parameters does not hold. Hence, the lower bounds derived in [32] do not apply to such networks.

⁸For example, when $c_{d,s} \rightarrow c_{\infty,s} > 0$ for $d \rightarrow \infty$ and the order s of the Sobolev space does not depend on d , for any positive integer n , the lower bound $c_{d,s} n^{-s/(d-1)}$ tends to $c_{\infty,s}$ as d tends to ∞ . Hence, for any desired approximation error (strictly) smaller than $c_{\infty,s}$, in this case any fixed n is not able to guarantee such accuracy when d is sufficiently large.

3.7.4 On the Approximation Accuracy of MHL Networks

As we have seen, one hidden layer is sufficient to achieve \mathcal{C} - and \mathcal{L}_p -density. However, from the point of view of approximation rates, using one or more than one hidden layer might be substantially different. The literature on networks with more than one hidden layer is quite limited. Relatively little is known about the approximation properties of models with more hidden layers and about the pros and cons of using a single hidden layer with many computational units over many hidden layers with a smaller number of such units (see [31] for a discussion). Theoretical analysis that complements empirical results and identifies tasks for which OHL networks require considerably more units than do MHL networks is the subject of current research.

Networks with several hidden layers are often called *deep* (see, e.g., [9, 23, 59, 65] and the references therein), in contrast to *shallow* nets, which have merely one hidden layer and so correspond to OHL networks. Recently proposed algorithms to tune the parameters in multi-hidden-layers motivated a renaissance of interest in MHL networks leading to the so-called “deep learning revolution” already mentioned in Chap. 1. We refer again the reader, e.g., to the seminal papers [65, 66] and the recent reference book [59]. Advantages of MHL networks with two hidden layers over OHL ones were pointed out in [140], in an application to the stabilization of nonlinear control systems. In [12] it was conjectured that “most functions that can be represented compactly by deep architectures cannot be represented by a compact shallow architecture.” It was suggested in [10, 11] that a cause of large model complexities of OHL networks might lie in the “amount of variations” of functions to be computed. Following this suggestion, to study limitations of OHL networks in [101] variations of functions were investigated in terms of \mathcal{G} -variation norm, which is defined in Sect. 3.8.

An approach to the investigation of the approximation of shallow and deep networks based on topological characteristics of input–output functions was proposed in [13]. In [115, 116] it was argued that deep networks are particularly suitable for the computation of composite functions. On the other hand, the empirical study in [4] showed that in some cases, shallow networks can learn functions previously learned by deep ones using the same numbers of parameters. For recent works comparing the approximation capabilities of OHL and MHL networks, we refer the reader to [119] and the references therein. Finally, the recent work [7] investigates approximation and estimation for high-dimensional deep networks.

In the next theorem, we report a result from [111] that, as noted in [127, Sect. 7], suggests that there might be advantages in using more than one hidden layer in ridge OHL networks.

Theorem 3.9 *There exists a strictly increasing, real analytic sigmoidal basis function σ such that for every $f \in \mathcal{C}([0, 1]^d)$ and every $\varepsilon > 0$ there exist d_i , c_{ij} , β_{ij} , $\delta_i \in \mathbb{R}$ and $\alpha_{ij} \in \mathbb{R}^d$ for which, for all $\mathbf{x} \in [0, 1]^d$, one has*

$$\left| f(\mathbf{x}) - \sum_{i=1}^{6d+3} d_i \sigma \left(\sum_{j=1}^{3d} c_{ij} \sigma(\mathbf{x}^\top \boldsymbol{\alpha}_i + \beta_i) + \delta_i \right) \right| < \varepsilon.$$

□

Theorem 3.9 means that there exists a “very smooth” sigmoidal basis function with the following property: every continuous function on the d -dimensional unit cube can be approximated up to any desired degree of accuracy in the supremum norm, by a two-hidden-layer network with $6d + 3$ and $3d$ such basis functions in the first and second hidden layers, respectively. Unfortunately, inspection of the proof of Theorem 3.9 shows that such a basis function is extremely complex to construct, which makes the result have almost no constructive interest. However, the importance of Theorem 3.9 consists in showing that by using two hidden layers one may theoretically overcome the intrinsic limitation of OHL networks expressed by Theorem 3.8. This provides a strong motivation to investigate networks with more than one hidden layer. We refer the reader to [127, Sect. 7] for a deeper discussion on Theorem 3.9, its proof, and its implications.

We conclude this section with a remark on the possibility of extending to MHL networks upper bounds for OHL networks. We do this referring to the case study of feedforward sigmoidal neural networks, addressed in Sect. 3.7.2. Combined with a Lipschitz continuity assumption, the upper bound (3.34) expressed in the supremum norm can be extended to an upper bound on the MHL approximation error of composite functions, still of order $O(n^{-1/2})$, where n is the model complexity of the MHL network. In this case, each component function belongs to $\Gamma_{D,c}$, and the final bound still refers to the supremum norm. This kind of result was explicitly pointed out in a recent investigation of MHL approximation capabilities [119]. However, the same kind of result was exploited previously in [57], where it was applied to obtain suboptimal solutions of optimization problems over stages through neural network approximations of the optimal policy functions. It is worth noting that a similar construction cannot be made starting from Upper Bound (3.32) on the worst-case approximation error in the \mathcal{L}_2 -norm, unless additional smoothness conditions are imposed. This highlights the importance of worst-case approximation error bounds in the supremum norm.

3.8 Extensions of the Maurey–Jones–Barron Theorem

Inspired by Theorem 3.3, several other estimates were derived in the literature about OHL networks on the worst-case approximation rates of some sets of functions in \mathcal{C} or \mathcal{L}_p spaces. Such estimates are sometimes called “dimension-independent”, but this is misleading since the condition of belonging to the sets on which they apply often becomes more and more constraining when the dimension d increases. Indeed, these estimates include several factors, one of which involves the number n of basis

functions, while another involves the number d of variables. The dependence on the dimension d is often cryptic, i.e., estimates usually involve parameters that are constant with respect to n but do depend on d and the manner of dependence is not specified; see, for instance, [5, 6, 16, 30, 33, 44, 60, 77, 113]. In some of the literature (see, e.g., [6]) the terms depending on d are referred to as “constants” since those papers focus on the number n of computational units and assume a fixed value for the dimension d of the input space. Such estimates are formulated therein in the form $O(\xi(n))$, where $\xi : \mathcal{N}^+ \rightarrow \mathbb{R}$ and the dependence on d is hidden in the “big O” notation. However, in some cases, it has been shown that such “constants” actually grow at an exponential rate in d [92, 104].

An emphasis on the model complexity n is certainly reasonable when the number d of variables is fixed, but in modern research, where technology allows ever-increasing amounts of data to be collected, it is natural to also take into account the role of d , as we do in this book. As remarked in [144], the dependence of approximation errors on d is generally much harder to estimate than the dependence on n . Not many such estimates are available. They are useful to investigate when, for a desired accuracy, approximation problems can be solved in a computationally efficient way as the dimension d of the input space grows. To deal with this issue, in [81, 82], conditions were derived on sets of functions for which the “constants” hidden in the “big O” notation above grow at most at a polynomial rate. However, a polynomial growth may also not provide control of model complexity unless the degree of the polynomial is quite small. For instance, in the case of a large dimension d of the input space, even a quadratic growth is not going to be sufficient. In [82], cases were studied where the dependence on d is *linear*, and others in which the dimension-dependent “constant” *decreases* – even exponentially fast – with the dimension. In [100, 105], geometric rates of approximation with respect to the number of computational units were derived, which improve the rate of approximation provided by Theorem 3.3, for functions belonging to suitable subsets of $\text{cl conv } \mathcal{G}$. Finally, in [89], Theorem 3.3 has been generalized in terms of a norm, called the \mathcal{G} -variation norm, whose investigation allows one to obtain insights into the properties of the sets of functions $\text{cl conv } \mathcal{G}$ to which the theorem applies. Geometric approximation rates and \mathcal{G} -variation are considered in the next two sections.

3.8.1 Geometric Rates of Approximation

Several authors have derived improvements of the $O(\varepsilon^{-2})$ upper bound on the minimal number n of computational units guaranteeing a desired approximation error ε , which is provided by Theorem 3.3 for various approximating sets \mathcal{G} (e.g., \mathcal{G} orthogonal [96, 104], \mathcal{G} formed by functions computable by sigmoidal perceptrons, and \mathcal{G} with certain properties of covering numbers [99, 113]). However, all these results are worst-case estimates in the sense that they give upper bounds holding for all functions from the closure of the symmetric convex hull of \mathcal{G} (i.e., the convex hull of

the set $\mathcal{G} \cup -\mathcal{G}$). Thus one can expect that better rates may hold for suitable subsets of such hull.

A step toward a description of subsets with better rates of approximation was done in [105]. It was noted therein that when the iterative construction derived in [77] and improved in [6] is applied to functions satisfying a certain angular condition, then the term $\frac{1}{n}$ in the statement of Theorem 3.3 can be replaced with τ^{n-1} , where $\tau \in [0, 1)$ is the cosine corresponding to the angular constraint. However, in [105] the problem of the characterization of functions satisfying such an angular condition was left open. Indeed, the obtained result was illustrated by one example, in which, as it was remarked [105, p. 280], “geometric convergence ... is evident and easy to establish without the use of” the angular condition itself.

In [100], it was proved that for every function γ in the convex hull of a bounded subset \mathcal{G} of a Hilbert space there exists $\tau_\gamma \in [0, 1)$ such that the rate of approximation of f by convex combinations of n functions from \mathcal{G} is bounded from above by $\sqrt{\tau_\gamma^{n-1} (s_{\mathcal{G}}^2 - \|\gamma\|_{\mathcal{H}}^2)}$. Such a geometric rate was derived by modifying the incremental construction originally used in [77], later improved in [6], then refined in [105]. In [100] it was also proved that a similar estimate holds for all functions from the linear span of \mathcal{G} in the approximation by linear combinations of n elements of \mathcal{G} . The next theorem reports the main result from [100].

Theorem 3.10 *Let \mathcal{H} be a Hilbert space, \mathcal{G} its bounded non-empty subset, and $s_{\mathcal{G}} \triangleq \sup_{g \in \mathcal{G}} \|g\|_{\mathcal{H}}$. For every $\gamma \in \text{conv } \mathcal{G}$ there exists $\tau_\gamma \in [0, 1)$ such that for every positive integer n*

$$\|\gamma - \text{conv}_n \mathcal{G}\|_{\mathcal{H}} \leq \sqrt{\tau_\gamma^{n-1} (s_{\mathcal{G}}^2 - \|\gamma\|_{\mathcal{H}}^2)}. \quad \square$$

Theorem 3.10 was illustrated in [100] by estimating values of parameters of geometric rates when \mathcal{G} is an orthonormal basis; insights into the structure of sets of functions with fixed values of parameters of such rates were derived therein, too. It is worth noting, however, that the geometric rate provided by Theorem 3.10 does not extend to functions $\gamma \in \text{cl conv } \mathcal{G}$, as shown in [100]. As for Theorem 3.3, the proof of Theorem 3.10 is based on a constructive incremental procedure; see [100] for details.

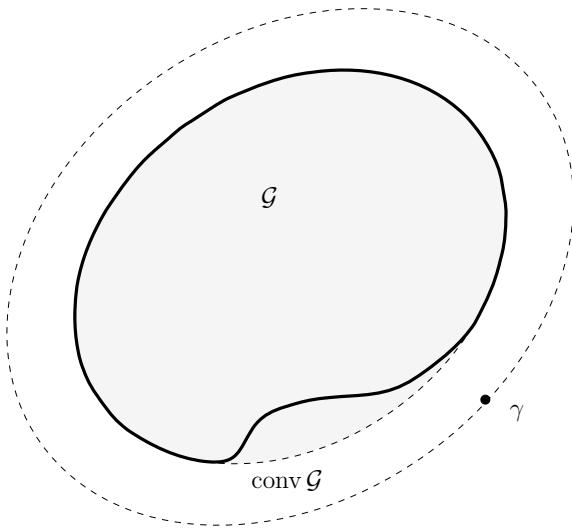
3.8.2 An Upper Bound in Terms of the \mathcal{G} -Variation Norm

The greedy construction of approximants in Theorem 3.3 has given a hint to the definition of a norm tailored to a bounded set \mathcal{G} in a Hilbert space, called “ \mathcal{G} -variation norm,” introduced in [89], and investigated in several papers (see, for example, [46, 56, 58, 80, 93, 95–97, 104] and the references therein). The idea is the following.

If in Theorem 3.3, for any $c \in \mathbb{R}^+$, one replaces the set \mathcal{G} with

$$\mathcal{G}(c) \triangleq \{wg : g \in \mathcal{G}, w \in \mathbb{R}, |w| \leq c\},$$

Fig. 3.8 A geometric interpretation of the \mathcal{G} -variation norm



one can apply the derived estimate to all elements of $\text{cl conv } \mathcal{G}(c)$, obtaining a different upper bound on the approximation error for each c . This approach can be formulated in terms of a norm tailored to a given subset \mathcal{G} of a normed linear space \mathcal{G} (in particular, to a set \mathcal{G} of the form \mathcal{G}_φ corresponding to various OHL networks), called the \mathcal{G} -variation norm (or the *variation with respect to \mathcal{G}*) and defined as follows:

$$\|\gamma\|_{\mathcal{G}var} \triangleq \inf\{c \in \mathbb{R}^+ : \gamma \in \text{cl conv } \mathcal{G}(c)\}. \quad (3.40)$$

This is a norm on the subspace

$$\{\gamma \in \mathcal{G} : \|\gamma\|_{\mathcal{G}var} < \infty\} \subseteq \mathcal{G}.$$

It is worth observing that the closure in Definition (3.40) is with respect to the norm of the space \mathcal{G} . Intuitively, $\|\gamma\|_{\mathcal{G}var}$ is the minimal dilation of the set \mathcal{G} (if $\|\gamma\|_{\mathcal{G}var} \geq 1$), or its maximal shrinking (if $\|\gamma\|_{\mathcal{G}var} < 1$), guaranteeing that γ is in the set $\text{cl conv } \mathcal{G}(c)$. In Fig. 3.8, a geometrical interpretation in which the set \mathcal{G} is a closed and bounded subset of the space \mathbb{R}^2 with the Euclidean norm induced by the usual inner product is provided. The term “ \mathcal{G} -variation norm” was introduced in [89] as an extension of the concept of variation with respect to half-spaces from [5]. It is also a generalization of the l_1 -norm and of the concept of *total variation* studied in integration theory [88, p. 328].

When the set \mathcal{G} is made of the parametrized basis functions used by OHL networks, the \mathcal{G} -variation norm becomes a *norm tailored to the corresponding kind of OHL network* and provides a mathematical framework for the study of approximation schemes that mitigate the curse of dimensionality. Exploiting the approach based on such a variational norm, a recent line of research has given insights into the rates of

approximation and optimization of OHL networks and of the ERIM, respectively; see [46, 53, 54, 56, 58, 67, 78, 81, 82, 96–98, 100–103] and the references therein.

In [89] (see also [91]), Theorem 3.3 was extended as follows using the concept of the \mathcal{G} -variation norm.

Theorem 3.11 *Let \mathcal{H} be a Hilbert space, \mathcal{G} its bounded non-empty subset, and $s_{\mathcal{G}} = \sup_{g \in \mathcal{G}} \|g\|_{\mathcal{H}}$. Then for every $\gamma \in \mathcal{H}$,*

$$\|\gamma - \text{span}_n \mathcal{G}\|_{\mathcal{H}} \leq \sqrt{\frac{(s_{\mathcal{G}} \|\gamma\|_{\mathcal{G}var})^2 - \|\gamma\|_{\mathcal{H}}^2}{n}}. \quad (3.41)$$

□

Upper Bound (3.41) is useful only when $\|\gamma\|_{\mathcal{G}var} < \infty$. Moreover, as $0 \in \text{span}_n \mathcal{G}$, for every $\gamma \in \mathcal{H}$ one has $\|\gamma - \text{span}_n \mathcal{G}\|_{\mathcal{H}} \leq \|\gamma\|_{\mathcal{H}}$. Thus Theorem 3.11 provides a nontrivial upper bound only when

$$\|\gamma\|_{\mathcal{H}} > \sqrt{\frac{(s_{\mathcal{G}} \|\gamma\|_{\mathcal{G}var})^2 - \|\gamma\|_{\mathcal{H}}^2}{n}}$$

or equivalently

$$\frac{\|\gamma\|_{\mathcal{H}}}{s_{\mathcal{G}} \|\gamma\|_{\mathcal{G}var}} > \frac{1}{\sqrt{n+1}}$$

(e.g., for $s_{\mathcal{G}} = 1$ and $\|\gamma\|_{\mathcal{G}var} = 1$ this means that $\|\gamma\|_{\mathcal{H}} \geq \frac{1}{\sqrt{n+1}}$).

Since, for every set \mathcal{G} , one has the inclusion $\text{conv}_n \mathcal{G} \subseteq \text{span}_n \mathcal{G}$, Theorem 3.11 implies the following upper bounds on the deviation from $\text{span}_n \mathcal{G}$ of the unit ball in the \mathcal{G} -variation norm and of its subsets defined by a constraint on the value of the norm $\|\cdot\|_{\mathcal{H}}$.

Corollary 3.1 *Let \mathcal{H} be a Hilbert space, \mathcal{G} be its subset, and $0 \leq r \leq s_{\mathcal{G}} \triangleq \sup_{g \in \mathcal{G}} \|g\|_{\mathcal{H}}$. Then for every positive integer n ,*

$$\delta(B_1(\|\cdot\|_{\mathcal{G}var}), \text{span}_n \mathcal{G}) \leq \frac{s_{\mathcal{G}}}{\sqrt{n}};$$

$$\delta(\{\gamma \in B_1(\|\cdot\|_{\mathcal{G}var}), \|\gamma\|_{\mathcal{H}} \geq r\}, \text{span}_n \mathcal{G}) \leq \sqrt{\frac{s_{\mathcal{G}}^2 - r^2}{n}}. \quad \square$$

Hence, all functions belonging to the unit ball in the \mathcal{G} -variation norm can be approximated within $s_{\mathcal{G}}/\sqrt{n}$ by linear combinations of n elements of \mathcal{G} . However, with an increasing number of variables, the condition of being in the unit ball in the \mathcal{G} -variation norm may become more and more constraining. It is worth noting that, by expressing rates of approximation in terms of \mathcal{G} -variation, the dependence on the

dimension d is constrained only in the way in which the unit ball varies: typically, it becomes “smaller” with d . In other words, the larger d is, the more constrained the unit ball $B_1(\|\cdot\|_{\mathcal{G}var})$ is expected to be. In any case, using the \mathcal{G} -variation, the ambiguity of the “constants” that may depend on d , discussed at the beginning of this section, is avoided.

Exploiting the approach based on \mathcal{G} -variation, in [46, 53, 54, 67, 96, 97] the comparison between approximation rates of OHL networks and the worst-case approximation by LCFBF, started in [5, 6], was investigated.

3.8.3 Estimating the \mathcal{G} -Variation

In general, the \mathcal{G} -variation norm is difficult to compute; so the possibility of deriving on it upper bounds in terms of other norms has been investigated in the literature. The \mathcal{G} -variation norms associated with certain classes of functions to be approximated and sets \mathcal{G} have been bounded from above in terms of various “classical” norms, which can be more easily computed. The first results in this direction were derived in [6, 16, 77], where upper bounds on the \mathcal{G} -variation norm were derived in terms of various spectral norms (i.e., norms expressed in terms of the Fourier transform), for sets \mathcal{G} corresponding to sines and cosines with variable frequencies [77], sigmoids [6], and so-called “hinging hyperplanes” [16]. Indeed, the inspection of the proofs of the estimates obtained in [6] for the approximation by sigmoidal OHL networks (see Sect. 3.7.2) shows that they are derived by estimating the variation with respect to sigmoids with variable weights and biases via a spectral norm. Other upper bounds in terms of spectral norms can be found in [104].

In [94, 95], the \mathcal{G} -variation norm with respect to a class of parametrized functions was bounded from above for functions that can be represented by a suitable integral formula (where integration is with respect to a continuous parameter), in terms of the \mathcal{L}_1 -norm of the weighting function associated to the parameter in such an integral representation. In [58], conditions were obtained under which, for such parametrized functions, their \mathcal{G} -variation norm is equal to the \mathcal{L}_1 -norm of the weighting function. Some upper bounds on the \mathcal{G} -variation norm in terms of the \mathcal{L}_1 -norm of the weighting function and under minimal assumptions on the set \mathcal{G} were investigated in [80] using the Bochner integral and in [93] exploiting a geometric characterization of the \mathcal{G} -variation norm. In [96, 104], the \mathcal{G} -variation norm was estimated in terms of l_1 - and l_2 -norms. Estimates via \mathcal{L}_2 - and Sobolev norms can be found in [46]. In [56], upper bounds on the \mathcal{G} -variation norm were derived for spaces associated with translation-invariant computational units and in terms of the classical \mathcal{L}_2 -, Bessel, and Sobolev norms.

3.8.4 Some OHL Networks with the Same Approximation Rate

Considering sets \mathcal{G} made up of different kinds of parametrized computational units and estimating \mathcal{G} -variation in terms of suitable classical or spectral norms allows the derivation of families of OHL networks with the same approximation rate. Table 3.2 from [46] slightly modifies [45, Table 3, p. 255] and expresses it in terms of OHL networks. The table shows various approximation schemes of the form $\{\text{span}_n \mathcal{G}_\varphi\}$ and the corresponding sets of d -variable functions that can be approximated by OHL networks with a basis function φ at a rate of order $O(n^{-1/2})$. The “ambient space” in the first column of the table is the normed linear space in which the approximation error is measured.

Note that the various sets of functions in the third column of Table 3.2 become more and more constrained as the number d of variables increases. This is evident in the case of Sobolev spaces (see the fourth, sixth, and seventh row in the table). The sets Θ_1 , Γ_1 , and Λ_1 in the first three rows are defined, respectively, as

Table 3.2 Approximation schemes $\{\text{span}_n \mathcal{G}_\varphi\}$ and corresponding sets of functions $f : \mathbb{R}^d \rightarrow \mathbb{R}$ with an upper bound of order $O(n^{-1/2})$ on the approximation error. Ω denotes an open subset of \mathbb{R}^d . The function $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ in the second row is sigmoidal. In the third row, $\kappa : \mathbb{R} \rightarrow \mathbb{R}$ denotes the ramp function, defined as $\kappa(t) \triangleq 0$ for $t < 0$ and $\kappa(t) \triangleq t$ for $t \geq 0$. The function $\xi : \mathbb{R} \rightarrow \mathbb{R}$ in the fourth row has to satisfy a technical condition (see [117] for details); for example, the squashing function, the generalized quadratics, the thin plate splines, and the Gaussian are allowed. In the fifth row, $\Psi_1 \triangleq \left\{ f : e^{-|x|^2} * \lambda, \lambda \in \mathcal{L}_1, \text{ and } \|\lambda\|_{\mathcal{L}_1} \leq 1 \right\}$, where the symbol $*$ denotes convolution and λ is any function with an integrable Fourier transform, and with the \mathcal{L}_1 -norm smaller than or equal to 1. In the sixth row, for $r > 0$, $\beta_r : \mathbb{R}^d \rightarrow \mathbb{R}$ is a Bessel potential of order r , i.e., the function whose Fourier transform is $\tilde{\beta}_r(\omega) = \frac{1}{(1+|\omega|^2)^{r/2}}$

Ambient space	Approximation scheme $\{\text{span}_n \mathcal{G}_\varphi\}$	Set of functions with approximation rate $O(n^{-1/2})$	References
$\mathcal{L}_2(\Omega)$	$\mathcal{G}_{\sin} = \left\{ f(\mathbf{x}) = \sin(\mathbf{x}^\top \mathbf{w} + \theta), \mathbf{w} \in \mathbb{R}^d, \theta \in \mathbb{R} \right\}$	Θ_1	[77]
$\mathcal{L}_2(\Omega)$	$\mathcal{G}_\sigma = \left\{ f(\mathbf{x}) = \sigma(\mathbf{x}^\top \mathbf{w} + \theta), \mathbf{w} \in \mathbb{R}^d, \theta \in \mathbb{R} \right\}$	Γ_1	[6]
$\mathcal{L}_2(\Omega)$	$\mathcal{G}_\kappa = \left\{ f(\mathbf{x}) = \kappa(\mathbf{x}^\top \mathbf{w} + \theta), \mathbf{w} \in \mathbb{R}^d, \theta \in \mathbb{R} \right\}$	Λ_1	[16]
$\mathcal{L}_p(\Omega)$ $1 \leq p \leq \infty$	$\mathcal{G}_\xi = \left\{ f(\mathbf{x}) = \xi(\mathbf{x}^\top \mathbf{w} + \theta), \mathbf{w} \in \mathbb{R}^d, \theta \in \mathbb{R} \right\}$	$\mathcal{B}_p^{[d/2],d}$	[117]
$\mathcal{L}_\infty(\Omega)$	$\mathcal{G}_{\exp} = \left\{ f(\mathbf{x}) = e^{- \mathbf{x}-\mathbf{w} ^2}, \mathbf{w} \in \mathbb{R}^d \right\}$	Ψ_1	[43]
$\mathcal{L}_\infty(\Omega)$	$\mathcal{G}_{\beta_r} = \left\{ f(\mathbf{x}) = \beta_r(\mathbf{x}-\mathbf{w} ^2), \mathbf{w} \in \mathbb{R}^d \right\}$	$\mathcal{B}_2^{2m,d}$ $2m > d$	[43]
$\mathcal{L}_2(\Omega)$	$\mathcal{G}_{\exp(\delta)} = \left\{ f(\mathbf{x}) = e^{- \mathbf{x}-\mathbf{w} ^2/\delta^2}, \mathbf{w} \in \mathbb{R}^d, \delta \in \mathbb{R} \right\}$	$\mathcal{B}_2^{2m,d}$ $2m > d$	[41]

$$\Theta_1 \triangleq \left\{ f : \mathbb{R}^d \rightarrow \mathbb{R} \text{ such that } \int_{\mathbb{R}^d} |\tilde{f}(\omega)| d\omega \leq 1 \right\},$$

$$\Gamma_1 \triangleq \left\{ f : \mathbb{R}^d \rightarrow \mathbb{R} \text{ such that } \int_{\mathbb{R}^d} |\omega| |\tilde{f}(\omega)| d\omega \leq 1 \right\},$$

and

$$\Lambda_1 \triangleq \left\{ f : \mathbb{R}^d \rightarrow \mathbb{R} \text{ such that } \int_{\mathbb{R}^d} |\omega|^2 |\tilde{f}(\omega)| d\omega \leq 1 \right\}.$$

As to the second and third rows, the constraints on the Fourier transform can be also written, respectively, as $\Gamma_1 \triangleq \{f : \mathbb{R}^d \rightarrow \mathbb{R}; f(\mathbf{x}) = |\mathbf{x}|^{1-d} * \lambda\}$ and $\Lambda_1 \triangleq \{f : \mathbb{R}^d \rightarrow \mathbb{R}; f(\mathbf{x}) = |\mathbf{x}|^{2-d} * \lambda\}$, where $*$ denotes convolution and λ is any function with an integrable Fourier transform, with the \mathcal{L}_1 -norm smaller than or equal to a suitable positive constant [43, 44]. As remarked in [45], for an increasing number d of variables, the terms $|\mathbf{x}|^{1-d}$ and $|\mathbf{x}|^{2-d}$ are more and more rapidly decaying, and the sets $\{f : \mathbb{R}^d \rightarrow \mathbb{R}; f(\mathbf{x}) = |\mathbf{x}|^{1-d} * \lambda\}$ and $\{f : \mathbb{R}^d \rightarrow \mathbb{R}; f(\mathbf{x}) = |\mathbf{x}|^{2-d} * \lambda\}$ become more and more constrained.

In the fourth row of Table 3.2, the set of admissible solutions with an approximation rate of order $O(n^{-1/2})$ is the unit ball in a Sobolev space, and the error is measured in the $\mathcal{L}_p(\Omega)$ -norm, where Ω is an open subset of \mathbb{R}^d . We can compare this result with those on approximation by LCFBFs, expressed in terms of the Kolmogorov n -width. Let $B_p^{s,d}(\Omega)$ be the unit ball of the Sobolev space $\mathcal{W}_p^s(\Omega)$ and $\Omega = (0, 1)^d$ (but the comparison can be extended to general open and bounded sets of \mathbb{R}^d [125, p. 233]). The following estimate holds [125, p. 232]:

$$d_n(B_p^{s,d}((0, 1)^d), \mathcal{L}_q((0, 1)^d)) = O\left(\frac{1}{n}\right)^{\frac{s}{d}}, \quad 1 \leq q \leq p \leq \infty \\ \text{or } 2 \leq p \leq q \leq \infty. \quad (3.42)$$

The rate $O(n^{-1/2})$ is obtained from (3.42) by letting $s = \lceil d/2 \rceil$, i.e., by constraining the set of functions more and more with an increasing number d of variables. In [46], various spaces of functions defined by spectral norms, each with an associated kind of OHL network that allows to get an $O(n^{-1/2})$ approximation rate, were compared with one another and with classical \mathcal{L}_p and Sobolev spaces.

3.9 The ERIM Versus the Classical Ritz Method

Since the birth of the Ritz method in 1909 [130], the literature has described rather few applications of it for the solution of IDO problems such as those arising in closed-loop stochastic optimal control. As for closed-loop optimal control, to the best of our knowledge, the Ritz method has been typically applied in deterministic contexts. In this case, the control function takes an open-loop form, thus depending

on time, i.e., on only one variable. For example, in [138], the deterministic nonlinear optimal control Problem 1.2 was addressed without imposing Constraints (1.15). The open-loop control function $\mathbf{u}(t)$ was approximated by LCFBFs (3.2). Such an approach is called the *control parametrization Ritz method*. In [138], the convergence of the approximate control and state trajectories to the optimal ones, when the model complexity n goes to infinity, was demonstrated under suitable local sufficient conditions (\mathcal{L}_2 -errors between the approximate and optimal trajectories were considered). Such convergence results were specialized for the case in which the basis functions are given by splines and the related solution was shown to be globally optimal in a linear-quadratic case.

Estimates of the quantities $\|\boldsymbol{\gamma}^\circ - \boldsymbol{\gamma}_n^\circ\|_{\mathcal{H}}$ and $F(\boldsymbol{\gamma}_n^\circ) - F(\boldsymbol{\gamma}^\circ)$ when the Ritz method is used in the approximate solution of Problem P^d were obtained, e.g., in [3, 15, 28, Sect. 3.7], [29, 35, 38, 61, 114, 120, 142]. However, such results either pertain to the case $d = 1$ (where the one-dimensional variable is time) or provide upper bounds which, although applicable to multidimensional cases, do not make the dependence on d explicit. To sum up, to the best of our knowledge, available estimates on the Ritz method for optimal control leave unanswered the question of whether it is possible to obtain arbitrarily accurate suboptimal solutions by means of moderate values of the model complexity n whenever the admissible control functions depend on a large number d of variables, as may occur in the case of closed-loop control schemes.

Besides the Ritz method, other direct methods of the calculus of variations [27] have often proved quite unfit to deal with sets \mathcal{S} of admissible decision functions depending on a large number d of variables. This also occurs in the most recent applications of another classical direct method that exploits those we called LCFBFs, i.e., the Galerkin procedure, which is strictly related to the Ritz method (see, for instance, [86, pp. 258–262]). Indeed, in its application to nonlinear optimal control problems, the curse of dimensionality still remains an open issue (see, for example, [8], where it is used to approximately solve the Hamilton–Jacobi–Bellman equations associated with certain nonlinear optimal control problems in which a state feedback law is sought).

As already stated, we conjecture that it is just switching from LCFBFs to other FSP functions such as the OHL networks (hence, from the Ritz method to the ERIM) that enables us to mitigate the curse of dimensionality. Our belief is supported by the theoretical result stated in Theorem 2.2 and in Theorem 3.12 (to come), which provide a way to extend the approximation capabilities of certain kinds of FSP functions to IDO problems, and by a large amount of experimental findings obtained in various application areas. Concerning the latter results, OHL networks based on ridge and radial constructions have been successfully used to compute approximately optimal decision functions of tens of variables in complex IDO problems.

In the following, we provide some insights into the reason why the Ritz method seems not to have met with a great success in problems where the control function depends on a large number of variables. This may be ascribed to the curse of dimensionality, that is, to the fact that the number of “free” parameters to be optimized grows too rapidly with the number of variables on which the control function depends

in closed-loop structures. The results presented in the remainder of this section support this conjecture. After recalling some concepts from Hilbert spaces, we discuss a family of IDO problems for which the application of the classical Ritz method incurs the curse of dimensionality, whereas the ERIM may avoid it.

3.9.1 Some Concepts from the Theory of Hilbert Spaces

We need to recall some basic concepts from the theory of Hilbert spaces.

Definition 3.8 A linear operator $\mathcal{T} : \mathcal{H} \rightarrow \mathcal{H}$ defined on a real Hilbert space \mathcal{H} is bounded if there exists a constant $\alpha > 0$ such that for any $\gamma \in \mathcal{H}$ one has

$$\|\mathcal{T}(\gamma)\|_{\mathcal{H}} \leq \alpha \|\gamma\|_{\mathcal{H}}.$$

It is compact if it is bounded and the image of every bounded subset of \mathcal{H} has a compact closure in the norm of \mathcal{H} . \triangleleft

Definition 3.9 A bounded linear operator $\mathcal{T} : \mathcal{H} \rightarrow \mathcal{H}$ defined on a real Hilbert space \mathcal{H} is self-adjoint if, for any $\gamma^{(1)}, \gamma^{(2)} \in \mathcal{H}$, one has

$$\langle \gamma^{(1)}, \mathcal{T}(\gamma^{(2)}) \rangle_{\mathcal{H}} = \langle \mathcal{T}(\gamma^{(1)}), \gamma^{(2)} \rangle_{\mathcal{H}}. \quad (3.43)$$

It is positive-definite if $\langle \mathcal{T}(\gamma), \gamma \rangle_{\mathcal{H}} > 0$ for any $\gamma \in \mathcal{H}$ with $\gamma \neq 0$, and positive-semidefinite if $\langle \mathcal{T}(\gamma), \gamma \rangle_{\mathcal{H}} \geq 0$ for any $\gamma \in \mathcal{H}$ with $\gamma \neq 0$. \triangleleft

Definition 3.10 Given a self-adjoint operator $\mathcal{T} : \mathcal{H} \rightarrow \mathcal{H}$, a real number λ is an eigenvalue of \mathcal{T} if there exists a nonzero element $\gamma \in \mathcal{H}$ such that $\mathcal{T}(\gamma) = \lambda\gamma$. Such an element γ is called an eigenfunction associated with the eigenvalue λ . \triangleleft

Likewise in the finite-dimensional case, all the eigenvalues of a self-adjoint linear operator are real, and all the eigenvalues of a self-adjoint and positive-definite (positive-semidefinite, resp.) linear operator are positive (nonnegative, resp.). Therefore, a compact self-adjoint and positive-definite (positive-semidefinite, respectively) linear operator defined on a real Hilbert space is (in informal terms) an extension to the case of a real Hilbert space of the concept of a positive-definite (positive-semidefinite, respectively) symmetric matrix $M \in \mathbb{R}^{d \times d}$ in finite dimension.

3.9.2 The IDO Problem Used for the Comparison

Let \mathcal{H} be a real Hilbert space, $\mathcal{T} : \mathcal{H} \rightarrow \mathcal{H}$ a self-adjoint and positive-definite operator defined on \mathcal{H} , $F : \mathcal{H} \rightarrow \mathbb{R}$ the functional defined, for any $\gamma \in \mathcal{H}$, as

$$F(\gamma) \triangleq \langle T(\gamma), \gamma \rangle_{\mathcal{H}}, \quad (3.44)$$

and

$$\mathcal{S} \triangleq \{\gamma \in \mathcal{H} : \|\gamma\|_{\mathcal{H}} = 1\}.$$

We consider the following class of IDO Problems Q (where “Q” stands for “quadratic”), which are instances of Problem P (we assume the existence and uniqueness of the optimal solution, as usually done in this book), restated in this case as a maximization problem for notational convenience.

Problem Q. *Find the optimal function $\gamma^* \in \mathcal{H}$ that maximizes the functional F over \mathcal{S} , i.e., find*

$$\gamma^* = \arg \max_{\gamma \in \mathcal{S}} F(\gamma). \quad (3.45)$$

▫

Let us consider a finite-dimensional problem that has some similarities with Problem Q. Let $f : \mathbb{R}^d \rightarrow \mathbb{R}$ be a quadratic function defined, for any $\mathbf{x} \in \mathbb{R}^d$, as

$$f(\mathbf{x}) = \mathbf{x}^\top M \mathbf{x}, \quad (3.46)$$

where $M \in \mathbb{R}^{d \times d}$ is a symmetric positive-definite matrix, and let

$$S \triangleq \{\mathbf{x} \in \mathbb{R}^d : |\mathbf{x}| = 1\}.$$

Then, a finite-dimensional version of Problem Q is the following (again, we suppose the existence and uniqueness of the optimal solution):

Problem Q_f . *Find the optimal vector $\mathbf{x}^* \in \mathbb{R}^d$ that maximizes the function f over S , i.e., find*

$$\mathbf{x}^* = \arg \max_{\mathbf{x} \in S} f(\mathbf{x}). \quad (3.47)$$

▫

By replacing the constraint $|\mathbf{x}| = 1$ in Problem Q_f with the equivalent condition $|\mathbf{x}|^2 = 1$, we obtain the Lagrangian function

$$L(\mathbf{x}, \lambda) \triangleq \mathbf{x}^\top M \mathbf{x} + \lambda(|\mathbf{x}|^2 - 1), \quad (3.48)$$

where $\lambda \in \mathbb{R}$. Setting the gradient vector of the Lagrangian function $L(\mathbf{x}, \lambda)$ with respect to \mathbf{x} to zero, we have to solve the following eigenvalue problem, which provides a necessary condition for a candidate optimal solution of Problem Q_f :

$$M\mathbf{x} = \lambda\mathbf{x} \quad (3.49)$$

with $|\mathbf{x}| = 1$.

By solving the characteristic equation $\det(M - \lambda I) = 0$, one gets the eigenvalues $\lambda_1, \dots, \lambda_d$. At least one eigenvector \mathbf{x}_i can be associated to each eigenvalue λ_i (each

eigenvector solves (3.49)). Inserting (3.49) inside (3.46) and using the condition $|\mathbf{x}|^2 = 1$, we have

$$f(\mathbf{x}_i) = \mathbf{x}_i^\top M \mathbf{x}_i = \lambda_i \mathbf{x}_i^\top \mathbf{x}_i, \quad i = 1 \dots, d.$$

Then, $f(\mathbf{x}_i)$ is maximized on S by the largest eigenvalue λ^{\max} . Therefore, the optimal solution \mathbf{x}° to Problem Q_f is given by a unit-norm eigenvector associated with λ^{\max} (the solution \mathbf{x}° is unique – apart from multiplication by the constant -1 – if λ^{\max} has multiplicity 1). Using again the condition $|\mathbf{x}|^2 = 1$, we obtain $f(\mathbf{x}^\circ) = \lambda^{\max}$.

Let us now go back to the IDO Problem Q. We replace the constraint $\|\gamma\|_{\mathcal{H}} = 1$ with the equivalent condition $\|\gamma\|_{\mathcal{H}}^2 = 1$ and we introduce the Lagrangian functional

$$L(\gamma, \lambda) \triangleq \langle T(\gamma), \gamma \rangle_{\mathcal{H}} + \lambda(\|\gamma\|_{\mathcal{H}}^2 - 1), \quad (3.50)$$

where $\lambda \in \mathbb{R}$ (compare with (3.48)). If one searches for the stationary points of $L(\gamma, \lambda)$ via the method of Lagrange multipliers for the infinite-dimensional case⁹ (see, for instance, [109, Chaps. 7–9]), it follows that solving Problem Q is equivalent to solving the eigenvalue problem

$$T(\gamma) = \lambda^{\max} \gamma \quad (3.51)$$

for the largest eigenvalue λ^{\max} of T , with the constraint $\|\gamma\|_{\mathcal{H}} = 1$. Hence, the optimal solution γ° of Problem Q is a unit-norm eigenfunction of T associated with its largest eigenvalue λ^{\max} , which is unique – apart from the multiplication by the constant -1 – when such an eigenvalue has multiplicity 1. Finally, one obtains $F(\gamma^\circ) = \lambda^{\max}$ by inserting (3.51) inside (3.50) and exploiting the fact that γ° has unit norm.

3.9.3 Approximate Solution by the Ritz Method

Let \mathcal{H} be a real Hilbert space of functions depending on d real arguments and the operator T be compact. Since we are interested in the dependence of several objects on d , we use – as in other parts of the book – the superscript d . So, for instance, we write F^d , \mathcal{H}^d , $\gamma^{d\circ}$ instead of merely F , \mathcal{H} , γ° . The Ritz method applied to Problem Q^d consists in restricting the maximization of the functional F^d to a fixed sequence of unit spheres in finite-dimensional subspaces \mathcal{H}_n^d of \mathcal{H}^d with increasing dimension n , where the sequence of subspaces

$$\mathcal{H}_1^d \subset \mathcal{H}_2^d \subset \dots \subset \mathcal{H}_n^d \subset \dots \subset \mathcal{H}^d \quad (3.52)$$

⁹Such an extension basically replaces the gradient in the finite-dimensional case with the Fréchet derivative.

is dense in \mathcal{H}^d . More precisely, for each n , letting

$$\mathcal{S}_n^d \triangleq \mathcal{S}^d \cap \mathcal{H}_n^d,$$

the Ritz method solves for every positive integer n the following finite-dimensional optimization (FDO) problem (again, we assume the existence and uniqueness of the optimal solution).

Problem Q_n^d. *Find the optimal function $\gamma_n^{d\circ} \in \mathcal{H}_n^d$ that maximizes the functional F^d over \mathcal{S}_n^d , i.e., find*

$$\gamma_n^{d\circ} = \arg \max_{\gamma_n^d \in \mathcal{S}_n^d} F^d(\gamma_n^d). \quad (3.53)$$

△

It may be noted that the sets \mathcal{H}_n^d in (3.52) play the same role as the sets \mathcal{A}_n defined in (2.2), when they are represented by LCFBFs (see (3.2)). Similarly, when $\mathcal{H}_n^d = \mathcal{A}_n$, the sets \mathcal{S}_n^d coincide with the ones defined in (2.6).

In the following paragraphs, we restate Problem Q_n^d in a form similar to the one of Problem Q, by replacing in (3.53) the set \mathcal{S}_n^d with \mathcal{S}^d through a suitable modification of the objective functional. In order to do this, denote by \mathcal{P}_n^d the *orthogonal projection operator* of \mathcal{H}^d on \mathcal{H}_n^d , i.e., the linear operator that maps each function $\gamma^d \in \mathcal{H}^d$ to its closest function $\gamma_n^d \in \mathcal{H}_n^d$. Such an operator is self-adjoint. Let us also define the linear operator

$$\mathcal{T}_n^d \triangleq \mathcal{P}_n^d \mathcal{T}^d \mathcal{P}_n^d, \quad (3.54)$$

which is self-adjoint and positive-semidefinite, and the functional

$$\begin{aligned} F_n^d(\gamma^d) &\triangleq \langle \mathcal{T}^d(\mathcal{P}_n^d(\gamma^d)), (\mathcal{P}_n^d(\gamma^d)) \rangle_{\mathcal{H}^d} = \langle (\mathcal{P}_n^d \mathcal{T}^d \mathcal{P}_n^d)(\gamma^d), \gamma^d \rangle_{\mathcal{H}^d}, \\ &= \langle \mathcal{T}_n^d(\gamma^d), \gamma^d \rangle_{\mathcal{H}^d}, \end{aligned} \quad (3.55)$$

where the first equality in (3.55) follows from the self-adjointness of \mathcal{P}_n^d . Then, it can be shown [48] that Problem Q_n^d is equivalent to the following one.

Problem Q_n^{d'}. *Find the optimal function $\gamma_n^{d\circ} \in \mathcal{H}^d$ that maximizes the functional F_n^d over \mathcal{S}^d , i.e., find*

$$\gamma_n^{d\circ} = \arg \max_{\gamma^d \in \mathcal{S}^d} F_n^d(\gamma^d). \quad (3.56)$$

△

Notice that, in Problem Q_n^{d'}, the function γ^d belongs to \mathcal{S}^d . Hence, we do not have the subscript n . Note also that the optimal function $\gamma_n^{d\circ}$ belongs to $\mathcal{S}_n^d \subset \mathcal{S}^d$ (then the subscript n remains). Therefore, the formulation of Problem Q_n^{d'} turns out to be similar to that of Problem Q, differing only in the objective functional, but not in the constraints.

The reason why Problem $Q_n^{d'}$ has been introduced is that we can apply arguments similar to those associated with Problem Q. This means that solving Problem $Q_n^{d'}$ is equivalent to solving the eigenvalue problem

$$\mathcal{T}_n^d(\gamma_n^d) = \lambda_{nd}^{\max} \gamma_n^d \quad (3.57)$$

for the largest eigenvalue λ_{nd}^{\max} of \mathcal{T}_n^d and with $\|\gamma_n^d\|_{\mathcal{H}^d} = 1$. Therefore, $\gamma_n^{d\circ}$ is a unit-norm eigenfunction of \mathcal{T}_n^d associated with its largest eigenvalue λ_{nd}^{\max} . Finally, by replacing (3.57) inside (3.55), and exploiting the fact that $\gamma_n^{d\circ}$ has unit norm, we get

$$F_n^d(\gamma_n^{d\circ}) = F^d(\gamma_n^{d\circ}) = \lambda_{nd}^{\max}.$$

Of course, by the definitions we have

$$F^d(\gamma^{d\circ}) \geq F_n^d(\gamma_n^{d\circ}) = F^d(\gamma_n^{d\circ}).$$

Finally, since $F^d(\gamma^{d\circ}) = \lambda_d^{\max}$ and $F_n^d(\gamma_n^{d\circ}) = \lambda_{nd}^{\max}$, we obtain $\lambda_d^{\max} \geq \lambda_{nd}^{\max}$.

3.9.4 The Curse of Dimensionality in the Ritz Method and the Possibility of Avoiding it in the ERIM

We are interested in the relationships between the accuracy of approximation of the optimal objective value of Problem Q^d , the dimension d , and the minimum model complexity n required to guarantee such an accuracy, for both LCFBFs (used in the Ritz method) and suitable FSP functions (used in the ERIM). The “suitable” FSP functions belong to sequences of sets that are called “adaptive” sequences in [48]. Such sequences must be provided with the same nestedness and density properties as the ones used in the Ritz method. We have seen that families of OHL networks enjoy these properties. In the following, for $n = 1, \dots, \infty$ we denote the elements of these “adaptive” sequences by $\mathcal{H}_n^{d,\text{ERIM}}$ (see [48] for their precise definition). We assume that $\mathcal{H}^d = \mathcal{L}_2((0, 1)^d)$.

We measure the performance of the Ritz method and of the ERIM by the *relative optimization error*

$$\varepsilon_n^{F^d}(\gamma^{d\circ}, \gamma_n^{d\circ}) \triangleq \frac{|F^d(\gamma^{d\circ}) - F^d(\gamma_n^{d\circ})|}{|F^d(\gamma^{d\circ})|}. \quad (3.58)$$

For the Ritz method, the error (3.58) takes on the form $\frac{\lambda_d^{\max} - \lambda_{nd}^{\max}}{\lambda_d^{\max}}$.

We say that a sequence of Problems Q^d , indexed by d , incurs the curse of dimensionality if, for a fixed $r > 0$, we have

$$\varepsilon_n^{F^d}(\gamma^{d\circ}, \gamma_n^{d\circ}) = \Omega\left(\frac{1}{n^{r/d}}\right). \quad (3.59)$$

For both the Ritz method and the ERIM, the computation of Error (3.58) is quite cumbersome. For brevity, we only give the conclusive results of Theorems 4.2 and 5.1 reported in [48], omitting the assumptions, which are technical but quite general.

Theorem 4.2 in [48] states that, for any choice of fixed-basis functions (indexed by both n and d), there exists a sequence of Problems Q^d for which Estimate (3.59) holds. Then, the Ritz method incurs the curse of dimensionality.

Theorem 5.2 in [48] states that

$$\varepsilon_n^{F^d, \text{ERIM}}(\gamma^{d\circ}, \gamma_n^{d\circ}) = O\left(\frac{1}{n^r}\right). \quad (3.60)$$

Since in (3.60) the “big O” notation hides a factor that may depend on the dimension d , this result allows us to conclude that the ERIM avoids the curse of dimensionality only when such a factor does not depend on d or when it does not grow “too fast” with d . Among the hypotheses of [48, Theorem 5.2], the following one plays a central role: $\{\mathcal{H}_n^{d, \text{ERIM}}\}_{n=1}^\infty$ are sequences of nested “adaptively chosen” n -dimensional subspaces of \mathcal{H}^d , suitably constructed (as specified in [48]) according to the particular instance of Problem Q^d to be solved. This is verified, e.g., by OHL neural networks with suitable computational units.

3.10 Rates of Optimization by the ERIM

Inspired by Theorem 3.3 and its formulation in terms of the \mathcal{G} -variation norm (see Theorem 3.11), in [98] the approximation rate (3.23) was extended to the convex IDO Problem P^d . In particular, conditions were investigated therein, under which the minimum number of basis functions that guarantees a desired accuracy of suboptimal solutions obtained using OHL networks exploited by the ERIM grows at most linearly with the number d of variables in admissible solutions. The critical term in the estimates from [98] is of the form $n^{-1/2}$, multiplied by the \mathcal{G} -variation norm of the optimal solution. Hence, such a norm plays a basic role in estimating not only the accuracy of approximation but also the accuracy of approximate optimization. Roughly speaking, by using OHL networks with respect to which the \mathcal{G} -variation norms of the admissible solutions are “small”, one can obtain suboptimal solutions that are good approximations of the optimal one and are made of a small number of basis functions; hence, they require the optimization of a small number of parameters.

To simplify the notation, we express the problem $\inf_{\gamma \in \mathcal{S}} F(\gamma)$ of minimizing the functional F over a subset \mathcal{S} of a Hilbert space \mathcal{H} as

$$(\mathcal{S}, F).$$

Recall that a sequence $\{\gamma_n\}$ of elements of \mathcal{S} is called *F-minimizing over \mathcal{S}* or a *minimizing sequence for F over \mathcal{S}* if $\lim_{n \rightarrow \infty} F(\gamma_n) = \inf_{\gamma \in \mathcal{S}} F(\gamma)$. By the definition of infimum, for any problem (\mathcal{S}, F) there exists a minimizing sequence. We denote by

$$\operatorname{argmin}(\mathcal{S}, F) \triangleq \{\gamma^\circ \in \mathcal{S} : F(\gamma^\circ) = \inf_{\gamma \in \mathcal{S}} F(\gamma)\}$$

the set of *minimum points* of the problem (\mathcal{S}, F) and, for $\varepsilon > 0$, by

$$\operatorname{argmin}_\varepsilon(\mathcal{S}, F) \triangleq \{\gamma^\varepsilon \in \mathcal{S} : F(\gamma^\varepsilon) < \inf_{\gamma \in \mathcal{S}} F(\gamma) + \varepsilon\}$$

the set of its ε -near minimum points.

Finally, we recall the two following definitions of *modulus of continuity* and *modulus of convexity* of a functional defined on a subset of a Hilbert space.

Definition 3.11 Let $F : \mathcal{S} \subseteq \mathcal{H} \rightarrow \mathbb{R}$ be a continuous functional defined on a subset \mathcal{S} of a Hilbert space \mathcal{H} , and $\bar{\gamma} \in \mathcal{S}$. Then, the modulus of continuity of F at $\bar{\gamma} \in \mathcal{S}$ is defined as the function $\alpha : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ such that

$$\alpha(t) = \sup\{|F(\gamma) - F(\bar{\gamma})| : \gamma \in \mathcal{S} \text{ and } \|\gamma - \bar{\gamma}\|_{\mathcal{H}} \leq t\}.$$

△

Definition 3.12 A functional $F : \mathcal{S} \subseteq \mathcal{H} \rightarrow \mathbb{R}$ defined on a subset \mathcal{S} of a Hilbert space \mathcal{H} is called uniformly convex if there exists a nonnegative function $\delta : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ such that $\delta(0) = 0$, $\delta(t) > 0$ for all $t > 0$, and for all $\gamma_1, \gamma_2 \in \mathcal{S}$ and all $\lambda \in [0, 1]$, one has

$$F(\lambda\gamma_1 + (1 - \lambda)\gamma_2) \leq \lambda F(\gamma_1) + (1 - \lambda)F(\gamma_2) - \lambda(1 - \lambda)\delta(\|\gamma_1 - \gamma_2\|_{\mathcal{H}}).$$

Any such function δ is called a modulus of convexity of F .

△

It is worth mentioning that a typical case of modulus of continuity is a nonnegative function α satisfying $\alpha(t) \leq L_F t$ (when F is Lipschitz-continuous with Lipschitz constant L_F). Moreover, a possible modulus of convexity is the quadratic function $\delta(t) = \frac{1}{2}c_F t^2$ (for some positive constant c_F).

The next theorem is a corollary of [98, Theorem 4.2] for the case of OHL networks. The theorem requires the existence of a minimum point for the IDO problem (\mathcal{H}, F) . This is guaranteed for various convex IDO problems [28, 37, 107, 129]. Moreover, many optimization problems that do not have minimum points can be transformed by regularization into convex IDO problems with minimum points [36, p. 29]. Therefore, the proposition applies to a wide class of IDO problems. We refer the reader to [47, Sect. 3] for a simplified version of Theorem 3.12, still based on the concepts of modulus of continuity and modulus of convexity.

Theorem 3.12 Let \mathcal{H} be a Hilbert space, $D \subseteq \mathbb{R}^p$, $\Omega \subseteq \mathbb{R}^d$, $\varphi : \mathbb{R}^d \times \mathbb{R}^k \rightarrow \mathbb{R}$ a function such that $\mathcal{G}_\varphi \triangleq \{\varphi(\cdot, \kappa) : \kappa \in W \subseteq \mathbb{R}^k\}$ is a bounded subset of

\mathcal{H} , and $s_\varphi \triangleq \sup_{g \in \mathcal{G}_\varphi} \|g\|_{\mathcal{H}}$. Let $F : \mathcal{H} \rightarrow (-\infty, +\infty]$ be a functional, $\gamma^\circ \in \operatorname{argmin}(\mathcal{H}, F)$, F continuous at γ° with a modulus of continuity α , $b \triangleq \|\gamma^\circ\|_{\mathcal{G}_\varphi \text{ var}}$, $\{\varepsilon_n\}_{n=1}^\infty$ a sequence of positive reals, and $\{\gamma_n\}_{n=1}^\infty$ such that $\gamma_n \in \operatorname{argmin}_{\varepsilon_n} (\operatorname{span}_n \mathcal{G}_\varphi, F)$. Thus, for every positive integer n the following estimates hold:

- (i) $\inf_{\gamma_n \in \operatorname{span}_n \mathcal{G}_\varphi} F(\gamma_n) - F(\gamma^\circ) \leq \alpha \left(s_\varphi \frac{b}{\sqrt{n}} \right);$
- (ii) if $b < \infty$ and $\lim_{n \rightarrow \infty} \varepsilon_n = 0$, then $\{\gamma_n\}$ is a F -minimizing sequence over \mathcal{H} and $F(\gamma_n) - F(\gamma^\circ) \leq \alpha \left(s_\varphi \frac{b}{\sqrt{n}} \right) + \varepsilon_n;$
- (iii) if F is uniformly convex on \mathcal{H} with a modulus of convexity δ , then $\delta(\|\gamma_n - \gamma^\circ\|_{\mathcal{H}}) \leq \alpha \left(s_\varphi \frac{b}{\sqrt{n}} \right) + \varepsilon_n.$ \square

The key point of the proof of Theorem 3.12 is to use the modulus of continuity of the functional at γ° and its modulus of convexity, which make it possible to “transform” the approximation rate provided by Theorem 3.3 into a rate of approximate optimization. In more detail, the proof of part (i) is based on Theorem 3.11. Indeed, under the assumptions of that theorem, one knows that there exists an approximation $\hat{\gamma}_n$ of γ° in $\operatorname{span}_n \mathcal{G}_\varphi$ such that $\|\gamma^\circ - \hat{\gamma}_n\|_{\mathcal{H}} \leq s_\varphi \frac{b}{\sqrt{n}}$. Then, by the optimality of γ° and the definition of modulus of continuity, one obtains

$$\inf_{\gamma \in \operatorname{span}_n \mathcal{G}_\varphi} F(\gamma) - F(\gamma^\circ) \leq F(\hat{\gamma}_n) - F(\gamma^\circ) \leq \alpha(\|\gamma^\circ - \hat{\gamma}_n\|_{\mathcal{H}}) \leq \alpha \left(s_\varphi \frac{b}{\sqrt{n}} \right),$$

which is the statement (i). Part (ii) is derived from part (i) and shows how one can construct an F -minimizing sequence over \mathcal{H} . Finally, part (iii) follows from the definition of the modulus of convexity and the property that $\gamma_n \in \operatorname{argmin}_{\varepsilon_n} (\operatorname{span}_n \mathcal{G}_\varphi, F)$. Its importance lies in the fact that, once an expression of the modulus of convexity δ is known, it allows one to find an upper bound on $\|\gamma^\circ - \gamma_n\|_{\mathcal{H}}$ for any $\gamma_n \in \operatorname{argmin}_{\varepsilon_n} (\operatorname{span}_n \mathcal{G}_\varphi, F)$. Indeed, supposing for simplicity that $\delta(\cdot)$ is invertible with inverse $\delta^{-1}(\cdot)$, one obtains from (iii)

$$\|\gamma_n - \gamma^\circ\|_{\mathcal{H}} \leq \delta^{-1} \left(\alpha \left(s_\varphi \frac{b}{\sqrt{n}} \right) + \varepsilon_n \right).$$

The upper bounds in Theorem 3.12 exhibit the following interesting feature: the denominator is $n^{1/2}$, for every number d of variables. However, since the term $n^{-1/2}$ is multiplied by the \mathcal{G}_φ -variation norm $\|\gamma^\circ\|_{\mathcal{G}_\varphi \text{ var}}$ of the minimum point, to keep the upper bounds small it is necessary to keep such a norm small, also for a growing number d of variables of admissible solutions. This can be done by imposing a certain type of regularity on the set of admissible solutions, which increases with the dimension d , as discussed in Sect. 3.8.3. In this way, Theorem 3.12 may allow one to obtain bounds on the accuracy of suboptimal solutions expressed by OHL networks which grow “not too fast” with the number d of variables in admissible solutions [98].

Finally, it is worth mentioning that, combining a result similar to Theorem 3.12 with a lower bound on the worst-case approximation error by LCFBFs similar to (3.33), a lower bound on the accuracy of approximate optimization using LCFBFs was obtained in [47, Sect. 3]. The idea exploited in [47] is to consider a convex IDO problem whose solution is one function known to be “difficult to approximate” by LCFBFs, then to use the modulus of continuity and the modulus of convexity of the functional to “transform” the error bound on the approximation of that function into a bound on the error of approximate optimization. Unlike the bound obtained from the estimates in Theorem 3.12, this second bound incurs the curse of dimensionality.

References

1. Adams RA (1975) Sobolev spaces. Academic Press
2. Adams RA, Fournier JJF (2003) Sobolev spaces. Academic Press
3. Alt W (1984) On the approximation of infinite optimization problems with an application to optimal control problems. *Appl Math Optim* 12:15–27
4. Ba LJ, Caruana R (2014) Do deep networks really need to be deep? In: Ghahramani Z, Welling M, Cortes C, Lawrence ND, Weinberger KQ (eds) Advances in neural information processing systems, vol 27, pp 1–9
5. Barron AR (1992) Neural net approximation. In: Narendra KS (ed) Proceedings of the 7th Yale workshop on adaptive and learning systems. Yale University Press, pp 69–72
6. Barron AR (1993) Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Trans Inf Theory* 39:930–945
7. Barron AR, Klusowski JM (2018) Approximation and estimation for high-dimensional deep learning networks. Technical report [arXiv:1809.03090v2](https://arxiv.org/abs/1809.03090v2)
8. Beard RW, McLain TW (1998) Successive Galerkin approximation algorithms for nonlinear optimal and robust control. *Int J Control* 71:717–743
9. Bengio Y (2009) Learning deep architectures for AI. *Found Trends Mach Learn* 2:1–127
10. Bengio Y, Delalleau O, Le Roux N (2005) The curse of dimensionality for local kernel machines. Technical Report 1258, Département d’Informatique et Recherche Opérationnelle, Université de Montréal
11. Bengio Y, Delalleau O, Le Roux N (2006) The curse of highly variable functions for local kernel machines. In: Advances in neural information processing systems, vol 18. MIT Press, pp 107–114
12. Bengio Y, LeCun Y (2007) Scaling learning algorithms towards AI. In: Bottou L, Chapelle O, DeCoste D, Weston J (eds) Large-scale kernel machines. MIT Press
13. Bianchini M, Scarselli F (2014) On the complexity of neural network classifiers: a comparison between shallow and deep architectures. *IEEE Trans Neural Netw Learn Syst* 25:1553–1565
14. Blum EK, Li LK (1991) Approximation theory and feedforward networks. *Neural Netw* 4:511–515
15. Bosarge WE Jr, Johnson OG, McKnight RS, Timlake WP (1973) The Ritz-Galerkin procedure for nonlinear control problems. *SIAM J Numer Anal* 10:94–111
16. Breiman L (1993) Hinging hyperplanes for regression, classification, and function approximation. *IEEE Trans Inf Theory* 39:993–1013
17. Brezis H (2011) Functional analysis. Sobolev spaces and partial differential equations. Springer
18. Carroll SM, Dickinson BW (1989) Construction of neural nets using the Radon transform. In: Proceedings of the international joint conference on neural networks, pp 607–611
19. Cervellera C, Macciò D (2013) Learning with kernel smoothing models and low-discrepancy sampling. *IEEE Trans Neural Netw Learn Syst* 24:504–509

20. Cervellera C, Macciò D (2014) Local linear regression for function learning: an analysis based on sample discrepancy. *IEEE Trans Neural Netw Learn Syst* 25:2086–2098
21. Chen T, Chen H (1995) Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and application to dynamical systems. *IEEE Trans Neural Netw* 6:911–917
22. Chen T, Chen H, Liu R (1995) Approximation capability in $C(\bar{\mathbb{R}}^n)$ by multilayer feedforward networks and related problems. *IEEE Trans Neural Netw* 6:25–30
23. Chui CK, Mhaskar HN (2018) Deep nets for local manifold learning. *Front Appl Math Stat* 4, Article 12
24. Courant R (1948) Differential and integral calculus, vol II. Interscience Publishers, Inc
25. Courant R, Hilbert D (1962) Methods of mathematical physics, vol II. Interscience Publishers, Inc
26. Cybenko G (1989) Approximation by superpositions of a sigmoidal function. *Math Control Signals Syst* 2:303–314
27. Dacorogna B (2008) Direct methods in the calculus of variations, 2nd edn. Springer
28. Daniel JW (1971) The approximate minimization of functionals. Prentice Hall
29. Daniel JW (1973) The Ritz-Galerkin method for abstract optimal control problems. *SIAM J Control* 11:53–63
30. Darken C, Donahue M, Gurvits L, Sontag E (1993) Rate of approximation results motivated by robust neural network learning. In: Proceedings of the sixth annual ACM conference on computational learning theory. ACM, pp 303–309
31. de Villiers J, Barnard E (1992) Backpropagation neural nets with one and two hidden layers. *IEEE Trans Neural Netw* 3:136–141
32. DeVore RA, Howard R, Micchelli C (1989) Optimal nonlinear approximation. *Manuscr Math* 63:469–478
33. Donahue M, Gurvits L, Darken C, Sontag E (1997) Rates of convex approximation in non-Hilbert spaces. *Constr Approx* 13:187–220
34. Donoho DL, Johnstone IM (1989) Projection-based approximation and a duality method with kernel methods. *Ann Stat* 17:58–106
35. Dontchev AL (1996) An a priori estimate for discrete approximations in nonlinear optimal control. *SIAM J Control Optim* 34:1315–1328
36. Dontchev AL, Zolezzi T (1993) Well-posed optimization problems. Lecture notes in mathematics, vol 1543. Springer
37. Ekeland I, Temam R (1976) Convex analysis and variational problems. North-Holland Publishing Company and American Elsevier
38. Felgenhauer U (1999) On Ritz type discretizations for optimal control problems. In: Proceedings of the 18th IFIP-ICZ conference. Chapman-Hall, pp 91–99
39. Friedman JH, Stuetzle W (1981) Projection pursuit regression. *J Am Stat Assoc* 76:817–823
40. Funahashi K (1989) On the approximate realization of continuous mappings by neural networks. *Neural Netw* 2:183–192
41. Girosi F (1994) Regularization theory, Radial Basis Functions and networks. In: Cherkassky V, Friedman JH, Wechsler H (eds) From statistics to neural networks. Theory and pattern recognition applications, Computer and systems sciences, Subseries F. Springer
42. Girosi F (1995) Approximating error bounds that use VC bounds. In: Proceedings of the international conference on artificial neural networks, pp 295–302
43. Girosi F, Anzellotti G (1992) Rates of convergence of approximation by translates. Technical Report 1288, Artificial Intelligence Laboratory, Massachusetts Institute of Technology
44. Girosi F, Anzellotti G (1993) Rates of convergence for Radial Basis Functions and neural networks. In: Mammone RJ (ed) Artificial neural networks for speech and vision. Chapman & Hall, pp 97–113
45. Girosi F, Jones M, Poggio T (1995) Regularization theory and neural networks architectures. *Neural Comput* 7:219–269
46. Giulini S, Sanguineti M (2009) Approximation schemes for functional optimization problems. *J Optim Theory Appl* 140:33–54

47. Gnecco G (2012) A comparison between fixed-basis and variable-basis schemes for function approximation and functional optimization. *J Appl Math* 2012:1–17
48. Gnecco G (2016) On the curse of dimensionality in the Ritz method. *J Optim Theory Appl* 168:488–509
49. Gnecco G, Gori M, Melacci S, Sanguineti M (2014) A theoretical framework for supervised learning from regions. *Neurocomputing* 129:25–32
50. Gnecco G, Gori M, Melacci S, Sanguineti M (2015) Foundations of support constraint machines. *Neural Comput* 27:388–480
51. Gnecco G, Gori M, Melacci S, Sanguineti M (2015) Learning with mixed hard/soft pointwise constraints. *IEEE Trans Neural Netw Learn Syst* 26:2019–2032
52. Gnecco G, Gori M, Sanguineti M (2012) Learning with boundary conditions. *Neural Comput* 25:1029–1106
53. Gnecco G, Kůrková V, Sanguineti M (2011) Can dictionary-based computational models outperform the best linear ones? *Neural Netw* 24:881–887
54. Gnecco G, Kůrková V, Sanguineti M (2011) Some comparisons of complexity in dictionary-based and linear computational models. *Neural Netw* 24:172–182
55. Gnecco G, Sanguineti M (2008) Estimates of the approximation error using Rademacher complexity: learning vector-valued functions. *J Inequalities Appl* 2008:1–16
56. Gnecco G, Sanguineti M (2010) Estimates of variation with respect to a set and applications to optimization problems. *J Optim Theory Appl* 145:53–75
57. Gnecco G, Sanguineti M (2010) Suboptimal solutions to dynamic optimization problems via approximations of the policy functions. *J Optim Theory Appl* 146:764–794
58. Gnecco G, Sanguineti M (2011) On a variational norm tailored to variable-basis approximation schemes. *IEEE Trans Inf Theory* 57:549–558
59. Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press
60. Gurvits L, Koiran P (1997) Approximation and learning of convex superpositions. *J Comput Syst Sci* 55:161–170
61. Hager WW (1975) The Ritz-Trefftz method for state and control constrained optimal control problems. *SIAM J Numer Anal* 12:854–867
62. Hastie T, Tibshirani R, Friedman J (2008) The elements of statistical learning. Springer
63. Haykin S (2008) Neural networks and learning systems. Pearson Prentice-Hall
64. Hecht-Nielsen R (1989) Theory of the backpropagation neural network. In: Proceedings of the international joint conference on neural networks, pp 593–605
65. Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. *Neural Comput* 18:1527–1554
66. Hinton GH (2007) Learning multiple layers of representation. *Trends Cogn Sci* 11:428–434
67. Hlaváčková-Schindler K, Sanguineti M (2003) Bounds on the complexity of neural-network models and comparison with linear methods. *Int J Adapt Control Signal Process* 17:179–194
68. Hornik K (1991) Approximation capabilities of multilayer feedforward networks. *Neural Netw* 4:251–257
69. Hornik K (1991) Functional approximation and learning in artificial neural networks. *Neural Netw World* 5:257–266
70. Hornik K (1993) Some new results on neural network approximation. *Neural Netw* 6:1069–1072
71. Hornik K, Stinchcombe M, White H (1989) Multilayer feedforward networks are universal approximators. *Neural Netw* 2:359–366
72. Irie B, Miyake S (1988) Capability of three-layered perceptrons. In: Proceedings of the international joint conference on neural networks, pp 641–648
73. Ito Y (1991) Approximation of functions on a compact set by finite sums of a sigmoid function without scaling. *Neural Netw* 4:817–826
74. Jackson D (2004) Fourier series and orthogonal polynomials. Dover
75. John F (1955) Plane waves and spherical means applied to partial differential equations. Interscience Publishers, Inc

76. Jones LK (1990) Constructive approximation for neural networks by sigmoid functions. *Proc IEEE* 78:1586–1589
77. Jones LK (1992) A simple lemma on greedy approximation in Hilbert space and convergence rates for projection pursuit regression and neural network training. *Ann Stat* 20:608–613
78. Kainen P, Kůrková V, Sanguineti M (2003) Minimization of error functionals over variable-basis functions. *SIAM J Optim* 14:732–742
79. Kainen PC (1997) Utilizing geometric anomalies of high dimension: when complexity makes computation easier. In: Warwick K, Karni M (eds) *Compute-intensive methods in control and signal processing. The curse of dimensionality*, Birkhäuser, pp 283–294
80. Kainen PC, Kůrková V (2009) An integral upper bound for neural network approximation. *Neural Comput* 21:2970–2989
81. Kainen PC, Kůrková V, Sanguineti M (2009) Complexity of Gaussian radial basis networks approximating smooth functions. *J Complex* 25:63–74
82. Kainen PC, Kůrková V, Sanguineti M (2012) Dependence of computational models on input dimension: tractability of approximation and optimization tasks. *IEEE Trans Inf Theory* 58:1203–1214
83. Kainen PC, Kůrková V, Vogt A (1999) Approximation by neural networks is not continuous. *Neurocomputing* 29:47–56
84. Kainen PC, Kůrková V, Vogt A (2000) Geometry and topology of continuous best and near best approximations. *J Approx Theory* 105:252–262
85. Kainen PC, Kůrková V, Vogt A (2001) Continuity of approximation by neural networks in L_p -spaces. *Ann Oper Res* 101:143–147
86. Kantorovich LV, Krylov VI (1958) *Approximate methods of higher analysis*. P. Noordhoff Ltd., Groningen
87. Kolmogorov AN (1991) On the best approximation of functions of a given class. In: Tikhomirov VM (ed) *Selected works of A. N. Kolmogorov*. Kluwer, pp 202–205
88. Kolmogorov AN, Fomin SV (1975) *Introductory real analysis*. Dover Publications Inc
89. Kůrková V (1997) Dimension-independent rates of approximation by neural networks. In: Warwick K, Kárný M (eds) *Computer-intensive methods in control and signal processing. The curse of dimensionality*, Birkhäuser, pp 261–270
90. Kůrková V (1998) Incremental approximation by neural networks. In Warwick K, Kárný M, Kůrková V (eds) *Complexity: neural network approach*. Springer, pp 177–188
91. Kůrková V (2003) High-dimensional approximation by neural networks. In: Suykens J et al (eds) *Advances in learning theory: methods, models, and applications (NATO Science Series III: Computer & Systems Sciences, vol 190)* (Chap 4). IOS Press, pp 69–88
92. Kůrková V (2008) Minimization of error functionals over perceptron networks. *Neural Comput* 20:252–270
93. Kůrková V (2009) Model complexity of neural networks and integral transforms. In: Polycarpou M, Panayiotou C, Alippi C, Ellinas G (eds) *Proceedings of the 2009 international conference on artificial neural networks. Lecture notes in computer science, vol 5768*. Springer, pp 708–718
94. Kůrková V (2012) Complexity estimates based on integral transforms induced by computational units. *Neural Netw* 33:160–167
95. Kůrková V, Kainen PC, Kreinovich V (1997) Estimates of the number of hidden units and variation with respect to half-spaces. *Neural Netw* 10:1061–1068
96. Kůrková V, Sanguineti M (2001) Bounds on rates of variable-basis and neural-network approximation. *IEEE Trans Inf Theory* 47:2659–2665
97. Kůrková V, Sanguineti M (2002) Comparison of worst case errors in linear and neural network approximation. *IEEE Trans Inf Theory* 48:264–275
98. Kůrková V, Sanguineti M (2005) Error estimates for approximate optimization by the extended Ritz method. *SIAM J Optim* 15:461–487
99. Kůrková V, Sanguineti M (2007) Estimates of covering numbers of convex sets with slowly decaying orthogonal subsets. *Discret Appl Math* 155:1930–1942

100. Kůrková V, Sanguineti M (2008) Geometric upper bounds on rates of variable-basis approximation. *IEEE Trans Inf Theory* 54:5681–5688
101. Kůrková V, Sanguineti M (2016) Model complexities of shallow networks representing highly-varying functions. *Neurocomputing* 171:598–604
102. Kůrková V, Sanguineti M (2017) Probabilistic lower bounds for approximation by shallow perceptron networks. *Neural Netw* 91:34–41
103. Kůrková V, Sanguineti M (2019) Classification by sparse neural networks. *IEEE Trans Neural Netw Learn Syst* 30(9):2746–2754
104. Kůrková V, Savický P, Hlaváčková K (1998) Representations and rates of approximation of real-valued Boolean functions by neural networks. *Neural Netw* 11:651–659
105. Lavretsky E (2002) On the geometric convergence of neural approximations. *IEEE Trans Neural Netw* 13:274–282
106. Leshno M, Ya V, Pinkus A, Schocken S (1993) Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Netw* 6:861–867
107. Levitin ES, Polyak BT (1966) Convergence of minimizing sequences in conditional extremum problems. *Dokl Akad Nauk SSSR* 168:764–767
108. Logan BF, Shepp LA (1975) Optimal reconstruction of a function from its projections. *Duke Math J* 42:645–659
109. Luenberger DG (1969) Optimization by vector space methods. Wiley
110. Maiorov V (1999) On best approximation by ridge functions. *J Approx Theory* 99:68–94
111. Maiorov V, Pinkus A (1999) Lower bounds for approximation by MLP neural networks. *Neurocomputing* 25:81–91
112. Maiorov VE, Meir R (2000) On the near optimality of the stochastic approximation of smooth functions by neural networks. *Adv Comput Math* 13:79–103
113. Makovoz Y (1998) Uniform approximation by neural networks. *J Approx Theory* 95:215–228
114. Malanowski K, Buskens C, Maurer H (1997) Convergence of approximations to nonlinear control problems. In: Fiacco AV (ed) Mathematical programming with data perturbation. Lecture notes in pure and applied mathematics, vol 195. Marcel Dekker, pp 253–284
115. Mhaskar H, Liao Q, Poggio T (2016) Learning functions: when is deep better than shallow. CBMM Memo No. 045. <https://arxiv.org/pdf/1603.00988v4.pdf>. Accessed 31 May 2016
116. Mhaskar H, Liao Q, Poggio T (2016) Learning real and Boolean functions: when is deep better than shallow. CBMM Memo No. 45. <https://arxiv.org/pdf/1603.00988v1.pdf>. Accessed 4 Mar 2016
117. Mhaskar HN (1995) Versatile Gaussian networks. In: Proceedings of the IEEE workshop on nonlinear signal and image processing, pp 70–73
118. Mhaskar HN, Micchelli CA (1992) Approximation by superposition of a sigmoidal function and radial basis functions. *Adv Appl Math* 13:350–373
119. Mhaskar HN, Poggio T (2016) Deep vs. shallow networks: an approximation theory perspective. *Anal Appl* 14:829–848
120. Mikhlin SG (1980) The approximate solution of one-sided variational problems. *Izvestija Vysšsiih Učebnyih Zavedenij Matematika* 213(2):45–48
121. Minsky M, Papert S (1969) Perceptrons. MIT Press
122. Mussa-Ivaldi FA (1992) From basis functions to basis fields: vector field approximation from sparse data. *Biol Cybern* 67:479–489
123. Mussa-Ivaldi FA, Gandolfo F (1993) Networks that approximate vector-valued mappings. In: Proceedings of the IEEE international conference on neural networks, pp 1973–1978
124. Park J, Sandberg IW (1991) Universal approximation using radial-basis-function networks. *Neural Comput* 3:246–257
125. Pinkus A (1985) *n-widths in approximation theory*. Springer
126. Pinkus A (1997) Approximation by ridge functions. In: Le Méhauté A, Rabut C, Schumaker LL (eds) *Surface fitting and multiresolution methods*. Vanderbilt University Press, pp 1–14
127. Pinkus A (1999) Approximation theory of the MLP model in neural networks. *Acta Numer* 8:143–195

128. Pisier G (1981) Remarques sur un résultat non publié de B. Maurey. In: Séminaire d'Analyse Fonctionnelle 1980–81, vol I, no 12. École Polytechnique, Centre de Mathématiques, Palaiseau
129. Polyak BT (1966) Existence theorems and convergence of minimizing sequences in extremum problems with restrictions. Dokl Akad Nauk SSSR 166:72–75
130. Ritz W (1909) Über eine neue Methode zur Lösung gewisser Variationsprobleme der mathematischen Physik. Journal für die Reine und Angewandte Mathematik 135:1–61
131. Rosenblatt F (1958) The perceptron: a probabilistic model for information storage and organization of the brain. Psychol Rev 65:386–408
132. Rosenblatt F (Feb 1960) On the convergence of reinforcement procedures in simple perceptrons. Technical Report Report VG-1196-G-4, Cornell Aeronautical Laboratory, Buffalo, NY
133. Rudin W (1964) Principles of mathematical analysis. McGraw-Hill
134. Sanguineti M (2008) Universal approximation by ridge computational models and neural networks: a survey. Open Appl Math J 2:31–58
135. Scarselli F, Tsoi AC (1998) Universal approximation using feedforward neural networks: a survey of some existing methods, and some new results. Neural Netw 11:15–37
136. Schölkopf B, Smola AJ (2001) Learning with kernels. MIT Press
137. Singer I (1970) Best approximation in normed linear spaces by elements of linear subspaces. Springer
138. Sirisena HR, Chou FS (1979) Convergence of the control parametrization Ritz method for nonlinear optimal control problems. J Optim Theory Appl 29:369–382
139. Sjöberg J, Zhang Q, Ljung L, Benveniste A, Gorenne P-Y, Delyon B, Hjalmarsson H, Juditsky A (1995) Nonlinear black-box modeling in system identification: a unified overview. Automatica 31:1691–1724
140. Sontag ED (1992) Feedback stabilization using two-hidden-layer nets. IEEE Trans Neural Netw 3:981–990
141. Stinchcombe M, White H (1989) Universal approximation using feedforward networks with non-sigmoid hidden layer activation functions. In: Proceedings of the international joint conference on neural networks, vol 1. SOS Printing, San Diego, pp 613–617. (Reprinted in Artificial neural networks: approximation & learning theory, White H (ed) Blackwell, 1992)
142. Tjuhtin VB (1982) An error estimate for approximate solutions in one-sided variational problems. Vestn Leningr Univ Math 14:247–254
143. Vapnik VN (1998) Statistical learning theory. Wiley
144. Wasilkowski GW, Woźniakowski H (2001) Complexity of weighted approximation over \mathbb{R}^d . J Complex 17:722–740
145. Widrow B, Hoff Jr ME (1960) Adaptive switching circuits. In: 1960 IRE western electric show and convention record, Part 4, pp 96–104
146. Widrow B, Lehr MA (1990) 30 years of adaptive neural networks: perceptron, madaline, and backpropagation. Proc IEEE 78:1415–1442

Chapter 4

Design of Mathematical Models by Learning From Data and FSP Functions



The problem of constructing a mathematical model that approximates an unknown relationship between some input variables (or a set of input data) and a consequent response (expressed by a set of output data) is a frequent issue in a wide variety of fields. It goes under different names: “identification” in control theory, “supervised machine learning” in artificial intelligence, “regression and classification” in statistics, etc. As in all these cases the goal consists in reconstructing a function from a finite collection of samples, we use the expression “learning from data.”¹ Two steps are required: choosing the most suitable family of models and finding the best model inside such a family, i.e., the “closest” model – in a sense to be specified – to the true input/output (I/O) relationship on the basis of the available data (the so-called “training set”).

In this chapter, we address the problem of learning from data when fixed-structure parametrized (FSP) functions are adopted as I/O models. In particular, we concentrate mainly on the use of one-hidden-layer (OHL) networks. Section 4.1 considers the problem of modeling the I/O relationship by FSP functions using a finite number of samples. Two different learning frameworks are considered: the case in which it is possible to control the generation of the input data and the case where the input data come from an external unknown random environment. The latter case refers to the so-called *statistical learning theory* (SLT) and is the one where most research has been carried out in the machine learning community. We treat also the former case because it addresses some issues related to the choice of sampling points that are very relevant for the topics of the book (e.g., sampling of the state space in dynamic programming (DP); see Chaps. 6 and 7). In a context of controlled generation of input data, one has to distinguish between input data generated according to a

¹ A more suitable expression is actually “supervised learning from data,” to distinguish it from other forms of learning, such as “unsupervised learning from data.” However, since in the book we shall deal with supervised learning problems, the term “supervised” will be typically omitted.

suitable probability density function (Monte Carlo context) and data generated by a deterministic algorithm, e.g., quasi-Monte Carlo methods, orthogonal arrays, Latin hypercubes, etc. The latter possibility is an intriguing alternative to the random generation of input data since it allows, at least in principle, to exploit sets of points endowed with specific characteristics in order to improve the performance of the learning algorithm.

The structure of the error associated with the learning process is investigated in Sect. 4.2 by introducing the concepts of *expected risk*, *empirical risk*, and *generalization error*. The latter is then decomposed into *approximation error* and *estimation error*.

Section 4.3 briefly reviews some main concepts of SLT. Chapter 3 is devoted to the relationships among the accuracy ε , the number d of variables, and the model complexity n of the OHL networks, while Sect. 4.3 illustrates the relationships among such parameters and the number L of samples. The integer L is called *sample cardinality*. Sections 4.3.4 and 4.3.5 investigate in detail the cases of OHL networks with Gaussian and sigmoidal basis functions, respectively, which are used to approximate d -variable functions on the basis of a finite number of samples. These two sections provide upper bounds on the generalization error.

In Sect. 4.4, we address the case in which there is the possibility of generating the input data by deterministic algorithms. When the I/O relationship is deterministic, the entire learning process and its mathematical framework were named *deterministic learning* in [4]. Here, we shall use the terminology *deterministic learning theory* (DLT), to distinguish it from the widely accepted term SLT. Deterministic learning relies on some basic quantities like *variation* and *discrepancy*. In particular, special families of deterministic sequences called *low-discrepancy sequences* may be associated with upper bounds on the estimation error that decrease as $1/L$ (apart from a logarithmic factor), whereas the typical convergence rate in SLT is $1/L^{1/2}$.

Finally, if the input data are generated by a deterministic algorithm and the I/O relationship can be described by a deterministic function but the output is affected by a random additive noise, the resulting context is in a way a hybrid between SLT and DLT. This framework is considered in Sect. 4.4.9.

4.1 Learning from Data

In this section, we review some features of the problem of (supervised) learning from data. Given two sets $X \subseteq \mathbb{R}^d$ and $Y \subseteq \mathbb{R}$, we call $x \in X$ *independent variable* or *input variable* and $y \in Y$ *response* or *output variable*; (x, y) is an *input/output pair*.

It is useful to define the set of *input data* or *input samples*

$$X_L \triangleq \{\mathbf{x}^{(l)} \in X, l = 1, \dots, L\}. \quad (4.1)$$

The set (4.1) is generated by the sequence

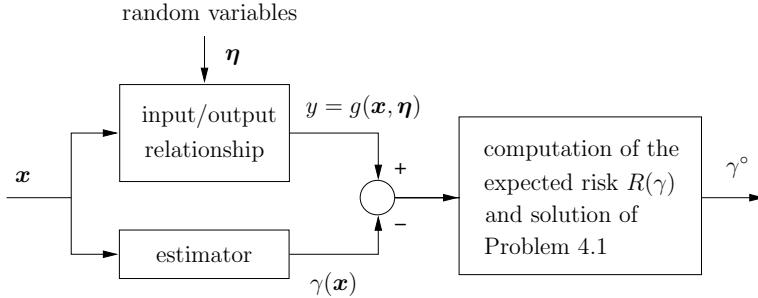


Fig. 4.1 Estimation/identification of an unknown input/output relationship

$$\{\mathbf{x}^L\} \triangleq \{\mathbf{x}^{(l)}\}_{l=1}^L. \quad (4.2)$$

Then, we introduce the set

$$\Sigma_L \triangleq \{(\mathbf{x}^{(l)}, y^{(l)}) \in X \times Y, l = 1, \dots, L\} \quad (4.3)$$

of L I/O data or samples or examples $(\mathbf{x}^{(l)}, y^{(l)})$, which are obtained by sampling L times the set $X \times Y$.

The target of the learning procedure is to design the mathematical model of the I/O relationship. More specifically, the problem consists in determining, on the basis of the training set Σ_L , a mapping

$$\gamma : X \rightarrow Y, \quad (4.4)$$

called *estimator*, that best represents (in some suitable sense) the unknown correspondence between the input variable \mathbf{x} and the output variable y (see Fig. 4.1). In particular, one wishes to find the *best* estimator, denoted by γ° , according to a suitable optimality criterion.

To avoid burdening the notation, we consider the case of a scalar output $y \in \mathbb{R}$. Then, we shall address scalar-valued estimators. In the case $Y \subset \mathbb{R}^m$ for some $m > 1$, one has to find a vector-valued function γ . In our framework, this can be dealt with by considering γ as composed by an m -tuple of scalar estimators and by applying the theory that we shall present in this chapter to each of them.

Depending on the type of the I/O relationship and on the possibility of controlling the generation of the sample set X_L , different situations may occur.

1. The I/O relationship to be “learned” or “identified” may be of *stochastic* or *deterministic* nature.
 - (a) In the stochastic case, the input and output variables are related by an unknown probability density function $p(\mathbf{x}, y)$ defined on $X \times Y$. Of course, $p(\mathbf{x}, y) = p(y|\mathbf{x}) p(\mathbf{x})$, where $p(y|\mathbf{x})$ is the conditional density of the response y , once the input variable \mathbf{x} is given. For example, referring to

the area of control systems, y may be the output variable of a noisy measurement device described by the unknown function (see (8.1))

$$y = g(\mathbf{x}, \boldsymbol{\eta}), \quad (4.5)$$

where \mathbf{x} is the state vector of a controlled plant and $\boldsymbol{\eta}$ is a random noise. Of course, any a priori information on the structure of the relationship is of great help in designing the estimator γ . For instance, if we know that the noise $\boldsymbol{\eta}$ is present in (4.5) in additive form, i.e.,

$$y = g(\mathbf{x}) + \eta, \quad (4.6)$$

then the task is greatly simplified. Note, in passing, that estimating g in (4.6) is strictly related to the *function approximation problem* in the presence of errors.

- (b) In the second situation, we shall address a *deterministic* I/O relationship. This means that (4.5) simply becomes

$$y = g(\mathbf{x}). \quad (4.7)$$

Having a deterministic I/O relationship corresponds to the limit situation in which the conditional density is the delta generalized function $p(y|\mathbf{x}) = \delta[y - g(\mathbf{x})]$.

- 2. As regards the generation of the input sample set X_L , we have two possible situations: *controllable* and *uncontrollable input generation*.

- (a) In the *uncontrollable* case, the set X_L is generated by an “external” environment, on which we cannot exercise any control. We may interpret the input samples $\mathbf{x}^{(l)}$, $l = 1, \dots, L$ as realizations of random variables generated by an autonomous source according to an unknown probability density $p(\mathbf{x})$. In the case of stochastic I/O relationships, the output samples $y^{(l)}$ are then obtained on the basis of the unknown conditional probability density $p(y|\mathbf{x})$. If the I/O link is deterministic, then the output samples are given by $y^{(l)} = g(\mathbf{x}^{(l)})$, in accordance with (4.7).

In the control area, the uncontrollable case typically occurs when the control system is working in real time and there is no possibility of freely generating \mathbf{x} (which is the input variable to the controlled plant or some other unknown device of the loop) in order to carry on identification experiments.

- (b) Unlike the previous case, in the *controllable* input generation one is able to control the input variables and to apply the samples $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(L)}$ (i) by drawing them randomly on the basis of a suitably chosen density function $p(\mathbf{x})$ or (ii) by generating the samples using some deterministic algorithm. In the case (i), the learning procedure is implemented in a Monte Carlo context, whereas in the case (ii) one can resort to quasi-Monte Carlo techniques,

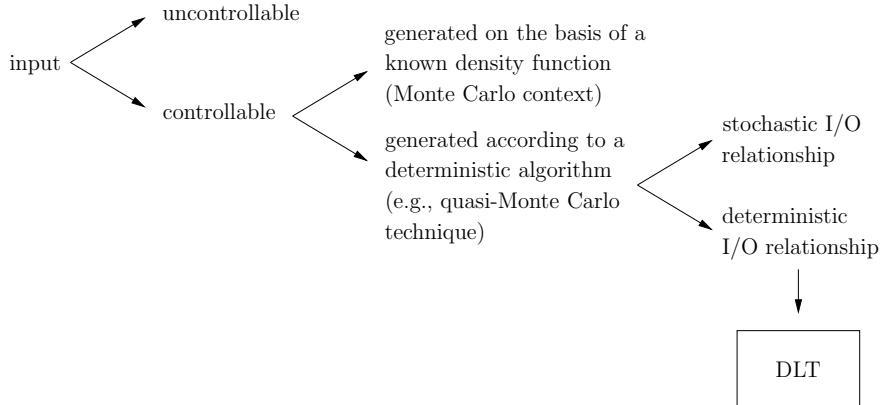


Fig. 4.2 A possible classification in learning from data. Note that only one learning framework belongs to DLT

orthogonal arrays, Latin hypercubes, etc. If the I/O relationship is deterministic, then the entire learning procedure is deterministic, too, and we refer to DLT in analogy with the original SLT paradigm [4]. Instead, if the I/O relationship is stochastic, we are – in a sense – in a hybrid situation between SLT and DLT. The various situations are summarized in Fig. 4.2.

The unknown conditional density $p(y|x)$ or the unknown function (4.7) provides the output samples $y^{(1)}, \dots, y^{(L)}$. It would be worth considering algorithms for the generation of input samples that exploit the values of the output samples $y^{(l)}$ observed one after the other. This approach is sometimes called *active learning* in the literature. However, this goes beyond our objectives.

The vectors in $\{\mathbf{x}^L\}$ are nearly always the first L elements of an infinite sequence $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots$. Depending on the specific case dealt with, it may be more suitable to refer to the set X_L or to the sequence $\{\mathbf{x}^L\}$.

Note that in the case of the generation of input samples by deterministic techniques (we shall address this subject in Sect. 4.4), we remain in a completely controllable input generation. If we use Monte Carlo methods, however, the term “controllable” has to be interpreted with some precaution: the input samples are realizations of random variables (so, we cannot “control” them) but we can choose the probability density function that generates them.

4.2 Expected Risk, Empirical Risk, and Generalization Error in Statistical Learning Theory

The SLT has reached a high level of completeness from both the theoretical and the experimental points of view. Instead, the DLT has a more recent history and many of its concepts are derived from SLT. So, first we present the main results of SLT, and then we describe the corresponding results of DLT more easily and concisely.

4.2.1 The Expected Risk

An intuitive measure of the error made by a given estimator γ is the functional

$$R(\gamma) \triangleq \int_{X \times Y} [y - \gamma(\mathbf{x})]^2 p(\mathbf{x}, y) d\mathbf{x} dy, \quad (4.8)$$

which is usually called the *expected risk* of γ . More general *loss functions* than the quadratic one can be used in the integrand (they will be addressed in Sect. 4.3.1). Our aim consists in finding the optimal estimator γ^* that minimizes $R(\gamma)$ over some family \mathcal{S} of functions γ to which γ^* is assumed to belong. In the terminology of learning theory, \mathcal{S} is called *concept class* and γ^* *target function*. Normally, a-priori information is useful to define \mathcal{S} . For example, if one knows that the unknown functional I/O relationship is endorsed with suitable regularity properties, it may be assumed that \mathcal{S} is a set of continuous functions or a subset of some Sobolev space or other.

In formal terms, if the density $p(\mathbf{x}, y)$ were known, one would have to solve the following infinite-dimensional optimization (IDO) problem, which is an instance of the basic Problem P.

Problem 4.1 Given a set \mathcal{S} of functions, find the optimal estimator $\gamma^* \in \mathcal{S}$ (that is, the target function) that minimizes the expected risk $R(\gamma)$ over \mathcal{S} . \triangleleft

Later in the chapter, we shall constrain the admissible solutions of Problem 4.1 to belong to more restricted classes of functions, called *hypothesis classes* in learning theory. As we have seen in the previous chapters, typical examples of such classes are families of FSP functions with given model complexities and, in particular, families of OHL networks with given basis cardinalities.

Let us now define the *regression function*

$$\gamma^*(\mathbf{x}) \triangleq \int_Y y p(y|\mathbf{x}) dy. \quad (4.9)$$

It is given by the conditional mean of the output variable y given the input variable \mathbf{x} . By adding and subtracting $\gamma^*(\mathbf{x})$ in (4.8), we have

$$\begin{aligned} R(\gamma) &= \int_X [\gamma^*(\mathbf{x}) - \gamma(\mathbf{x})]^2 p(\mathbf{x}) d\mathbf{x} + \int_{X \times Y} [y - \gamma^*(\mathbf{x})]^2 p(\mathbf{x}, y) d\mathbf{x} dy \\ &\quad + 2 \int_{X \times Y} [\gamma^*(\mathbf{x}) - \gamma(\mathbf{x})] [y - \gamma^*(\mathbf{x})] p(\mathbf{x}, y) d\mathbf{x} dy. \end{aligned} \quad (4.10)$$

The third term in the right-hand side of (4.10) is given by

$$\begin{aligned} 2 \int_{X \times Y} [\gamma^*(\mathbf{x}) - \gamma(\mathbf{x})] [y - \gamma^*(\mathbf{x})] p(y|\mathbf{x}) p(\mathbf{x}) d\mathbf{x} dy \\ = 2 \int_X [\gamma^*(\mathbf{x}) - \gamma(\mathbf{x})] \left\{ \int_Y [y - \gamma^*(\mathbf{x})] p(y|\mathbf{x}) dy \right\} p(\mathbf{x}) d\mathbf{x} = 0, \end{aligned}$$

since by Definition (4.9) of $\gamma^*(\mathbf{x})$ the quantity inside the curly brackets is equal to 0. So, we obtain

$$R(\gamma) = \int_X [\gamma^*(\mathbf{x}) - \gamma(\mathbf{x})]^2 p(\mathbf{x}) d\mathbf{x} + R(\gamma^*). \quad (4.11)$$

Suppose that \mathcal{S} is “sufficiently large” so that $\gamma^* \in \mathcal{S}$ (this assumption is made in the remaining of the chapter). Then, Eq. (4.11) implies that the regression function γ^* minimizes the expected risk over \mathcal{S} , as γ^* makes the first term of the left-hand side of (4.11) equal to zero. Hence, the optimal solution of Problem 4.1 is given by

$$\gamma^\circ = \gamma^* = \operatorname{argmin}_{\gamma \in \mathcal{S}} R(\gamma) \quad (4.12)$$

and

$$R(\gamma^\circ) = R(\gamma^*) = \int_{X \times Y} [y - \gamma^\circ(\mathbf{x})]^2 p(\mathbf{x}, y) d\mathbf{x} dy. \quad (4.13)$$

Note that the first term in the right-hand side of (4.11) will depend on the estimator γ we choose and on $p(\mathbf{x})$. Instead, the second term $R(\gamma^*)$ is due to the probabilistic nature of the learning problem and constitutes an intrinsic limitation below which the expected risk $R(\gamma)$ cannot decrease.

Let us consider the particular case of the estimation of the function g in (4.6) in presence of an additive noise η . If \mathbf{x} and η are mutually independent and g belongs to the set \mathcal{S} , $E(\eta) = 0$, and $\operatorname{var}(\eta) = \sigma^2$, then $\gamma^\circ(\mathbf{x}) = g(\mathbf{x})$ and $R(\gamma^\circ) = \sigma^2$ in (4.13) and this is the intrinsic limitation of the expected risk.

The following relationship, derived from (4.11) for $\gamma^* = \gamma^\circ$, will be useful later on:

$$\|\gamma - \gamma^\circ\|_{\mathcal{L}_2(p)}^2 = \int_X [\gamma(\mathbf{x}) - \gamma^\circ(\mathbf{x})]^2 p(\mathbf{x}) d\mathbf{x} = R(\gamma) - R(\gamma^\circ), \quad \forall \gamma \in \mathcal{S}, \quad (4.14)$$

where the notation $\|\cdot\|_{\mathcal{L}_2(p)}$ refers to the \mathcal{L}_2 -norm on X weighted by the probability density function $p(\mathbf{x})$.

4.2.2 The Empirical Risk

In most practical situations, γ^* cannot be computed since, in general, $p(y|\mathbf{x})$ is unknown. However, one can rely on the I/O samples in Σ_L and use them to estimate the regression function. Typically, we assume the I/O samples to be i.i.d., i.e., to be statistically independent outcomes of the same source. More formally, they are realizations of independent random variables having the same probability distribution. By means of such samples, one can evaluate the so-called *empirical (arithmetic) risk*, defined as

$$R_{\text{emp}}(\gamma, \Sigma_L) \triangleq \frac{1}{L} \sum_{l=1}^L [y^{(l)} - \gamma(\mathbf{x}^{(l)})]^2. \quad (4.15)$$

Note that in the empirical risk we use the set Σ_L as argument instead of the sample cardinality L . Indeed, using merely L , one cannot specify the structure of the I/O data set Σ_L .

It is worth noting that, in principle, one would like to minimize the expected risk $R(\gamma)$ over the concept class \mathcal{S} , thus obtaining γ° via the solution of Problem 4.1. Instead, since one cannot compute the expected risk because $p(\mathbf{x}, y)$ is unknown, one computes the empirical risk $R_{\text{emp}}(\gamma, \Sigma_L)$ by means of the L I/O samples at his/her disposal. In practice, to actually find a function that is sufficiently close to the target γ° , one resorts to the previously defined *hypothesis classes* embedded in the concept class \mathcal{S} . The functions belonging to such classes can be easily “tuned” (e.g., by optimizing suitable parameters) to approximate any reasonably well-behaved target function.

In the sequel, we resort to the extended Ritz method (ERIM), that is, we choose sets of OHL networks with increasing basis cardinality n or sets of FSP functions $\gamma(\mathbf{x}, \mathbf{w}_n)$, provided with the nestedness property, as hypothesis classes. Then, the IDO Problem 4.1 over \mathcal{S} is replaced with a sequence of finite-dimensional optimization (FDO) ones that minimize the expected risk over the nested sequences of sets

$$\mathcal{A}_1 \subset \cdots \subset \mathcal{A}_n \subset \cdots \subset \mathcal{S}. \quad (4.16)$$

Again, we assume that \mathcal{S} is “sufficiently large” so that not only γ° but also the hypothesis classes $\mathcal{A}_1, \dots, \mathcal{A}_n, \dots$ are embedded in it. Thus, for a given integer n , we state the following FDO problem, which is a nonlinear programming (NLP) problem.

Problem 4.2 Find the optimal parameter vector \mathbf{w}_n° (i.e., the optimal OHL network or the optimal FSP function γ_n°) that minimizes the expected risk

$$R(\gamma_n) = \int_{X \times Y} [y - \gamma(\mathbf{x}, \mathbf{w}_n)]^2 p(\mathbf{x}, y) d\mathbf{x} dy \quad (4.17)$$

over the set \mathcal{A}_n . \triangleleft

For the same class of functions, Empirical Risk (4.15) becomes

$$R_{\text{emp}}(\gamma_n, \Sigma_L) = \frac{1}{L} \sum_{l=1}^L [y^{(l)} - \gamma(\mathbf{x}^{(l)}, \mathbf{w}_n)]^2. \quad (4.18)$$

Now, we introduce the basic idea of the principle of *empirical risk minimization* (ERM principle). It consists in replacing the expected risk $R(\gamma_n)$ with the empirical risk $R_{\text{emp}}(\gamma_n, \Sigma_L)$ and minimizing the empirical risk instead of the expected risk. Working with (4.18) instead of (4.17) offers two important advantages.

1. The knowledge of $p(\mathbf{x}, y)$ is not needed, whereas it is required to compute $R(\gamma_n)$.
2. One can exploit the available I/O data set Σ_L .

Then, we can approximate the “ideal” Problem 4.2 with the following “practical” NLP problem:

Problem 4.3 Find the optimal parameter vector $\mathbf{w}_{n\Sigma_L}^\circ$ (i.e., the optimal OHL network or the optimal FSP function $\gamma_{n\Sigma_L}^\circ$) that minimizes Empirical Risk (4.18) over the set \mathcal{A}_n . \triangleleft

Clearly, the validity of the ERM principle depends on the behavior of the sequence $\{\gamma_{n\Sigma_L}^\circ\}_{L=1}^\infty$ that solves Problem 4.3 when L increases. More specifically, this sequence has to converge (in a suitable sense) to the function γ_n° minimizing the expected risk over the set \mathcal{A}_n . We have then to measure the “distance” between γ_n° and $\gamma_{n\Sigma_L}^\circ$. Consistently with the concept of expected risk, this distance is measured by the difference

$$R(\gamma_{n\Sigma_L}^\circ) - R(\gamma_n^\circ). \quad (4.19)$$

The next property states the meaning of the “convergence” of $R(\gamma_{n\Sigma_L}^\circ)$ to $R(\gamma_n^\circ)$ in the probabilistic framework of SLT, where the stochastic generation of the I/O samples makes the minimizing vector $\mathbf{w}_{n\Sigma_L}^\circ$ and the expected risk $R(\gamma_{n\Sigma_L}^\circ)$ random variables. To simplify the notation, in the following, we let

$$R_{\text{emp}}(\gamma_{n\Sigma_L}^\circ) \triangleq R_{\text{emp}}(\gamma_{n\Sigma_L}^\circ, \Sigma_L). \quad (4.20)$$

Property 4.1 (SLT framework) As $L \rightarrow \infty$, the expected risk $R(\gamma_{n\Sigma_L}^\circ)$ converges in probability to $R(\gamma_n^\circ)$, i.e.,

$$\lim_{L \rightarrow \infty} P \{ |R(\gamma_{n\Sigma_L}^\circ) - R(\gamma_n^\circ)| > \varepsilon \} = 0 \quad \text{for any } \varepsilon > 0. \quad (4.21)$$

\triangleleft

It is useful to anticipate here the concept of convergence of $R(\gamma_{n\Sigma_L}^\circ)$ to $R(\gamma_n^\circ)$ also in the deterministic framework considered in DLT. These risks will be explained in Sect. 4.4.1, which introduces DLT. According to what has been said in Sect. 4.1, the deterministic I/O relationship is given by (4.7). Furthermore, the sequence $\{\mathbf{x}^L\}$ is

generated by a deterministic algorithm. Since there is no reason to prefer certain regions of the input space X to others, the algorithm aims at spreading the samples $\mathbf{x}^{(l)}$ of $\{\mathbf{x}^L\}$ as uniformly as possible over X . Then, the density $p(\mathbf{x}, y)$ in (4.17) is no longer required and the risk simply becomes

$$R(\gamma_n) = \int_X [y - \gamma(\mathbf{x}, \mathbf{w}_n)]^2 d\mathbf{x},$$

with $y = g(\mathbf{x})$. Empirical Risk (4.8) and Problem 4.3 remain unchanged. Of course, $R(\gamma_{n,\Sigma_L}^\circ)$ and the minimizing vector $\mathbf{w}_{n,\Sigma_L}^\circ$ are no longer random variables. Thus, for this case, the convergence can be stated in a deterministic sense.

Property 4.2 (DLT framework) As $L \rightarrow \infty$, the risk $R(\gamma_{n,\Sigma_L}^\circ)$ converges to $R(\gamma_n^\circ)$, i.e.,

$$\lim_{L \rightarrow \infty} R(\gamma_{n,\Sigma_L}^\circ) = R(\gamma_n^\circ). \quad (4.22)$$

△

In Sects. 4.3 and 4.4, we shall give conditions for both Convergences (4.21) and (4.22) to hold, respectively.

4.2.3 The Generalization Error

We have seen that the “ideal” minimization of the expected risk $R(\gamma)$ over \mathcal{S} is replaced by the “realistic” minimization of $R_{\text{emp}}(\gamma, \Sigma_L)$ over \mathcal{A}_n . A question arises: “how good” $\gamma_{n,\Sigma_L}^\circ$ is? This, of course, depends on how one decides to measure the “goodness” of approximation. However, independently of this choice, one can evaluate qualitatively what happens when n and L go to infinity.

As regards the number L of samples, one expects the estimate of the expected risk to improve when such number grows. On the other hand, an increase of n allows one to have a larger number of free parameters to be tuned, so at first sight the estimate should improve. However, with a larger value of n and a fixed value of L , more parameters have to be estimated on the basis of the same number L of samples. Then, the estimate might become worse in the sense that it might be associated with a larger value of the expected risk. If one wishes the overall error to decrease to zero, the growths of n and L have to be related to each other. The rates of growth are determined by the properties of the density $p(\mathbf{x}, y)$ (on which the regression function depends), the set \mathcal{S} over which the expected risk would be ideally minimized, and the sequence $\{\mathcal{A}_n\}_{n=1}^\infty$ of sets over which the empirical risk is actually minimized.

In accordance with the criterion specified by (4.8), we go on evaluating the “goodness” of an estimator γ by the distance between γ and γ° measured in the weighted $\mathcal{L}_2(p)$ norm. Then, we define the so-called *generalization error*:

$$\begin{aligned}\mathcal{E}_{n\Sigma_L}^{\text{gen}} &\triangleq \|\gamma^\circ - \gamma_n^\circ\|_{\mathcal{L}_2(p)}^2 = \int_X [\gamma^\circ(\mathbf{x}) - \gamma^\circ(\mathbf{x}, \mathbf{w}_n^\circ)]^2 p(\mathbf{x}) d\mathbf{x} \\ &= R(\gamma_n^\circ) - R(\gamma^\circ).\end{aligned}\quad (4.23)$$

The last equality in (4.23) is derived from (4.14) (which is obtained under the assumption $\gamma^* = \gamma^\circ$), by letting $\gamma = \gamma_n^\circ$.

Although in general $p(\mathbf{x})$ and γ° are unknown, defining the generalization error is the starting point of major theoretical developments. Note that, in doing so, one decides to measure the error using the same probability $p(\mathbf{x})$ characterizing the variable \mathbf{x} . One would also like to construct bounds on the generalization error that depend on the model complexity n and on the sample cardinality L . This will be the subject of the next section.

4.2.4 The Approximation and Estimation Errors are Components of the Generalization Error

Two components contribute to the generalization error.

1. The first one derives from the practical impossibility of perfectly representing γ° by a function defined by a finite number of parameters (i.e., an OHL network or an FSP function). Such a component is measured by the quantity

$$\begin{aligned}\mathcal{E}_n^{\text{appr}} &\triangleq \|\gamma^\circ - \gamma_n^\circ\|_{\mathcal{L}_2(p)}^2 = \int_X [\gamma^\circ(\mathbf{x}) - \gamma_n^\circ(\mathbf{x})]^2 p(\mathbf{x}) d\mathbf{x} \\ &= R(\gamma_n^\circ) - R(\gamma^\circ),\end{aligned}\quad (4.24)$$

called *approximation error*. The last equality in (4.24) is derived from (4.14) by letting $\gamma = \gamma_n^\circ$. Note that $\mathcal{E}_n^{\text{appr}}$ does not depend on the specific sample set Σ_L but on the approximating properties of the hypothesis class that we are using (in our case, once n has been fixed, it depends only on the type of FSP function).

As we have seen in Chap. 3, the approximation error can be studied within the context of function approximation theory. More specifically, for many estimators γ implemented by OHL networks, the approximation error can be bounded from above by a quantity $\varepsilon(n)$ that, if $\{\mathcal{A}_n\}_{n=1}^\infty$ is dense in \mathcal{S} , goes to zero as n goes to infinity. The following inequality will be useful:

$$\mathcal{E}_n^{\text{appr}} \leq \varepsilon(n). \quad (4.25)$$

The rate of decrease to zero of $\varepsilon(n)$ generally depends on the smoothness properties of γ° and on the properties of $\{\mathcal{A}_n\}_{n=1}^\infty$.

2. The second component contributing to the generalization error is due to the limited number of I/O data at our disposal, which may contain insufficient information about the expected risk to be minimized. Actually, we minimize the

empirical risk over \mathcal{A}_n , thus obtaining $\gamma_{n\Sigma_L}^\circ$, instead of minimizing the expected risk, whose minimization over \mathcal{A}_n would give γ_n° . Indeed, as previously pointed out, it is impossible to compute this risk since the density $p(\mathbf{x}, y)$ is unknown. However, on the basis of what we said in Sect. 4.2.2, we can rely on the sample set Σ_L , consider Empirical Risk (4.15), solve Problem 4.3, and determine $\gamma_{n\Sigma_L}^\circ$. From (4.23), by adding and subtracting $R(\gamma_n^\circ)$, we get the following decomposition (see also [6]):

$$\mathcal{E}_{n\Sigma_L}^{\text{gen}} = \mathcal{E}_n^{\text{appr}} + R(\gamma_{n\Sigma_L}^\circ) - R(\gamma_n^\circ). \quad (4.26)$$

Now, we take into account the fact that $\gamma_{n\Sigma_L}^\circ$ has been obtained by minimizing the empirical risk R_{emp} in Problem 4.3. By adding and subtracting the term $R_{\text{emp}}(\gamma_{n\Sigma_L}^\circ)$ in (4.26), we get

$$\begin{aligned} \mathcal{E}_{n\Sigma_L}^{\text{gen}} &= \mathcal{E}_n^{\text{appr}} + R(\gamma_{n\Sigma_L}^\circ) - R_{\text{emp}}(\gamma_{n\Sigma_L}^\circ) + R_{\text{emp}}(\gamma_{n\Sigma_L}^\circ) - R(\gamma_n^\circ) \\ &\leq \mathcal{E}_n^{\text{appr}} + |R_{\text{emp}}(\gamma_{n\Sigma_L}^\circ) - R(\gamma_{n\Sigma_L}^\circ)| + |R_{\text{emp}}(\gamma_{n\Sigma_L}^\circ) - R(\gamma_n^\circ)|. \end{aligned} \quad (4.27)$$

Thus, we define the *estimation (or data) error*

$$\mathcal{E}_{n\Sigma_L}^{\text{est}} \triangleq |R_{\text{emp}}(\gamma_{n\Sigma_L}^\circ) - R(\gamma_{n\Sigma_L}^\circ)|. \quad (4.28)$$

Finally, by replacing (4.25) and (4.28) in (4.27), we get

$$\mathcal{E}_{n\Sigma_L}^{\text{gen}} \leq \varepsilon(n) + \mathcal{E}_{n\Sigma_L}^{\text{est}} + |R_{\text{emp}}(\gamma_{n\Sigma_L}^\circ) - R(\gamma_n^\circ)|.$$

Similarly, with a little abuse of terminology, one can also define the estimation error

$$\mathcal{E}_{n\Sigma_L}^{\text{est}}(\gamma_n) \triangleq |R_{\text{emp}}(\gamma_n, \Sigma_L) - R(\gamma_n)| \quad (4.29)$$

associated with each function $\gamma_n \in \mathcal{A}_n$.

As shown in Proposition 4.1, upper bounds on Estimation Error (4.29), which hold uniformly with respect to $\gamma_n \in \mathcal{A}_n$, allow one to bound from above the term $R(\gamma_{n\Sigma_L}^\circ) - R(\gamma_n^\circ)$ in the right-hand side of (4.26). Notice that different definitions of the estimation error are used in some literature. For instance, in [6], the term “estimation error” is used to denote the quantity $R(\gamma_{n\Sigma_L}^\circ) - R(\gamma_n^\circ)$.

The various estimators as well as the concept and the hypothesis classes are reported in Fig. 4.3 (inspired by Fig. 1 in [26]).

We know that the errors $\mathcal{E}_{n\Sigma_L}^{\text{est}}$ and $\mathcal{E}_{n\Sigma_L}^{\text{gen}}$ cannot be computed because of the impossibility of computing the various risks, except $R_{\text{emp}}(\gamma_{n\Sigma_L}^\circ)$. However, upper bounds can be often found for these errors. To this end, the result given in the following proposition is helpful and will be used elsewhere in this chapter to establish links among these bounds. The result is derived from the one reported in [25, Appendix C].

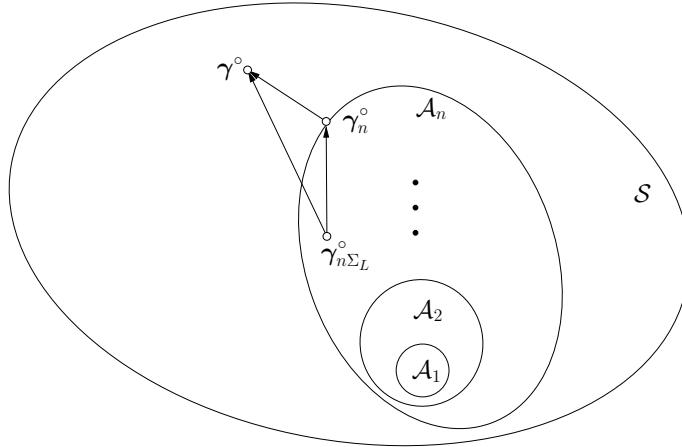


Fig. 4.3 The largest class represents the concept class \mathcal{S} . The nested sets $\mathcal{A}_1, \mathcal{A}_2, \dots$ (hypothesis classes) are embedded in \mathcal{S} . γ_n° is the optimal FSP function in \mathcal{A}_n , and $\gamma_{n\Sigma_L}^\circ$ is the element in \mathcal{A}_n that minimizes the empirical risk

Proposition 4.1 Let k be a positive constant and the inequality

$$|R_{\text{emp}}(\gamma_n, \Sigma_L) - R(\gamma_n)| \leq k, \quad \forall \gamma_n \in \mathcal{A}_n, \quad (4.30)$$

be satisfied. Then, also the inequality

$$R(\gamma_{n\Sigma_L}^\circ) - R(\gamma_n^\circ) \leq 2k \quad (4.31)$$

holds true. \square

Proof As Inequality (4.30) is uniform on \mathcal{A}_n , for $\gamma_n = \gamma_n^\circ$, we obtain

$$R_{\text{emp}}(\gamma_n^\circ, \Sigma_L) \leq R(\gamma_n^\circ) + k. \quad (4.32)$$

For $\gamma_n = \gamma_{n\Sigma_L}^\circ$, we also have

$$R(\gamma_{n\Sigma_L}^\circ) \leq R_{\text{emp}}(\gamma_{n\Sigma_L}^\circ) + k.$$

Since $\gamma_{n\Sigma_L}^\circ$ is an optimal solution to Problem 4.3, we have

$$R_{\text{emp}}(\gamma_{n\Sigma_L}^\circ) \leq R_{\text{emp}}(\gamma_n^\circ, \Sigma_L).$$

Therefore, the following chain of inequalities is verified, hence proving (4.31):

$$R(\gamma_{n\Sigma_L}^\circ) \leq R_{\text{emp}}(\gamma_{n\Sigma_L}^\circ) + k \leq R_{\text{emp}}(\gamma_n^\circ, \Sigma_L) + k \leq R(\gamma_n^\circ) + 2k. \quad \blacksquare$$

Suitable choices for k will make Inequality (4.31) useful in considering the estimation error in the contexts of both SLT and DLT. Of course, the distinction between SLT and DLT does not regard the approximation error (which, if OHL networks are chosen as FSP functions, can be investigated using the tools discussed in Chap. 3) but it is of major importance for the estimation error $\mathcal{E}_{n\Sigma_L}^{\text{est}}$.

The definitions of the approximation and estimation errors help one to understand the effect of increasing n and L to infinity, showing how this can lead, asymptotically, to the convergence to 0 of the generalization error. However, from a practical point of view, an important caveat must be kept in mind. In principle, we might want our hypothesis class to be as “rich as possible” (according to some measure of complexity like the ones that will be defined in the following sections), in the sense that its members are sufficiently flexible to allow us to find one such member that is as close as desired to the target function. In this way, the distance from the best function in the hypothesis class and the target function, i.e., the approximation error, can be arbitrarily reduced. In fact, however, the bounds on the estimation error that are presented in the following sections will show that the larger the complexity of the hypothesis class, the harder it is to find such an element inside it (i.e., the higher the estimation error), because of the large number of data samples required. Then, a trade-off arises between the richness of the class of estimators and the accuracy of the empirical risk minimization procedure to find the best function inside the class.

This kind of argumentation is also related to the well-known phenomenon called “bias-variance dilemma” that arises in statistical modeling problems (see, e.g., [12, 16, 18]). According to it, decreasing the bias of an estimator typically leads, for a given size of the sample data, to increase its variance.

4.3 Bounds on the Generalization Error in Statistical Learning Theory

We begin this section with a quite general remark pointed out in [26]. For estimators satisfying a variety of different hypotheses (see, e.g., the references given in [26]), it is possible to prove what follows by using tools from SLT.

Let $\delta \in (0, 1)$. With probability at least $1 - \delta$ on the random extraction of the I/O sample set Σ_L , let the bound

$$|R_{\text{emp}}(\gamma_n, \Sigma_L) - R(\gamma_n)| \leq \omega(L, n, \delta), \quad \forall \gamma_n \in \mathcal{A}_n \quad (4.33)$$

hold, where $\omega(L, n, \delta)$ is a suitable function of L, n , and δ . Then, the following result can be given.

Proposition 4.2 *For the estimators verifying Inequalities (4.25) and (4.33) (with probability at least $1 - \delta$ on the random extraction of the I/O sample set Σ_L), the generalization error can be bounded as follows:*

$$\mathcal{E}_{n \Sigma_L}^{\text{gen}} \leq \varepsilon(n) + 2\omega(L, n, \delta). \quad (4.34)$$

□

Proof For $k = \omega(L, n, \delta)$, Inequalities (4.31) and (4.33) give

$$R(\gamma_{n \Sigma_L}^\circ) - R(\gamma_n^\circ) \leq 2\omega(L, n, \delta). \quad (4.35)$$

From (4.25), (4.26), and (4.35), Inequality (4.34) follows. ■

Inequality (4.34) expresses the trade-off between the approximation error and the estimation error. For a fixed value of L , generally $\varepsilon(n)$ is a decreasing function of n but usually the quantity $\omega(L, n, \delta)$ increases with n . Given the number L of samples at our disposal, the ideal choice of n (e.g., of the basis cardinality in the chosen type of OHL network) should balance these two contributions. For a given family of OHL networks, one can estimate from above $\omega(L, n, \delta)$. As an example, in Sect. 4.3.4 we shall consider an upper bound on the generalization error for the case of radial OHL networks with Gaussian basis functions. The upper bound will be derived by estimating from above $\varepsilon(n)$ and $\omega(L, n, \delta)$. In Sect. 4.3.5, we shall consider the case of ridge OHL networks with sigmoidal basis functions.

4.3.1 Probabilistic Convergence of the Empirical Risk

In order to introduce some concepts (e.g., convergence of sequences, the ERM principle, etc.), which are not limited to quadratic expected and empirical risks, we consider more general loss functions. As regards the estimators, we go on using OHL networks or FSP functions $\gamma_n = \gamma(\cdot, \mathbf{w}_n)$.

Let us define, for every $\gamma_n \in \mathcal{A}_n$, the expected risk (compare with (4.17))

$$R^v(\mathbf{w}_n) \triangleq \mathbb{E}_{\mathbf{x}, y} \{v(y, \gamma_n)\} = \int_{X \times Y} v[y, \gamma(\mathbf{x}, \mathbf{w}_n)] p(\mathbf{x}, y) d\mathbf{x} dy, \quad (4.36)$$

and the empirical risk (compare with (4.18))

$$R_{\text{emp}}^v(\mathbf{w}_n, \Sigma_L) \triangleq \frac{1}{L} \sum_{l=1}^L v[y^{(l)}, \gamma(\mathbf{x}^{(l)}, \mathbf{w}_n)], \quad (4.37)$$

where

$$v : Y \times \mathbb{R} \rightarrow \mathbb{R} \quad (4.38)$$

is a *loss function*, which measures the error between y and its estimated value given by $\gamma(\mathbf{x}, \mathbf{w}_n)$. To simplify the notation, we write

$$V(z, \mathbf{w}_n) \triangleq v[y, \gamma(\mathbf{x}, \mathbf{w}_n)], \quad \mathbf{w}_n \in W_n, \quad (4.39)$$

where $z \triangleq \text{col}(\mathbf{x}, y) \in Z$, $Z \triangleq X \times Y$, and W_n is a set of parameters defining the set \mathcal{A}_n . As regards Risks (4.8) and (4.15), let γ_n° and $\gamma_{n\Sigma_L}^\circ$ be the FSP functions minimizing Expected Risk (4.36) and Empirical Risk (4.37) on \mathcal{A}_n , respectively. Such minimizing functions are obtained by solving problems similar to Problems 4.2 and 4.3. We also denote by \mathbf{w}_n° and $\mathbf{w}_{n\Sigma_L}^\circ$ the corresponding parameter vectors.

Accordingly, we define the estimation error in a general form, that is (compare with (4.28)),

$$\mathcal{E}_{n\Sigma_L}^{\text{est},v} \triangleq |R_{\text{emp}}^v(\mathbf{w}_{n\Sigma_L}^\circ, \Sigma_L) - R^v(\mathbf{w}_n^\circ)|. \quad (4.40)$$

The most widespread loss function in regression is the quadratic loss (4.8). Other loss functions are, e.g., the following:

$$V(z, \mathbf{w}_n) = |y - \gamma(\mathbf{x}, \mathbf{w}_n)|^p, \quad 1 \leq p < \infty, \quad (4.41)$$

employed in regression, and

$$V(z, \mathbf{w}_n) = \begin{cases} 0 & \text{if } y = \gamma(\mathbf{x}, \mathbf{w}_n) \\ 1 & \text{if } y \neq \gamma(\mathbf{x}, \mathbf{w}_n) \end{cases},$$

used in classification problems.

As stated previously, for any given estimator $\gamma_n \in \mathcal{A}_n$, the empirical risk $R_{\text{emp}}^v(\mathbf{w}_n, \Sigma_L)$ is a random variable since it depends on the I/O sample set Σ_L . Therefore, we go on considering the *convergence in probability* also for $R_{\text{emp}}^v(\mathbf{w}_{n\Sigma_L}^\circ)$, that is,

$$\lim_{L \rightarrow \infty} P \left\{ \left| R_{\text{emp}}^v(\mathbf{w}_{n\Sigma_L}^\circ, \Sigma_L) - R^v(\mathbf{w}_n^\circ) \right| > \epsilon \right\} = 0, \quad \forall \epsilon > 0. \quad (4.42)$$

Let us consider the convergence in probability of Expected Risk (4.36) computed for $\gamma_{n\Sigma_L}^\circ$, that is,

$$\lim_{L \rightarrow \infty} P \left\{ \left| R^v(\mathbf{w}_{n\Sigma_L}^\circ) - R^v(\mathbf{w}_n^\circ) \right| > \epsilon \right\} = 0, \quad \forall \epsilon > 0. \quad (4.43)$$

Then, the *ERM principle is said to be consistent* for the set of loss functions $V(z, \mathbf{w}_n)$, $\mathbf{w}_n \in W_n$, and for the probability density $p(z)$, if it provides a sequence of estimators $\gamma_{n\Sigma_L}^\circ$, $L = 1, 2, \dots$, for which both Conditions (4.42) and (4.43) are verified. (4.43) asserts that the expected risks of the minimizers of the empirical risks converge to $R^v(\mathbf{w}_n^\circ)$, whereas (4.42) asserts that one can estimate $R^v(\mathbf{w}_n^\circ)$ on the basis of the minimum value of the empirical risk.

In [34], a drawback of defining consistency in terms of the requirements (4.42) and (4.43) is pointed out. Indeed, such a definition does not allow one to derive conditions for consistency that are expressed in terms of only general properties of the set \mathcal{A}_n of parametrized estimators and of the probability density $p(\mathbf{x}, y)$. By “general properties,” we mean properties that do not depend on a specific function

$\gamma_n \in \mathcal{A}_n$ but are pertinent to the set \mathcal{A}_n “as a whole.” One of such properties is the *Vapnik–Chervonenkis dimension* (VC dimension). For this reason, the following definition of *nontrivial consistency* is introduced (for a deeper understanding of the foregoing we refer to the example given in [34, p. 37]).

Definition 4.1 ([34]) The ERM principle is said to be “nontrivially consistent” for the set of the loss functions $V(\cdot, \mathbf{w}_n)$, $\mathbf{w}_n \in W_n$ and the probability density $p(z)$ if, for any non-empty subset W_n , $c \in (-\infty, +\infty)$, and

$$W_n(c) \triangleq \{\mathbf{w}_n \in W_n : R^v(\mathbf{w}_n) > c\},$$

one has

$$\lim_{L \rightarrow \infty} P \left\{ \left| \inf_{\mathbf{w}_n \in W_n(c)} R_{\text{emp}}^v(\mathbf{w}_n, \Sigma_L) - \inf_{\mathbf{w}_n \in W_n(c)} R^v(\mathbf{w}_n) \right| > \varepsilon \right\} = 0, \quad \forall \varepsilon > 0. \quad (4.44)$$

□

Clearly, (4.44) implies (4.42). Moreover, it can be shown [33, p. 82] that Condition (4.43) is implied by (4.44). To be more specific, we give the following result (see the lemma in [33, p. 82]).

Proposition 4.3 *If the ERM principle is nontrivially consistent, Convergences in Probability (4.42) and (4.43) hold. Then, Property 4.1 is verified.*

□

We can now present the following key result by Vapnik and Chervonenkis [36], which plays a central role in SLT. To simplify the terminology, from now on we shall call nontrivial consistency merely “consistency”.

Theorem 4.1 ([34, Theorem 2.1]) *Let constants a and b exist such that, for a given density function p , all loss functions $V(\cdot, \mathbf{w}_n)$, $\mathbf{w}_n \in W_n$, satisfy the inequalities*

$$a \leq R^v(\mathbf{w}_n) = \int_{X \times Y} V(z, \mathbf{w}_n) p(z) dz \leq b.$$

Then, for the ERM principle to be consistent, it is necessary and sufficient that the empirical risk $R_{\text{emp}}^v(\mathbf{w}_n, \Sigma_L)$ has the property of “one-side uniform convergence in probability” to the expected risk $R^v(\mathbf{w}_n)$, i.e.,

$$\lim_{L \rightarrow \infty} P \left\{ \sup_{\mathbf{w}_n \in W_n} \left[R^v(\mathbf{w}_n) - R_{\text{emp}}^v(\mathbf{w}_n, \Sigma_L) \right] > \varepsilon \right\} = 0, \quad \forall \varepsilon > 0. \quad (4.45)$$

□

The proof of Theorem 4.1 is given in [33]. Note that the term “uniform convergence” comes from the presence of the “sup” with respect to $\mathbf{w}_n \in W_n$ in (4.45). Note also that the type of uniform convergence in (4.45) is called “one-sided” in contrast with the uniform “two-sided” convergence defined by the equation

$$\lim_{L \rightarrow \infty} P \left\{ \sup_{\mathbf{w}_n \in W_n} \left| R^v(\mathbf{w}_n) - R_{\text{emp}}^v(\mathbf{w}_n, \Sigma_L) \right| > \varepsilon \right\} = 0, \quad \forall \varepsilon > 0. \quad (4.46)$$

4.3.2 The VC Dimension of a Set of Functions

Theorem 4.1 is the departure point for deriving probabilistic uniform non-asymptotic (i.e., for finite values of L) bounds on the estimation error. Let us consider a set \mathcal{G} of fixed-structure functions $\varphi(z, \mathbf{w})$ parametrized by the vector \mathbf{w} (i.e., the FSP functions in our terminology), that is,

$$\mathcal{G} \triangleq \{\varphi(\cdot, \mathbf{w}) : \mathbb{R}^q \rightarrow \mathbb{R} \text{ such that } \mathbf{w} \in W \subseteq \mathbb{R}^p\}, \quad (4.47)$$

where W is a set of parameters.

The bounds on the estimation error are determined in terms of a property of the function family \mathcal{G} . This property is expressed by a scalar, called *VC dimension*. The VC dimension was initially conceived for sets of indicator functions [35] and then generalized to sets of real-valued functions [32]. Therefore, we assume at first that \mathcal{G} is a set of indicator functions. We know that any indicator function separates a given set of vectors into two subsets: a subset of vectors where the indicator function takes on the value 0 and a subset where the function takes on the value 1.

Now, the VC dimension of a set \mathcal{G} of indicator functions $\varphi(z, \mathbf{w})$ is the maximal number h of vectors z_1, \dots, z_h that, using functions of \mathcal{G} , can be separated into two classes in all the possible 2^h pairs of distinct subsets of $\{z_1, \dots, z_h\}$, including the empty set and the set itself. Clearly, if for any positive integer n there exists a set of n vectors that can be separated by the functions of \mathcal{G} , then the VC dimension of \mathcal{G} is infinite.

One can easily generalize the concept of VC dimension to *sets of real-valued functions*. Consider a set (4.47) where $\varphi(z, \mathbf{w})$ are real-valued FSP functions bounded by some constants A and B , that is,

$$A \leq \varphi(z, \mathbf{w}) \leq B,$$

for every $z \in \mathbb{R}^q$ and $\mathbf{w} \in \mathbb{R}^p$. For any function of this type, we can construct the indicator function

$$I(z, \mathbf{w}, a) \triangleq \theta[\varphi(z, \mathbf{w}) - a], \quad (4.48)$$

where $\theta(t)$ is the step function

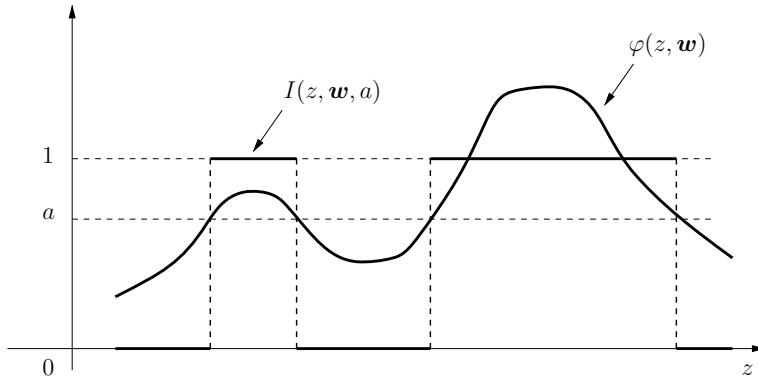


Fig. 4.4 Relationship between $\varphi(z, \mathbf{w})$ and the corresponding indicator function for a given value of the “level” a

$$\theta(t) \triangleq \begin{cases} 1, & \text{if } t \geq 0 \\ 0, & \text{if } t < 0 \end{cases},$$

and a is a constant such that $A \leq a \leq B$.

Now, we can construct the following set of indicator functions parametrized by \mathbf{w} and a :

$$\mathcal{G}_I \triangleq \{I(\cdot, \mathbf{w}, a) : \mathbb{R}^q \rightarrow \text{such that } \mathbf{w} \in W \subseteq \mathbb{R}^p, A \leq a \leq B\}. \quad (4.49)$$

As can be seen in Fig. 4.4 for the scalar case $z \in \mathbb{R}$, the indicator function “of level a ” for the real-valued function $\varphi(z, \mathbf{w})$ shows for which z the function $\varphi(z, \mathbf{w})$ exceeds a and for which it does not. Then, $\varphi(z, \mathbf{w})$ can be described by the set of all its indicators.

We conclude by defining the VC dimension of a set (4.47) of real-valued functions as the VC dimension of the set (4.49) of the corresponding indicator functions (4.48), obtained by varying both \mathbf{w} and a . From the foregoing, we can also provide the following equivalent definition of the VC dimension of the set (4.47) of real FSP functions. The definition stands independently of the introduction of indicator functions.

Definition 4.2 The VC dimension of a set \mathcal{G} of real-valued functions defined by (4.47) is the maximal number h of vectors $\mathbf{z}_1, \dots, \mathbf{z}_h$ that can be separated into all the 2^h possible pairs of distinct subsets of $\{\mathbf{z}_1, \dots, \mathbf{z}_h\}$, including the empty set and the set itself. The vectors \mathbf{z}_i belonging to one subset verify the first of the following inequalities; the vectors belonging to the other subset verify the second inequality, i.e.,

$$\begin{aligned} \varphi(\mathbf{z}_i, \mathbf{w}) - a &\geq 0 \\ \varphi(\mathbf{z}_i, \mathbf{w}) - a &< 0, \end{aligned} \quad (4.50)$$

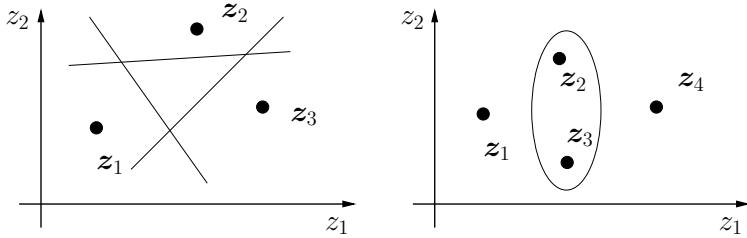


Fig. 4.5 In \mathbb{R}^2 , the lines s , given by $w_1 z_1 + w_2 z_2 = a - w_0$, can shatter any three vectors not on the same line, but not four. Then, the VC dimension of the set of linear functions is $h = 3$. ©2000 Adapted by permission from Springer Nature: [34]

where a is some scalar. If all the separations are possible, then we say that the set $\{z_1, \dots, z_h\}$ is “shattered” by \mathcal{G} . \triangleleft

Note that the term “shattered” has the same meaning as “completely distinguished.”

In order to present a simple example to determine graphically the VC dimension of a set of functions, let us address the set of FSP linear functions

$$\mathcal{G}_{\text{lin},2} \triangleq \{\varphi(\cdot, \mathbf{w}) = w_0 + w_1 z_1 + w_2 z_2 : \mathbb{R}^2 \rightarrow \mathbb{R} \text{ such that } w_0, w_1, w_2 \in \mathbb{R}\}, \quad (4.51)$$

where $\mathbf{z} \triangleq \text{col}(z_1, z_2)$ and $\mathbf{w} \triangleq \text{col}(w_0, w_1, w_2)$. Clearly, in this case any subset of vectors $z_1, \dots, z_h \in \mathbb{R}^2$ belongs to one of the two half planes separated by the straight line s given by

$$w_1 z_1 + w_2 z_2 = a - w_0.$$

The line s can be translated or rotated in all possible ways by varying the parameters w_1, w_2 , and $a - w_0$ (this difference can be considered as a single parameter). If the lines s are drawn in the plane \mathbb{R}^2 , it can be seen that all possible lines can shatter any three vectors not on the same line, but not four (see Fig. 4.5). Then, the VC dimension of the set (4.51) of linear functions is $h = 3$.

The reasoning can be extended to the set of linear functions

$$\mathcal{G}_{\text{lin},q} \triangleq \left\{ \varphi(\cdot, \mathbf{w}) = \sum_{i=1}^q w_i z_i + w_0 : \mathbb{R}^q \rightarrow \mathbb{R} \text{ such that } w_0, w_1, \dots, w_q \in \mathbb{R} \right\}, \quad (4.52)$$

where $\mathbf{z} \triangleq \text{col}(z_1, \dots, z_q)$, $\mathbf{w} \triangleq \text{col}(w_0, w_1, \dots, w_q)$. One can conclude that the VC dimension of the set (4.52) is $h = q + 1$.

Some examples of calculating the VC dimension via its definition (i.e., by imposing the shattering property) are reported, among others, in [5]. Unfortunately, this approach works only for rather simple sets of functions. A general approach, based on experimental results, is discussed in [5].

An interesting case regards *linear combinations of fixed-basis functions* (LCFBFs) with a scalar output and an additional constant basis function. For these

FSP functions, we have (see (3.1))

$$\mathcal{G}_{\text{LCFBF},n} \triangleq \left\{ \varphi(\cdot, \mathbf{w}) = \sum_{i=1}^n w_i g_i(\mathbf{z}) + w_0 : \mathbb{R}^q \rightarrow \mathbb{R} \text{ such that} \right. \\ \left. w_0, w_1, \dots, w_n \in \mathbb{R} \right\},$$

where \mathbf{w} is the usual weight vector. As the basis functions have been assumed to be linearly independent, we can introduce the $(n+1)$ -dimensional space spanned by the set $\{g_1(\mathbf{z}), \dots, g_n(\mathbf{z}), 1\}$. Then, the set of LCFBFs is equivalent to the set of linear functions (4.52) and it has the same VC dimension $h = n+1$.

Let us consider a particular case of OHL networks (see (3.3)), where the parameter vectors $\boldsymbol{\kappa}_i$ are the same for all basis functions and equal to $\boldsymbol{\kappa}$, that is,

$$\mathcal{G}'_{\text{OHL},n} \triangleq \left\{ \varphi(\cdot, \mathbf{w}) = \sum_{i=1}^n w_i g_i(\mathbf{z}, \boldsymbol{\kappa}) + w_0 : \mathbb{R}^q \rightarrow \mathbb{R} \text{ such that} \right. \\ \left. \boldsymbol{\kappa} \in \mathbb{R}^k, w_0, w_1, \dots, w_n \in \mathbb{R} \right\}.$$

Unlike the case of LCFBFs, the computation of the VC dimension may now be quite difficult, even if the VC dimension of the set of functions $g_i(\cdot, \boldsymbol{\kappa})$ is known. In particular, the VC dimension of the set associated with two parametrized basis functions may be infinite even when the VC dimension of the set associated with each basis function is finite [5, Sect. 4.2].

It is important to point out that, in general, the VC dimension does not correspond to the number of parameters of the FSP functions, i.e., to $\dim(\mathbf{w})$. For example, the family

$$\mathcal{G}_{\sin} \triangleq \{ \varphi(\cdot, w) = \sin(wz) : \mathbb{R} \rightarrow \mathbb{R} \text{ such that } w \in \mathbb{R} \}$$

contains only one parameter, but its VC dimension is infinite as it is easy to show (see e.g., [18]) that any number of suitably located points of the interval $[-1, 1]$ of the real line can be shattered by a sinusoidal function with sufficiently high frequency.

For a thorough treatment of the VC dimension, we refer the reader, among others, to [5, 7, 18, 33, 37].

4.3.3 Bounds on the Estimation Error

We are now in a position to provide bounds on the estimation errors of the FSP estimators $\gamma(\cdot, \mathbf{w})$ introduced in Sect. 4.2.2 and considered from there on. These estimators neither are necessarily given by OHL networks nor are required to belong to nested sets of FSP functions. Then, there is no need to characterize them by the

basis cardinality n (in the case of OHL networks) or other integers specifying their model complexity (in the case of generic FSP functions).

Let us consider the error

$$\mathcal{E}_{\Sigma_L}^{\text{est}, v} \triangleq |R_{\text{emp}}^v(\mathbf{w}, \Sigma_L) - R^v(\mathbf{w})|, \quad (4.53)$$

which differs from Estimation Error (4.40) for the absence of the subscript n as it refers to a generic parameter vector \mathbf{w} . Reasoning likewise in the proof of Proposition 4.1, an upper bound on (4.53), valid for any admissible \mathbf{w} , provides also an upper bound on Estimation Error (4.40), and also on $R^v(\mathbf{w}_{n\Sigma_L}^\circ) - R^v(\mathbf{w}_n^\circ)$. For this reason, similarly to the case of the quadratic loss function, with a little abuse of terminology, in the following we call also (4.53) “estimation error.” Now, we consider a general loss function with the structure (4.39) for $\mathbf{z} = \text{col}(\mathbf{x}, \mathbf{y})$, with $\dim(\mathbf{z}) = q = d + 1$, that is,

$$V(\mathbf{z}, \mathbf{w}) = v[y, \gamma(\mathbf{x}, \mathbf{w})], \quad \mathbf{w} \in W \subseteq \mathbb{R}^p. \quad (4.54)$$

In (4.53) the set Σ_L has the same meaning as in (4.37) (see the compact notation in (4.39) and (4.54)) and it is given by $\Sigma_L = \{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(L)}\}$. Note that the estimator γ does not figure explicitly in $V(\cdot, \mathbf{w})$ since the structure of γ is fixed; consequently, γ is completely specified by \mathbf{w} . Then, we define a set of loss functions taking on the form (4.47), i.e.,

$$\mathcal{G}_v \triangleq \{V(\cdot, \mathbf{w}) : \mathbb{R}^{d+1} \rightarrow \mathbb{R} \text{ such that } \mathbf{w} \in W \subseteq \mathbb{R}^p\}. \quad (4.55)$$

Our aim is to determine an upper bound on Estimation Error (4.53) pointing out the dependence of this bound on the VC dimension of the set \mathcal{G}_v . Of course, one may wonder why we consider the VC dimension of the loss function $V(\cdot, \mathbf{w})$ instead of the VC dimension of the set of estimators $\gamma(\cdot, \mathbf{w})$, which are the subject of our attention. Clearly, once the mapping γ is fixed, the VC dimension of \mathcal{G}_v is related to the one of the set of the estimators

$$\mathcal{G}_\gamma \triangleq \{\gamma(\cdot, \mathbf{w}) : \mathbb{R}^d \rightarrow \mathbb{R} \text{ such that } \mathbf{w} \in W \subseteq \mathbb{R}^p\}. \quad (4.56)$$

through (4.54). However, in many practical cases the connection between the VC dimension h of \mathcal{G}_v and the VC dimension of \mathcal{G}_γ (denote it by h_γ) is rather simple. This statement is summarized in [5, Sect. 4.2.1], where conclusions drawn from Vapnik’s results are reported.

Consider a quadratic loss function

$$V(\mathbf{z}, \mathbf{w}) \triangleq [y - \gamma(\mathbf{x}, \mathbf{w})]^2. \quad (4.57)$$

In [5, Sect. 4.2.1] it is shown that the VC dimension of (4.57) is bounded as follows:

$$h_\gamma \leq h \leq ch_\gamma,$$

where c is an absolute constant.

More in general, for loss functions of practical interest, one can let

$$h_\gamma \simeq h .$$

To sum up, the VC dimension of a set \mathcal{G}_v of loss functions $V(\cdot, \mathbf{w})$ usually equals approximately the VC dimension of the associated set \mathcal{G}_γ of estimators. We can now state a fundamental theorem that gives an upper bound on Estimation Error (4.53) (see [34, Sect. 3.7]; see also [13]).

Theorem 4.2 *Let $V(\cdot, \mathbf{w})$ be a loss function such that the inequalities $A \leq V(\cdot, \mathbf{w}) \leq B$ hold true for any $\mathbf{w} \in W$ (A and B are suitable constants), the VC dimension h of the set \mathcal{G}_v be finite, and L be the number of samples $\mathbf{z}^{(l)} \in \Sigma_L$, drawn independently and according to the probability density function $p(\mathbf{z})$. Then, for any $0 < \delta < 1$, the following inequality holds simultaneously for all $\mathbf{w} \in W$ with probability at least $1 - \delta$:*

$$\mathcal{E}_{\Sigma_L}^{\text{est}, v} = |R_{\text{emp}}^v(\mathbf{w}, \Sigma_L) - R^v(\mathbf{w})| \leq (B - A) \sqrt{\frac{h \ln \frac{2eL}{h} - \ln \frac{\delta}{4}}{L}}, \quad (4.58)$$

where e is the base of the natural logarithm. \square

Note that Bound (4.58) is

- probabilistic;
- uniform, i.e., it holds for any $V(\cdot, \mathbf{w}) \in \mathcal{G}_v$;
- distribution-independent, i.e., it does not depend on $p(\mathbf{z})$.

Let

$$\omega'(L, h, \delta) \triangleq (B - A) \sqrt{\frac{h \ln \frac{2eL}{h} - \ln \frac{\delta}{4}}{L}} . \quad (4.59)$$

The bound (4.59) points out very clearly its dependence on the ratio L/h . From [34, p. 93], we report the following: “The sample size L is considered to be small if the ratio L/h ... is small, say $L/h < 20$.”

The following remarks can be made on the dependence of ω' on h , L , and d :

- **Dependence on h and L ; trade-off between h and L .** Note that, for a fixed sample cardinality L , when $h \rightarrow \infty$ the upper bound ω' grows to infinity, too. The finiteness of the VC dimension guarantees distribution-independent convergence of empirical means to expected values and then consistency of the ERM principle. Looking at the expression in (4.59), we can see that, in terms of L alone, the bound on the estimation error is of order $O((1/L)^{1/2})$ (ignoring the logarithmic factor). We can also see that there exists a trade-off between h and L . In fact, a higher value of h requires a larger number L of samples to obtain the same guaranteed accuracy in the estimation error. We also point out the compromise between the

two competing tasks of approximation and estimation: in order to have a low approximation error, we need to increase the “richness” of the hypothesis class of estimators so that we can get arbitrarily close to γ° (see Fig. 4.3). But, at the same time, in order to keep the ERM procedure computationally feasible in terms of L , we need our class of models to be “simple”, so that the VC dimension h is not too large.

Of course, in order to make all this clear, we have to specify the meaning of the term “richness” of the hypothesis classes. Recall that we have established the dependence of the sets \mathcal{G}_v of loss functions on the sets \mathcal{G}_γ of estimators (see (4.54), (4.55), and (4.56)). We have also stated that, for several loss functions of practical interest, $h_\gamma \simeq h$. Now, we provide a “structure” to \mathcal{G}_v and \mathcal{G}_γ by indexing the parameter vectors \mathbf{w} , i.e., by writing \mathbf{w}_n . Recall that the integer $n = 1, 2, \dots$ denotes the *basis cardinality* if the estimators γ are OHL networks or the *model complexity* in the more general case in which FSP functions are used. In the following, we assume that they are endowed with the nestedness property. For each n , the set \mathcal{G}_γ corresponds to the hypothesis class \mathcal{A}_n of the estimators γ_n . Then, we obtain a nested sequence similar to (4.16), which we rewrite here as (see also Fig. 4.3)

$$\mathcal{A}_1 \subset \cdots \subset \mathcal{A}_n \subset \cdots \subset \mathcal{S}.$$

We can now derive an upper bound on the expected risk $R^v(\mathbf{w}_n)$. Let us assume that the VC dimension h_n of each set \mathcal{A}_n is finite. Note that the sets \mathcal{A}_n are such that the corresponding VC dimensions grow with n , i.e.,

$$h_1 \leq h_2 \leq \cdots \leq h_n \leq \cdots.$$

From (4.58) we obtain the following inequality, which holds with probability at least $1 - \delta$ simultaneously for all $\mathbf{w}_n \in W_n$:

$$R^v(\mathbf{w}_n) \leq R_{\text{emp}}^v(\mathbf{w}_n, \Sigma_L) + \omega'(L, h_n, \delta). \quad (4.60)$$

Note that in (4.60), for a fixed value of the probability δ , two variables are involved, that is, L and n , hence h_n ; we want to establish a link between them. Note also that the first term in the right-hand side of (4.60) depends on the specific FSP function γ_n belonging to the set \mathcal{A}_n , whereas the second term depends on the VC dimension of the whole set \mathcal{A}_n .

From (4.20) and (4.60) one has also, with probability at least $1 - \delta$,

$$R^v(\mathbf{w}_{n\Sigma_L}^\circ) \leq R_{\text{emp}}^v(\mathbf{w}_{n\Sigma_L}^\circ) + \omega'(L, h_n, \delta) \quad (4.61)$$

for the minimizer $\mathbf{w}_{n\Sigma_L}^\circ$ of the empirical risk on W_n .

Now, for a given set Σ_L of data, we can express Upper Bound (4.61) on the expected risk $R^v(\mathbf{w}_{n\Sigma_L}^\circ)$ as a function of the model complexity n of the estimator. The number n exerts an opposite influence on the two terms of (4.61). As n increases, the empirical risk $R_{\text{emp}}^v(\mathbf{w}_{n\Sigma_L}^\circ)$ tends to decrease (due to the nestedness

of the sets \mathcal{A}_n), while the second term $\omega'(L, h_n, \delta)$ (which controls the amplitude of the *confidence interval* where the expected risk $R^v(\mathbf{w}_{n\Sigma_L}^\circ)$ lies with probability at least $1 - \delta$) increases (see Fig. 4.6). The procedure that finds the smallest upper bound to the expected risk $R^v(\mathbf{w}_{n\Sigma_L}^\circ)$ (that is, the trade-off between the two terms in (4.61)) is called *structural risk minimization* (SRM) principle in [34]. In practice, different values for δ are actually used in (4.60), to guarantee – via the so-called *union-bound technique* [23] – that they hold simultaneously for all $\mathbf{w}_n \in W_n$ and all the sets W_n .

Taking into account the “competition” between the empirical risk and the confidence interval, the SRM principle determines the optimal model complexity n^* by choosing the subset \mathcal{A}_{n^*} (or, equivalently, the VC dimension h_{n^*}) in the sequence (4.16) that minimizes the upper bound (4.61) on the expected risk $R^v(\mathbf{w}_{n\Sigma_L}^\circ)$. Then, the optimal estimator is obtained by minimizing the empirical risk $R_{\text{emp}}^v(\mathbf{w}_{n^*}, \Sigma_L)$ within the subset \mathcal{A}_{n^*} . Of course, what has been said has a practical meaning provided that the VC dimension can be determined exactly, which is quite difficult in most cases, or at least that suitable upper bounds are known on it.

- **Dependence on d .** It is very important to point out that $\omega'(L, h, \delta)$ may depend on d . This dependence is “hidden” in the possible dependence on d of the VC dimension h , which makes (4.59) become a function $\omega''(L, d, \delta)$. For instance, the VC dimension of the set of indicator functions of d -dimensional balls is equal to $d + 1$ [9]. In [13], it was shown that this is also the VC dimension of the set of indicator functions associated with the d -dimensional Gaussian kernel. If h is independent on d , then ω' is such, as well. When it grows moderately with d , (4.58) is an upper bound on Estimation Error (4.53) that grows moderately with d , too. In the case of OHL networks, the computation of the VC dimension as a function of d may be very difficult. Anyway, the finiteness of the VC dimension has been proved for various OHL networks. For sigmoidal neural networks, finiteness of the VC dimension has been proved for different sigmoids, and non-exponential bounds in d have been reported (see, e.g., [7]).

Besides the VC dimension, the problem of learning from data has been formalized through other related paradigms in the literature, such as the *probably approximately correct* (PAC) paradigm [17] or the *Glivenko–Cantelli* paradigm [10]. At the same time, other measures of complexity of the involved functions have been introduced in place of the VC dimension, leading to different expressions for the bounds on the convergence of the estimation error. Among these, we can cite *Pollard dimension* [29], *effective dimension* [15], ε -*covering*, and ε -*packing numbers* (see, e.g., [37]). The Ref. [1] illustrates equivalences and similarities among various formalizations and various dimensions. Finally, we recall that in the last years, new estimation bounds for SLT were derived, which improve those based on the VC dimension. A survey of such results, which are expressed in terms of a quantity called *Rademacher’s complexity*, can be found in [23]. There are several reasons for which these estimates are better than those based on the VC dimension. For our purposes, it is enough to recall that classical SLT bounds are obtained by “reducing” a family of functions to a simpler one via ε -coverings, then applying to such a simpler family the union-bound

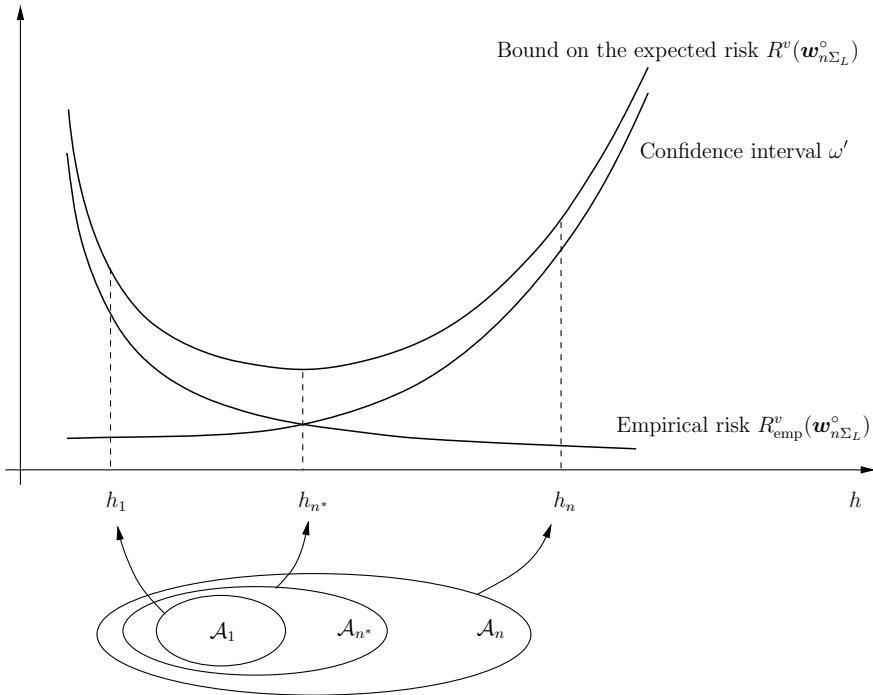


Fig. 4.6 The upper bound on the expected risk is the sum of the empirical risk and the confidence interval. The smallest of the upper bounds on the expected risk is achieved by the subset \mathcal{A}_{n^*} , i.e., by the VC dimension h_{n^*} . ©2000 Adapted by permission from Springer Nature: [34]

technique. The use of union bounds may cause loss of tightness in the final estimates. Techniques based on Rademacher's complexity result in tighter final estimates.

In the next sections, we investigate some upper bounds on the generalization error for Gaussian and sigmoidal OHL networks, derived under the hypothesis of using random sampling. These bounds will not be derived as consequences of Theorem 4.2 but exploiting techniques specifically developed for such networks.

4.3.4 Bounds on the Generalization Error for Gaussian OHL Networks

In [26], upper bounds on the generalization error were derived for OHL networks with Gaussian basis functions, estimating from above the two terms on the right-hand side of (4.34) for such networks. Let

$$\begin{aligned} \mathcal{R}_n &\triangleq \left\{ \gamma_n : \mathbb{R}^d \rightarrow \mathbb{R} \text{ such that } \gamma(\mathbf{x}, \mathbf{w}_n) = \sum_{i=1}^n c_i \exp\left(-\frac{\|\mathbf{x} - \boldsymbol{\tau}_i\|^2}{2\varsigma_i^2}\right), \right. \\ &\quad \left. \text{where } \sum_{i=1}^n |c_i| \leq M, \boldsymbol{\tau}_i \in \mathbb{R}^d, \varsigma_i \in \mathbb{R}^+, i = 1, \dots, n \right\}, n = 1, 2, \dots, \quad (4.62) \end{aligned}$$

where M is a positive real number, and \mathbf{w}_n is the usual free parameter vector. Note that the sets \mathcal{R}_n are particular cases of the sets \mathcal{A}_n defined by (2.2) and (4.16). Then, according to the terminology introduced in Sect. 4.2.2 the sets \mathcal{R}_n are hypothesis classes.

Let the sample set be given by (see (4.6))

$$\Sigma_L \triangleq \{[\mathbf{x}^{(l)}, y^{(l)} = g(\mathbf{x}^{(l)}) + \eta^{(l)}], l = 1, \dots, L\}, \quad (4.63)$$

where the pairs $(\mathbf{x}^{(l)}, y^{(l)})$ are drawn independently according to the probability density $p(\mathbf{x}, y)$, and the $\eta^{(l)}$ are independent realizations of the same random variables. Although the density $p(\mathbf{x}, y)$ is unknown, we assume that it is subject to certain constraints. In particular, we assume that $E(y|\mathbf{x}) = g(\mathbf{x})$ (hence $E(\eta|\mathbf{x}) = 0$), and that $E(y|\mathbf{x})$ (which is the regression function $\gamma^*(\mathbf{x})$) belongs to the concept class of functions defined in [26, p. 829]. We further assume that the conditional probability density $p(y|\mathbf{x})$ has compact support and that $g(\mathbf{x})$ has compact support, too.²

The next theorem from [26] provides an upper bound on the generalization error for OHL networks belonging to \mathcal{R}_n . For simplicity, to illustrate this kind of results, in the following theorem we consider functions to be approximated belonging to the Sobolev space $\mathcal{W}_1^s(\mathbb{R}^d)$ (see Definition 3.6 for the definition of Sobolev spaces). However, the estimate holds for a more general family of functional spaces; see [26] for details.

Theorem 4.3 *Let the I/O sample set Σ_L be given by (4.63), the Gaussian OHL networks (4.62) be used, $\gamma^* = \gamma^\circ \in \mathcal{W}_1^s(\mathbb{R}^d)$ with $s > d$ and s even, and γ^* belongs to the concept class of functions defined in [26, p. 829]. Then, with probability at least $1 - \delta$ for every $\delta \in (0, 1)$ the generalization error (4.23) is bounded as follows:*

$$\begin{aligned} \mathcal{E}_{n\Sigma_L}^{\text{gen}} &= \|\gamma^\circ - \gamma_{n\Sigma_L}^\circ\|_{\mathcal{L}_2(p)}^2 = \int_{\mathbb{R}^d} [\gamma^\circ(\mathbf{x}) - \gamma_{n\Sigma_L}^\circ(\mathbf{x})]^2 p(\mathbf{x}) d\mathbf{x} \\ &= O\left(\frac{1}{n}\right) + O\left(\left[\frac{n d \ln(nL) - \ln \delta}{L}\right]^{1/2}\right). \quad (4.64) \end{aligned}$$

□

The proof of Theorem 4.3 is quite complex; it is reported in [25] for a more general kind of concept classes \mathcal{S} . The particular case of \mathcal{S} equal to the Sobolev space $\mathcal{W}_1^s(\mathbb{R}^d)$

²Recall that the support of a function $f : X \rightarrow \mathbb{R}$, where X is a normed linear space, is defined as $\text{supp } f \triangleq \text{cl}(\{x \in X \mid f(x) \neq 0\})$, i.e., it is the closure in the norm of X of the set of points where $f \neq 0$.

with an even value of $s > d$ follows from [14, p. 9, point 2]. We give here only the final and very simple step. Claim D.10 in [25] states that with probability at least $1 - \delta$ the following bound holds (see (4.30)):

$$|R_{\text{emp}}(\gamma_n) - R(\gamma_n)| = O\left(\left[\frac{n d \ln(nL) - \ln \delta}{L}\right]^{1/2}\right), \quad \forall \gamma_n \in \mathcal{R}_n.$$

Then, from Proposition 4.1 we have

$$R(\gamma_{n\Sigma_L}^\circ) - R(\gamma_n^\circ) \leq 2\omega(L, n, \delta), \quad (4.65)$$

where

$$\omega(L, n, \delta) = O\left(\left[\frac{n d \ln(nL) - \ln \delta}{L}\right]^{1/2}\right). \quad (4.66)$$

In Sect.D.1 of [25] it is also shown that

$$\mathcal{E}_n^{\text{appr}} \leq \varepsilon(n) = O\left(\frac{1}{n}\right), \quad (4.67)$$

which looks like an upper bound independent of the dimension d . This seems to be in contrast with the results presented in Chap. 3 and addressed in the next section, whose bound has the form $O(c^2/n)$ with the constant c possibly dependent on d . However, there is no contradiction between the two results since the families to be approximated are different. As shown in [14, p. 9, point 2], the concept class \mathcal{S} corresponding to the Sobolev space $\mathcal{W}_1^s(\mathbb{R}^d)$ with an even value of $s > d$ is provided by a sufficient amount of smoothness to make the bound in (4.67) appear as independent of d .

Therefore, substituting (4.65) and (4.67) into (4.26) we obtain, with probability at least $1 - \delta$,

$$\mathcal{E}_{n\Sigma_L}^{\text{gen}} \leq \varepsilon(n) + 2\omega(L, n, \delta) \quad (4.68)$$

and the result stated in Theorem 4.3 follows. Comparing (4.34) with (4.68), it turns out that Theorem 4.3 is a special case of Proposition 4.2.

A number of important remarks can be made about the estimates provided by Theorem 4.3:

- **Dependence on d .** Besides the bound in (4.64), the dimension d is present in the hypotheses of Theorem 4.3 via the requirement that the function γ° to be estimated belongs to the Sobolev space $\mathcal{W}_1^s(\mathbb{R}^d)$ with $s > d$: the order of the integrable derivatives of γ is then required to increase with the dimension d .
- **Dependence on n .** Let δ , the dimension d , and the sample cardinality L be fixed. When the basis cardinality n goes to infinity, the upper bound on the approximation error goes to zero with order $O(1/n)$, whereas the upper bound on the estimation error goes to infinity with order $O([n \ln n]^{1/2})$.

- **Dependence on L .** Now, let δ , n , and d be fixed. When L goes to infinity, the upper bound on the approximation error obviously does not vary, whereas the upper bound on the estimation error goes to zero with order $O\left([(ln L)/L]^{1/2}\right)$.
- **Trade-off between n and L .** Summing up, Theorem 4.3 has the following interpretation for fixed values of δ and d . Upper Bound (4.67) guarantees that for $n \rightarrow \infty$ the approximation error goes to zero, independently of L . Instead, Upper Bound (4.66) implies the following. A necessary condition for the upper bound on the generalization error to converge to zero is that the sample cardinality L goes to infinity faster than the basis cardinality n in the OHL networks. As noted in [26, p. 830], if $L = n^r$ for some $r \in \mathbb{R}^+$, then

$$\lim_{n \rightarrow +\infty} \frac{n \ln(nL)}{L} = \lim_{n \rightarrow +\infty} \frac{n \ln n^{r+1}}{n^r} = (r+1) \lim_{n \rightarrow +\infty} \frac{\ln n}{n^{r-1}},$$

which, for $r > 1$, converges to zero. This, combined with (4.64), shows convergence in probability of $R(\gamma_n^\circ | \Sigma_L)$ to $R(\gamma_n^\circ)$, so Property 4.1 holds.

The upper bound in (4.64) implies that, for every fixed number L of samples, there is an optimal basis cardinality n (denote it by $n^*(L)$) that provides the best performance, that is, the minimum upper bound on the generalization error. Simple calculations show that, for a fixed value of δ , we have

$$n^*(L) \propto \left(\frac{L}{d}\right)^{1/3}. \quad (4.69)$$

For a fixed value of L , a Gaussian OHL network with a small basis cardinality n would provide a small estimation error but a large approximation error. On the other hand, if n is very large, one has a small approximation error but a large estimation error, since too many free parameters should be estimated by means of an insufficient number L of samples. In other words, the number n of Gaussian basis functions has to be sufficiently large to allow a good approximation but not too large, so that a good estimate for the values of the free parameters can be obtained by means of the L available samples.

Upper Bound (4.64) on the generalization error has the form

$$\frac{a}{n} + b \left[\frac{n d \ln(nL) - \ln \delta}{L} \right]^{1/2}, \quad (4.70)$$

where a and b are two (unknown) positive constants implicitly contained in the “big O” notation. Just to illustrate the behavior of the bound, (4.70) is shown in Fig. 4.7 (which reproduces Fig. 4 in [26]) for $a = 0.01$, $b = 0.0006$, $d = 5$, and $\delta = 0.01$.

The figure shows the bound as a function of the basis cardinality n and for different values of the number L of samples ($L = 50$, $L = 100$, and $L = 500$). It can be seen that the minimum of the curve moves toward the right for increasing values of L , i.e., the basis cardinality $n^*(L)$ in correspondence with which the upper bound on

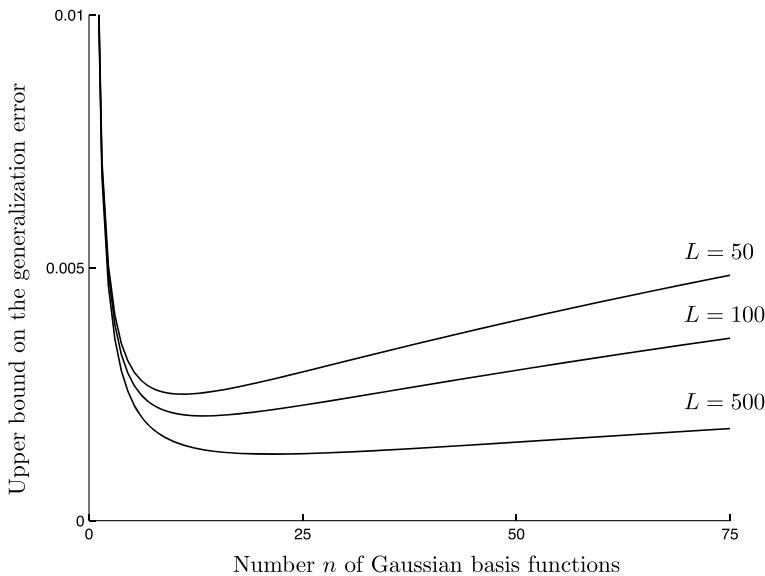


Fig. 4.7 Plot of Upper Bound (4.70) for $a = 0.01$, $b = 0.0006$, $d = 5$, and $\delta = 0.01$ as a function of the basis cardinality n in a Gaussian OHL network, for different numbers L of samples. ©1996 Adapted by permission from MIT Press: [26]

the generalization error is minimized increases when more samples are available. Moreover, the bigger the value of L , the flatter the curves in a neighborhood of $n^*(L)$. Then, the choice of the optimal basis cardinality is less critical when many samples are available.

Figure 4.8 (see Fig. 5 in [26]) shows that, for the same values of a , b , d , and δ mentioned above, there is an infinite number of choices of the pair (n, L) guaranteeing a desired value of the upper bound (4.70). The three choices correspond to the values 0.003, 0.004, and 0.005 in (4.70), respectively.

Finally, it has to be remarked that the above-reported discussion on the trade-off between the basis cardinality n and the sample cardinality L is based on an upper bound on the generalization error (the one provided by Theorem 4.3). As a consequence, the guidelines for the choices on n and L , illustrated above, follow by optimality arguments applied to estimates whose tightness is an open issue. In other words, knowing the exact dependencies of the generalization error on n and L might provide better values for the optimal choices of the basis and sample cardinalities.

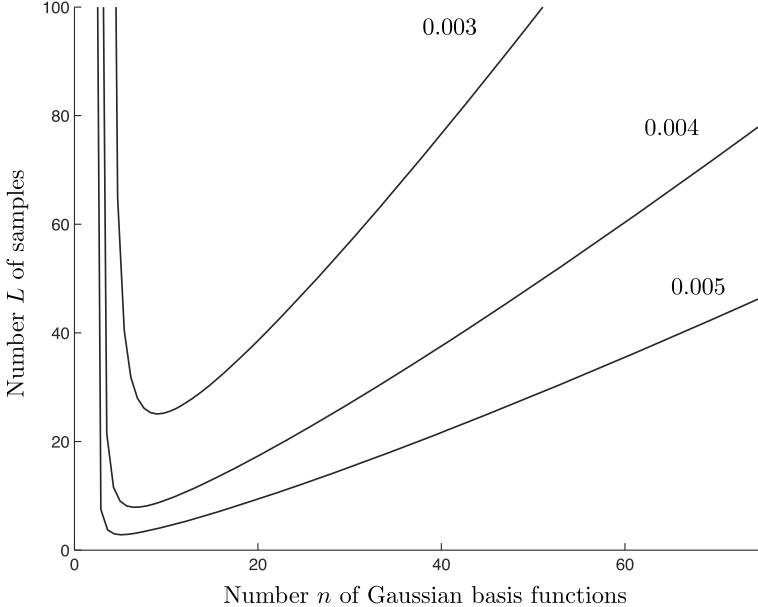


Fig. 4.8 The sample cardinality L is plotted as a function of the basis cardinality n in a Gaussian OHL network for three values of Upper Bound (4.70) (i.e., 0.003, 0.004, and 0.005), $a = 0.01$, $b = 0.0006$, $d = 5$, and $\delta = 0.01$. ©1996 Adapted by permission from MIT Press: [26]

4.3.5 Bounds on the Generalization Error for Sigmoidal OHL Networks

In this section, we give an upper bound on the rate of decrease of the generalization error as a function of n , L , and d , obtained in [3] for the case of sigmoidal OHL networks. These results were not derived from Theorem 4.2, but via probabilistic methods. For such networks, the rates of decrease of the estimation error are expressed in terms of the same smoothness properties that we used in Chap. 3.

The set of sigmoidal OHL networks with n basis functions (i.e., the hypothesis class) is given by

$$\begin{aligned} \mathcal{A}_n = \Upsilon_n &\triangleq \left\{ \gamma_n : \mathbb{R}^d \rightarrow \mathbb{R} \text{ such that} \right. \\ &\quad \left. \gamma(\mathbf{x}, \mathbf{w}_n) = \sum_{i=1}^n c_i \sigma(\mathbf{x}^\top \boldsymbol{\alpha}_i + \beta_i) + c_0 \right\}, \quad n = 1, 2, \dots, \end{aligned}$$

where $\boldsymbol{\alpha}_i \in \mathbb{R}^d$ and c_0, c_i, β_i ($i = 1, \dots, n$) $\in \mathbb{R}$. Such parameters make up the vector \mathbf{w}_n .

The functions $g : \mathbb{R}^d \rightarrow \mathbb{R}$ to be approximated belong to the set

$$\Gamma(\mathbb{R}^d) \triangleq \left\{ g : \mathbb{R}^d \rightarrow \mathbb{R} \text{ such that } \int_{\mathbb{R}^d} |\omega|_1 |\tilde{g}(\omega)| d\omega < \infty \right\}, \quad (4.71)$$

where \tilde{g} is the Fourier transform of g and $|\omega|_1 \triangleq \sum_{i=1}^d |\omega_i|$ is the l_1 -norm of ω

(ω_i denotes the i th component of the vector ω in \mathbb{R}^d). The L samples $(x^{(l)}, y^{(l)})$ are independently drawn according to the same probability density $p(x, y)$. We assume that $E(y|x) = g(x)$. Among other I/O relationships, the important case of the sample set (4.63) may occur. This includes the possibility that the values given by the function g are observed without errors at randomly selected points, i.e., $y^{(l)} = g(x^{(l)})$, $l = 1, \dots, L$ (see the I/O relationship (4.7)). Various probabilistic details should be added; we refer the reader to [3].

We denote by $\Theta_{n\Sigma_L} \subset \mathbb{R}^{\mathcal{N}(n)}$ a discrete subset of parameter values that satisfies certain technical assumptions (see [3] for details). For example, suitable conditions must be imposed on the fineness of the grid spacing that gives the set $\Theta_{n\Sigma_L}$. This is needed for the additional error, due to the discretization of the parameters of the OHL network, not to be large if compared with the approximation and estimation errors.

Finally, let

$$\begin{aligned} \bar{\gamma}_n &\triangleq \{ \gamma_n \in \gamma_n \text{ such that } w_n \in \Theta_{n\Sigma_L} \text{ and } \sigma \text{ are Lipschitz sigmoidal} \\ &\quad \text{basis functions that approach their limits } -1 \text{ at } -\infty \text{ and } 1 \text{ at} \\ &\quad +\infty \text{ at least polynomially fast} \}, \\ n &= 1, 2, \dots. \end{aligned} \quad (4.72)$$

The condition on the basis functions σ in (4.72) implies that $[-1 - \sigma(z)] \cdot |z|^p$ and $[1 - \sigma(z)] \cdot |z|^p$ remain bounded as $z \rightarrow -\infty$ and $z \rightarrow +\infty$, respectively, for some $p > 0$. This condition is introduced to obtain the result stated in the next theorem. We need the following technical assumption.

- Assumption 4.1** (i) The marginal density $p(x|y)$ has support $X \subseteq [-1, 1]^d$.
(ii) The magnitudes of the parameters of the OHL networks are suitably bounded (see [3, p. 123]) in terms of the Lipschitz constant of the sigmoidal basis function and the quantity $c_{\gamma}^d \triangleq \int_{\mathbb{R}^d} |\omega|_1 |\hat{\gamma}^{\circ}(\omega)| d\omega$.
(iii) The optimal network parameters are searched for within a grid of points whose coarseness satisfies a suitable relationship with respect to the above-defined quantity c_{γ}^d and the basis cardinality (see [3, formulas (19), (24), (25), and (26)]). \triangleleft

For a thorough discussion of this assumption, we refer the reader to [3].

The following theorem gives an upper bound on the expectation of the generalization error in approximating $\gamma^{\circ} \in \Gamma(\mathbb{R}^d)$ by means of sets Σ_L of I/O samples and OHL networks belonging to $\bar{\gamma}_n$. The theorem is stated for $X \subseteq [-1, 1]^d$ but can be extended to other bounded domains (see [3]).

Theorem 4.4 Let the I/O sample set Σ_L be given by (4.63), sigmoidal OHL networks belonging to $\tilde{\Upsilon}_n$ be used, $\gamma^\circ \in \Gamma(\mathbb{R}^d)$, $X \subseteq [-1, 1]^d$, and Assumption 4.1 be satisfied. Then, the mean value of the generalization error (4.23) is bounded as follows:

$$\begin{aligned} E(\mathcal{E}_{n\Sigma_L}^{\text{gen}}) &= E\left(\|\gamma^\circ - \gamma_{n\Sigma_L}^\circ\|_{\mathcal{L}_2(p)}^2\right) = E\left\{\int_X [\gamma^\circ(\mathbf{x}) - \gamma_{n\Sigma_L}^\circ(\mathbf{x})]^2 p(\mathbf{x}) d\mathbf{x}\right\} \\ &= O\left(\frac{1}{n}\right) + O\left(\frac{nd}{L} \ln L\right). \end{aligned} \quad (4.73)$$

□

It has to be remarked that the upper bound in Theorem 4.3 is expressed in probability (i.e., via the confidence interval associated with the probability $1 - \delta$), whereas the bound on the generalization error provided by Theorem 4.4 is formulated in mean value. Then, in the following, the two terms in the right-hand side of (4.73) will be interpreted as upper bounds, respectively, on the approximation error (which does not depend on $\gamma_{n\Sigma_L}^\circ$) and on the mean value of the estimation error, which in this specific case refers to the term $R(\gamma_{n\Sigma_L}^\circ) - R(\gamma_n^\circ)$ in (4.26).

The first term in the right-hand side of (4.73) follows from

$$\mathcal{E}_n^{\text{appr}} = O\left(\frac{1}{n}\right), \quad (4.74)$$

which is Maurey–Jones–Barron’s theorem (see Theorem 3.3).

The second term gives an upper bound on the mean value of the estimation error, obtained in [3] by means of statistical arguments. Then, we have

$$E(\mathcal{E}_{n\Sigma_L}^{\text{est}}) = O\left(\frac{nd}{L} \ln L\right). \quad (4.75)$$

Note that the bound (4.75) depends on the same variables d , L , and n as (4.66) but not on δ , since it refers to the expectation of the estimation error.

A number of remarks can be made about the estimate provided by Theorem 4.4. Compare with the corresponding remarks about the bounds provided by Theorem 4.3 for the Gaussian OHL networks.

- **Dependence on d .** In Theorem 4.3, the effect of the number d of variables is different than in Theorem 4.4. In the former, it is expressed by the belonging of γ° to a suitable subset of the Sobolev space $\mathcal{W}_1^s(\mathbb{R}^d)$ with $s > d$ even. In the latter, instead, it is required that $\gamma^\circ \in \Gamma(\mathbb{R}^d)$. As remarked in Sect. 3.8.4, the space $\Gamma(\mathbb{R}^d)$ is defined by a condition that becomes more and more constraining when the dimension d grows. Indeed, for a function $g : \mathbb{R}^d \rightarrow \mathbb{R}$ such that

$$c_g^d \triangleq \int_{\mathbb{R}^d} |\omega|_1 |\tilde{g}(\omega)| d\omega < \infty,$$

the quantity c_g^d may grow very fast with d (e.g., exponentially).

- **Dependence on n .** For fixed dimension d and sample cardinality L , when $n \rightarrow \infty$, the upper bound on the approximation error goes to zero with rate $O(1/n)$. The upper bound on the mean value of the estimation error goes to ∞ with rate $O(n)$.
- **Dependence on L .** Let d and n be fixed. When the sample cardinality L goes to ∞ , the upper bound on the approximation error obviously does not vary, whereas the upper bound on the mean value of the estimation error goes to zero with rate $O(\ln L/L)$. The rate $O(\ln L/L)$ has to be compared with $O\left([(\ln L)/L]^{1/2}\right)$ for Gaussian OHL networks. Although the difference between the two orders is important, it has to be remarked that the dependencies on L of the upper bounds on the estimation error (respectively, on its mean value) for the two kinds of OHL networks have been estimated under different hypotheses and by exploiting different proof techniques.
- **Trade-off between n and L .** Like for Gaussian OHL networks, for a given number d of variables, there is a trade-off between the basis cardinality n and the sample cardinality L . The right-hand side of Bound (4.73) takes on the form

$$\frac{a}{n} + b \frac{n d}{L} \ln L , \quad (4.76)$$

where a and b are (unknown) constants implicitly contained in the “big O” notation. Likewise in Fig. 4.7, the upper bound (4.76) is shown in Fig. 4.9 for $a = 0.01$, $b = 0.0006$, and $d = 5$. The figure shows the bound as a function of the basis cardinality for different values of the sample cardinality L ($L = 50$, $L = 100$, and $L = 500$). For a fixed value of d , we have the following interpretation. Upper Bound (4.74) implies that for $n \rightarrow \infty$ the approximation error goes to zero independently of L , as one expects. Instead, according to Upper Bound (4.75), a necessary condition for the convergence to zero of the upper bound on the mean value of the estimation error is that the basis cardinality n grows slower than $L/\ln L$. To estimate via Theorem 4.4 the optimal basis cardinality $n^*(L)$ corresponding to a sample cardinality L , we proceed as we did for Gaussian OHL networks via Theorem 4.3. As we are interested just in the kind of dependence of the optimal basis cardinality on L and not in the exact expression of the function $n^*(L)$, for simplicity we set the constants a and b in (4.76) equal to 1, thus getting

$$\frac{1}{n} + \frac{n d}{L} \ln L . \quad (4.77)$$

Minimization of (4.77) with respect to n yields

$$n^*(L) \propto \left(\frac{L}{d \ln L} \right)^{1/2} . \quad (4.78)$$

Substituting (4.78) into (4.77), the bound becomes

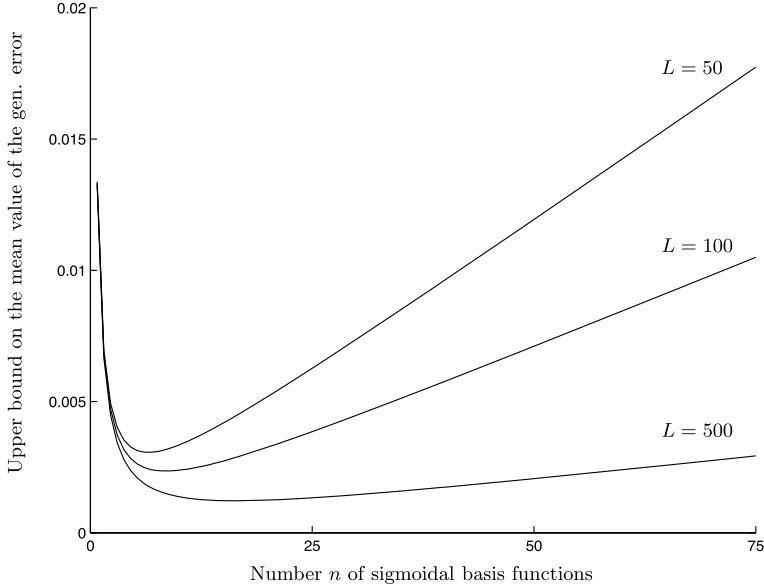


Fig. 4.9 Plot of Upper Bound (4.77) for $a = 0.01$, $b = 0.0006$, and $d = 5$, as a function of the basis cardinality n in a sigmoidal OHL network for different values of the sample cardinality L

$$\mathbb{E}(\mathcal{E}_{n \Sigma_L}^{\text{gen}}) = O\left(\left[\frac{d}{L} \ln L\right]^{1/2}\right). \quad (4.79)$$

Remarks similar to those made at the end of Sect. 4.3.4 apply here about the fact that the above-reported discussion is based on upper bounds whose tightness is an open issue.

In [22], the authors considered the particular case of sigmoidal OHL networks using the *cosine squasher* as activation function. This function is defined as

$$\psi(x) \triangleq \begin{cases} 0, & x \leq -\pi/2, \\ [\cos(x + 3\pi/2) + 1]/2, & -\pi/2 \leq x \leq \pi/2, \\ 1, & x \geq \pi/2. \end{cases}$$

Let some suitable technical assumption on the magnitudes of the parameters of the corresponding OHL networks be verified (see [22] for details) and $\gamma^\circ \in \mathcal{W}_2^s(\mathbb{R}^d)$ with $s \geq d$. Assume also that, for any given sample cardinality L , the following basis cardinality is chosen (compare with (4.69) and (4.78)):

$$n^*(L) = \left\lceil \frac{d+1}{2s+d+5} \right\rceil. \quad (4.80)$$

Relationship (4.80) is derived by equating the order of the approximation error to the order of a bound on the estimation error in terms of metric entropy (see [22, p.153]). Then, under the assumptions mentioned above, [22, Corollary 1] provides the following upper bound in probability on the generalization error for sigmoidal OHL networks with the cosine squasher as activation function: for every $\varepsilon > 0$ there exists $M_\varepsilon > 0$ such that

$$P \left\{ \frac{\mathcal{E}_{n\Sigma_L}^{\text{gen}}}{L - \frac{2s}{2s+d+5} + \varepsilon} > M_\varepsilon \right\} < \varepsilon. \quad (4.81)$$

When the smoothness of the Sobolev space $\mathcal{W}_2^s(\mathbb{R}^d)$ is such that $\gamma^\circ \in \Gamma(\mathbb{R}^d)$, (4.81) implies a bound on the generalization error in the same space $\Gamma(\mathbb{R}^d)$ considered in Theorem 4.4. Estimate (4.81) has some pros and cons with respect to the bound (4.73) from Theorem 4.4. An advantage is that (4.81) does not require that the optimal network parameters are searched for within a suitable grid of points (see Assumption 4.1(iii)) as remarked in [3, Remark at p. 132]. However, whereas (4.73) holds in mean value, Bound (4.81) is expressed in probability, and the special sigmoidal basis function given by the cosine squasher has to be used.

We conclude this section by comparing the upper bound stated in Theorem 4.4 with available bounds for LCFBFs. For sigmoidal OHL networks with basis cardinality given by (4.78), ignoring the logarithmic factor, Bound (4.79) becomes $E(\mathcal{E}_{n\Sigma_L}^{\text{gen}}) = O((d/L)^{1/2})$. Let us now suppose that functions belonging to classical smoothness families, such as balls in Sobolev spaces of order s , have to be estimated by using LCFBFs, like fixed-center RBFs, fixed-knot splines, polynomials, etc. Then, it can be shown (see, e.g., [27]) that the corresponding mean value of the generalization error can be bounded from above by $O(1/L^{2s/(2s+d)})$. The dependence of the exponent on the dimension d implies that, in order to keep the mean value of the generalization error below a fixed threshold, the smoothness index s has to grow linearly with d . In other words, when LCFBFs are used, in order to avoid the curse of dimensionality with respect to the sample cardinality L , the family of functions to be estimated has to be more and more constrained as the number of variables d increases.

Finally, note that, in the bounds on the approximation errors in (4.73), the dependence on the dimension d is “hidden” in the condition $\gamma^\circ \in \Gamma(\mathbb{R}^d)$. As already remarked, the space $\Gamma(\mathbb{R}^d)$ is defined by a requirement (see (4.71)) that becomes more and more constraining when d grows. The other way round, the quantity $\int_{\mathbb{R}^d} |\omega|_1 |\tilde{\gamma}^\circ(\omega)| d\omega$ may grow very fast with d . Note that Estimate (4.81) holds for $\gamma^\circ \in \mathcal{W}_2^s(\mathbb{R}^d)$ with $s \geq d$, so even in this case the dependence on d is contained in the hypothesis on γ° .

4.4 Deterministic Learning Theory

In SLT, the available I/O samples are viewed as realizations of random variables, generated by a signal source on the basis of a probability density function $p(\mathbf{x}, y)$. Learning from data is formalized in terms of the optimal solution of an IDO problem as the minimization of Expected Risk (4.8) or, in terms of the optimal solution of an FDO problem (i.e., of an NLP problem), of Expected Risk (4.36). However, as pointed out in Sect. 4.1, in general $p(\mathbf{x}, y)$ is unknown. Then, since the expected risk cannot be minimized directly, one applies the ERM principle.

However, if one has the possibility of generating the input samples $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(L)}$ by a deterministic algorithm and if the I/O relationship can be described by Mapping (4.7), then we are in a completely deterministic learning framework and (as we said at the beginning of the chapter) the term “deterministic learning theory” sounds appropriate (see also Fig. 4.2). There are at least two good reasons for considering DLT: (1) the upper bounds eventually derived via this approach are deterministic instead of just probabilistic as in SLT, and (2) choosing “good” samples might give better results than those provided by SLT. DLT exploits tools developed for numerical integration (see, e.g., [8, 11, 24]). Methods based on this approach generate a deterministic sequence of points trying to “spread” them in the most uniform way.

If one has the possibility to generate a sequence of deterministic input samples but the I/O relationship is stochastic, then the learning framework has to draw concepts from both SLT and DLT.

In this section, we mainly consider the completely deterministic context. In particular, the estimation of functions without noise, addressed in the next sections, will be useful in Chaps. 6 and 7, where we shall approximate the optimal cost-to-go functions by applying approximate dynamic programming. The important problem of estimating functions in the presence of additive noise will be considered in Sect. 4.4.9.

4.4.1 Estimation of Functions Without Noise: The Expected Risk, the Empirical Risk, and the Generalization Error

In this and in the following sections, we consider the case of a deterministic I/O relationship given by the unknown function introduced in Sect. 4.1, i.e.,

$$y = g(\mathbf{x}) . \quad (4.82)$$

Let us consider Expected Risk (4.8). As said at the beginning of Sect. 4.1, the deterministic I/O relationship corresponds to the limit case in which the conditional density is the delta generalized function $p(y|\mathbf{x}) = \delta[y - g(\mathbf{x})]$. Then, (4.8) becomes

$$R(\gamma) = \int_X [y - \gamma(\mathbf{x})]^2 p(\mathbf{x}) d\mathbf{x} . \quad (4.83)$$

We know that the input \mathbf{x} is generated by a deterministic algorithm in order to make up a data set Σ_L and estimate the function g . Ideally, this algorithm should run for $L \rightarrow \infty$ spreading the samples $\mathbf{x}^{(l)}$ on X as uniformly as possible. In this way, Σ_L aims at approximating a set of i.i.d. samples with a uniform distribution. Then, assuming for simplicity of notation that X has measure 1, it is reasonable to employ an expected risk having the form

$$R(\gamma) = \int_X [y - \gamma(\mathbf{x})]^2 d\mathbf{x}. \quad (4.84)$$

In a deterministic context, the adjective “expected” is not appropriate but, to make the comparison between SLT and DLT easier, we continue to use this term.

Under the assumption that the function g belongs to the set \mathcal{S} of the admissible estimators, the optimal estimator turns out to be $\gamma^\circ = g$, hence $R(\gamma^\circ) = 0$.

As regards the empirical risk, we give the same definition as in SLT (see (4.15)), that is,

$$R_{\text{emp}}(\gamma, \Sigma_L) = \frac{1}{L} \sum_{l=1}^L [y^{(l)} - \gamma(\mathbf{x}^{(l)})]^2. \quad (4.85)$$

Following the same notation as in Sect. 4.1, let γ_n° and $\gamma_{n\Sigma_L}^\circ$ be the FSP functions that minimize the expected risk and the empirical risk over the set \mathcal{A}_n , respectively. We obtain (see (4.23) and (4.24) with $R(\gamma^\circ) = 0$)

$$\mathcal{E}_{n\Sigma_L}^{\text{gen}} = \int_X [\gamma^\circ(\mathbf{x}) - \gamma(\mathbf{x}, \mathbf{w}_{n\Sigma_L}^\circ)]^2 d\mathbf{x} = R(\gamma_{n\Sigma_L}^\circ), \quad (4.86)$$

and

$$\mathcal{E}_n^{\text{appr}} = \int_X [\gamma^\circ(\mathbf{x}) - \gamma_n^\circ(\mathbf{x})]^2 d\mathbf{x} = R(\gamma_n^\circ). \quad (4.87)$$

Likewise in (4.28), we also define the estimation error as follows:

$$\mathcal{E}_{n\Sigma_L}^{\text{est}} \triangleq |R_{\text{emp}}(\gamma_{n\Sigma_L}^\circ) - R(\gamma_{n\Sigma_L}^\circ)|. \quad (4.88)$$

Now, we decompose the generalization error as in (4.26), i.e.,

$$\mathcal{E}_{n\Sigma_L}^{\text{gen}} = R(\gamma_{n\Sigma_L}^\circ) - R(\gamma_n^\circ) + R(\gamma_n^\circ) = \mathcal{E}_n^{\text{appr}} + R(\gamma_{n\Sigma_L}^\circ) - R(\gamma_n^\circ). \quad (4.89)$$

Concerning the approximation error, we can determine an upper bound $\varepsilon(n)$ on it as in (4.25) by using the concepts presented in Chap. 3. Only the estimation error has to be dealt with in a different way when switching from SLT to DLT. Therefore, in the remaining part of this section, we focus on $\mathcal{E}_{n\Sigma_L}^{\text{est}}$.

4.4.2 Deterministic Convergence of the Empirical Risk

The same reasons of generality, set out at the beginning of Sect. 4.3.1, lead us to express Risks (4.84) and (4.85) in the forms

$$R^v(\mathbf{w}_n) = \int_X v[y, \gamma(\mathbf{x}, \mathbf{w}_n)] d\mathbf{x} \quad (4.90)$$

and

$$R_{\text{emp}}^v(\mathbf{w}_n, \Sigma_L) = \frac{1}{L} \sum_{l=1}^L v[y^{(l)}, \gamma(\mathbf{x}^{(l)}, \mathbf{w}_n)] \quad (4.91)$$

(see (4.36) and (4.37), respectively). Such a generality is not lost if we go on using, as estimators, some family of OHL networks $\gamma(\cdot, \mathbf{w}_n)$ with a given basis cardinality n .

The convergence in probability of the empirical risk defined in (4.42) is now replaced by the following deterministic condition:

$$\lim_{L \rightarrow \infty} \sup_{\mathbf{w}_n \in W_n} |R_{\text{emp}}^v(\mathbf{w}_n, \Sigma_L) - R^v(\mathbf{w}_n)| = 0, \quad (4.92)$$

where the sample set (see (4.1) and (4.2))

$$X_L = \{\mathbf{x}^L\} = \{\mathbf{x}^{(l)}, l = 1, \dots, L\}$$

is generated by a deterministic algorithm. Then, we can state the following result which, as regards the conclusion, shows similarities with Proposition 4.3:

Theorem 4.5 ([4, Theorem 2]) *If the sequence $\{\Sigma_L\}_{L=1}^\infty$ (generated by a deterministic algorithm) satisfies Condition (4.92), then Property 4.2 (see Sect. 4.2.2) holds true.* \square

Proof For any $\varepsilon > 0$, Condition (4.92) enables one to choose an \bar{L} such that, for any $L \geq \bar{L}$,

$$|R_{\text{emp}}^v(\mathbf{w}_n, \Sigma_L) - R^v(\mathbf{w}_n)| \leq \frac{\varepsilon}{2}, \quad \forall \mathbf{w}_n \in W_n.$$

Then, we let $k = \frac{\varepsilon}{2}$ in (4.30). From Proposition 4.1 (which obviously holds also for Definitions (4.90) and (4.91)), we obtain (see (4.31))

$$R^v(\mathbf{w}_{n\Sigma_L}) - R^v(\mathbf{w}_n^\circ) \leq \varepsilon, \quad \forall L \geq \bar{L}.$$

Thus, Property 4.2 follows. \blacksquare

4.4.3 Discrepancy of a Sequence

Following [24], we shall assume that the input domain X is given by the d -dimensional closed unit cube $[0, 1]^d$; so, the sample set X_L is a subset of $[0, 1]^d$ of finite cardinality. The results can be extended to more complex input sets by means of suitable transformations. In particular, one can consider spheres and other compact convex domains such as simplexes [11].

Let us denote by χ_E the characteristic function of E , that is,

$$\chi_E(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x} \in E \\ 0 & \text{otherwise} \end{cases},$$

where E is an arbitrary subset in $[0, 1]^d$. Denote also by $\#(E, X_L)$ the number of points of X_L that belong to E . Then, we can write

$$\#(E, X_L) = \sum_{l=1}^L \chi_E(\mathbf{x}^{(l)}).$$

The next two definitions introduce the concept of discrepancy of a sample set or a sample sequence in $[0, 1]^d$.

Definition 4.3 Let \mathcal{B} be a non-empty family of Lebesgue-measurable subsets in $[0, 1]^d$. The “discrepancy” of the sample set X_L (or the sequence $\{\mathbf{x}^L\}$) is defined as

$$D_L(\mathcal{B}, X_L) \triangleq \sup_{E \in \mathcal{B}} \left| \frac{\#(E, X_L)}{L} - \lambda(E) \right|, \quad (4.93)$$

where $\lambda(E)$ denotes the Lebesgue measure of E . □

Note that it is always

$$0 \leq D_L(\mathcal{B}, X_L) \leq 1.$$

By considering the special case of \mathcal{B} being composed of all subintervals of $[0, 1]^d$, we obtain the following notion of “star discrepancy.”

Definition 4.4 The “star discrepancy” $D_L^*(X_L)$ of the sample set X_L is defined by (4.93), where \mathcal{B} is the family of all the subintervals of $[0, 1]^d$ of the form $\prod_{i=1}^d [0, b_i]$. △

As noted in [24, Remark 2.3], in the definition of “star discrepancy” one can replace $\prod_{i=1}^d [0, b_i]$ with $\prod_{i=1}^d [0, b_i]$ when all the points of the sample set X_L belong to $\prod_{i=1}^d [0, 1]$. This occurs in most cases of practical interest. Then, in the sequel, we shall consider the closed cube $[0, 1]^d$. By the way, note that the differences between the sets $[0, 1]^d$ and $[0, 1]^d$ have Lebesgue measure equal to zero. As from now on we shall integrate on these domains, such a difference does not influence the results.

A classic result [21] states that the following properties are equivalent:

1. $\lim_{L \rightarrow \infty} \frac{\#(E, X_L)}{L} = \lambda(E)$ for all subintervals E of $[0, 1]^d$ of the form $\prod_{i=1}^d [a_i, b_i]$;
2. $\lim_{L \rightarrow \infty} D_L^*(X_L) = 0$.

Thus, the smaller the discrepancy or the star discrepancy, the more uniformly distributed the sample set X_L in $[0, 1]^d$ or in $[0, 1)^d$.

4.4.4 Variation of a Function and Conditions for Bounded Variations

Let us now introduce the concept of variation of a multivariable function. Let us consider the vertices V_k ($k = 1, \dots, 2^d$) of a given subinterval

$$E \triangleq \prod_{i=1}^d [a_i, b_i]$$

of $[0, 1]^d$. Each vertex V_k is described by the vector $x = \text{col } (v_1, \dots, v_d)$. The component v_i takes the value a_i or b_i (see Fig. 4.10). Let us associate each vertex with a binary string (made of d bits) whose i th bit is 0 if $v_i = a_i$ and 1 if $v_i = b_i$. Let also e_E and o_E denote the set of vertices having an even and an odd number of “1”s in their strings, respectively.

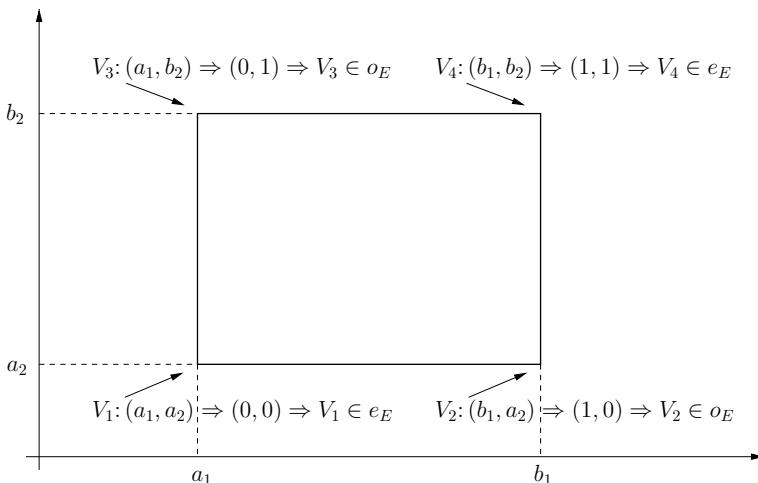


Fig. 4.10 Example of a subinterval $E \triangleq \prod_{i=1}^2 [a_i, b_i]$

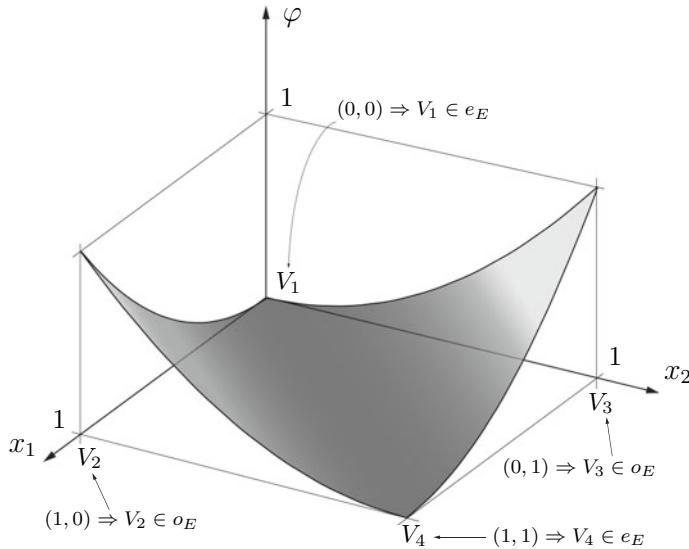


Fig. 4.11 Example with the function $\varphi(x_1, x_2) = (x_1 - x_2)^2$ and $E = [0, 1]^2$. For each of the four vertices of $[0, 1]^2$ (see Fig. 4.10), the binary strings and the belonging to the sets e_E and o_E are shown

Given a function

$$\varphi : [0, 1]^d \rightarrow \mathbb{R}, \quad (4.94)$$

we let

$$\Delta(\varphi, E) \triangleq \sum_{\mathbf{x} \in e_E} \varphi(\mathbf{x}) - \sum_{\mathbf{x} \in o_E} \varphi(\mathbf{x}). \quad (4.95)$$

Then, $\Delta(\varphi, E)$ is the alternating sum of the values of the function φ computed at the vertices of E (i.e., in the sum, the values of φ at adjacent vertices of E are multiplied by opposite unit weights). See the example of Fig. 4.11. The following definition can now be given.

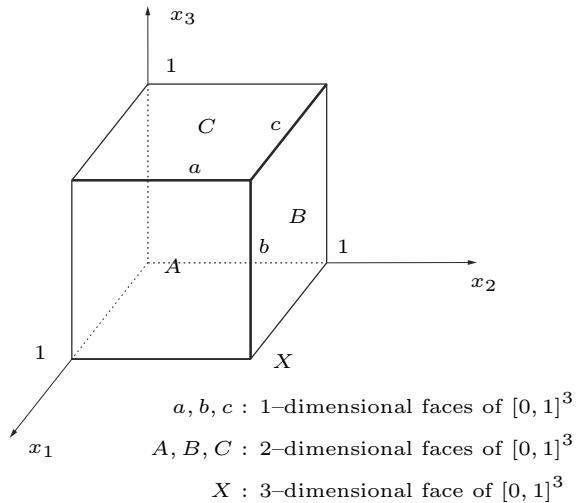
Definition 4.5 The “variation of φ on $[0, 1]^d$ in the sense of Vitali” is defined as

$$V^{(d)}(\varphi) \triangleq \sup_{\wp} \sum_{E \in \wp} |\Delta(\varphi, E)|, \quad (4.96)$$

where \wp is any partition of $[0, 1]^d$ into subintervals. \triangleleft

If the partial derivatives of φ up to the order d are continuous on $[0, 1]^d$, then a convenient formula to compute (4.96) (with respect to computations based on (4.96)) is given by

Fig. 4.12 One-, two-, and three-dimensional faces of the unit cube $[0, 1]^3$



$$V^{(d)}(\varphi) = \int_0^1 \cdots \int_0^1 \left| \frac{\partial^d \varphi}{\partial x_1 \cdots \partial x_d} \right| dx_1 \cdots dx_d, \quad (4.97)$$

where x_i is the i th component of \mathbf{x} . Hence, in such a case, $V^{(d)}(\varphi)$ is finite. A proof of (4.97) can be found in [28].

Referring to the example in Fig. 4.11 and applying Formula (4.96), we obtain the variation of the function $\varphi(x_1, x_2) = (x_1 - x_2)^2$ on $[0, 1]^2$ in the sense of Vitali, that is,

$$\int_0^1 \int_0^1 \left| \frac{\partial(x_1 - x_2)^2}{\partial x_1 \partial x_2} \right| dx_1 dx_2 = 2.$$

Definition 4.5 is also used to introduce the following different notion of variation [24]. For a function (4.94) and positive integers $1 \leq k \leq d$ and $1 \leq i_1 < i_2 < \cdots < i_k \leq d$, let

$$V^{(k)}(\varphi, i_1, \dots, i_k)$$

be the variation in the sense of Vitali of the *restriction of φ to the k -dimensional face*

$$\{(x_1, \dots, x_d) \in [0, 1]^d : x_i = 1 \text{ for } i \neq i_1, \dots, i_k\}.$$

The one-, two-, and three-dimensional faces of the unit cube $[0, 1]^3$ are shown in Fig. 4.12.

Definition 4.6 The “variation of φ on $[0, 1]^d$ in the sense of Hardy and Krause” is defined as

$$V_{HK}(\varphi) \triangleq \sum_{k=1}^d \sum_{1 \leq i_1 < i_2 < \dots < i_k \leq d} V^{(k)}(\varphi, i_1, \dots, i_k). \quad (4.98)$$

△

Note that the three-dimensional face X of $[0, 1]^3$ is given by the unit cube itself.

In general, boundedness of $V_{HK}(\varphi)$ is attained if the function φ is sufficiently smooth. In this connection, from (4.97) we see that, if φ has continuous partial derivatives up to the order d , then $V_{HK}(\varphi)$ is finite. More precisely, for a positive scalar M and a positive integer d , we define the class of functions

$$\begin{aligned} \mathcal{W}_M([0, 1]^d) = \{f : [0, 1]^d \rightarrow \mathbb{R} \text{ such that } \partial_{i_1, \dots, i_k} f \text{ is continuous on } [0, 1]^d \\ \text{and} \\ |\partial_{i_1, \dots, i_k} f| \leq M \text{ for } 1 \leq k \leq d \text{ and } 1 \leq i_1 \leq \dots \leq i_k \leq d\}, \end{aligned} \quad (4.99)$$

where

$$\partial_{i_1, \dots, i_k} \triangleq \frac{\partial^k}{\partial x_{i_1} \cdots \partial x_{i_k}}.$$

Then, from (4.97), functions in $\mathcal{W}_M([0, 1]^d)$ have bounded variation in the sense of Hardy and Krause. Moreover, for a finite positive scalar M and two positive integers d and r , $d < r$, we define the class of functions

$$\begin{aligned} \mathcal{W}_{M,d}([0, 1]^r) = \{f : [0, 1]^r \rightarrow \mathbb{R} \text{ such that } \partial_{i_1, \dots, i_k} f \text{ is continuous on } [0, 1]^r \\ \text{and} \\ |\partial_{i_1, \dots, i_k} f| \leq M \text{ for } 1 \leq k \leq d \text{ and } 1 \leq i_1 \leq \dots \leq i_k \leq r\}. \end{aligned}$$

More generally, we use the notations $\mathcal{W}_M(\prod_j H_j)$ and $\mathcal{W}_{M,d}(\prod_j H_j)$ when the sets $[0, 1]^d$ and $[0, 1]^r$ are replaced by the box $\prod_j H_j$, where the H_j are closed and bounded intervals, for a finite number of indices j .

The following lemma from [4] gives sufficient conditions for the boundedness of the variation of a composite function. The lemma and its corollary will be exploited later on in the next section and in Sect. 6.6.2 where we apply DLT to the numerical solution of DP.

Lemma 4.1 Consider the function $\psi : \mathbb{R}^r \rightarrow \mathbb{R}$ defined by the composition $\psi[\mu_1(\xi_1, \dots, \xi_d), \dots, \mu_r(\xi_1, \dots, \xi_d)]$, where $\mu_j : [0, 1]^d \rightarrow H_j \subset \mathbb{R}$ for $1 \leq j \leq r$, and suppose that

1. $\psi \in \mathcal{W}_{M,d}(\prod_j H_j)$;
2. $\mu_j \in \mathcal{W}_M([0, 1]^d)$ for all $1 \leq j \leq r$.

Then, ψ has bounded variation in the sense of Hardy and Krause on $[0, 1]^d$. □

Two results follow directly from Lemma 4.1, and they will be useful in other parts of the book. They are collected in Corollary 4.1.

Corollary 4.1 *Consider two functions $\mu_1 : [0, 1]^d \rightarrow H_1 \subset \mathbb{R}$ and $\mu_2 : [0, 1]^d \rightarrow H_2 \subset \mathbb{R}$, where H_1 and H_2 are closed and bounded intervals. If $\mu_1 \in \mathcal{W}_M([0, 1]^d)$, $\mu_2 \in \mathcal{W}_M([0, 1]^d)$, then both $\mu_1 + \mu_2$ and $\mu_1 \mu_2$ have bounded variation in the sense of Hardy and Krause on $[0, 1]^d$.* \square

4.4.5 Examples of Functions with Bounded Hardy and Krause Variation

We give now five examples (see [4]) of functions in $\mathcal{W}_M([0, 1]^d)$. Their variations in the sense of Hardy and Krause are computed via (4.97).

$$1. \quad \varphi_1(\mathbf{x}) = c + \sum_{i=1}^d \alpha_i x_i, \quad \alpha_i \in \mathbb{R}.$$

For $k > 1$, all terms $V^{(k)}(\varphi, i_1, \dots, i_k)$ are equal to 0. Therefore, we have

$$V_{HK}(\varphi_1) = \sum_{i=1}^d \int_0^1 |\alpha_i| dx_i = \sum_{i=1}^d |\alpha_i|.$$

$$2. \quad \varphi_2(\mathbf{x}) = \prod_{i=1}^d x_i.$$

As every term $V^{(k)}(\varphi, i_1, \dots, i_k)$ is equal to 1, we get

$$V_{HK}(\varphi_2) = \sum_{k=1}^d \sum_{1 \leq i_1 < i_2 < \dots < i_k \leq d} \int_0^1 dx_i = \sum_{k=1}^d \binom{d}{k} = 2^d - 1.$$

$$3. \quad \varphi_3(\mathbf{x}) = \prod_{i=1}^d (1 - x_i).$$

In this case, the restriction of φ to the k -dimensional face is equal to 0 for each $k < d$. Therefore, we have only the term $V^{(d)}$ in (4.98) and so

$$V_{HK}(\varphi_3) = \int_0^1 dx_i = 1.$$

Examples 2 and 3 show that quite “similar” functions can give completely different values for the variation in the sense of Hardy and Krause. It has to be remarked that computing $V_{HK}(\varphi)$ is in general a very hard task. However, in the case of functions with continuous partial derivatives up to a suitable order, the computation of upper bounds on their variations may be much simpler. This is the case of common radial and ridge OHL networks. For such networks, we report (see [4]) the following estimates, which will be used in Sect. 6.6.2.

4. Sigmoidal OHL Networks.

From

$$\varphi_4(\mathbf{x}) \triangleq \gamma(\mathbf{x}, \mathbf{w}_n) = \sum_{i=1}^n c_i \sigma(\mathbf{x}^\top \alpha_i + \beta_i) + c = \sum_{i=1}^n c_i \sigma\left(\sum_{j=1}^d \alpha_{ij} x_j + \beta_i\right) + c,$$

we have

$$\partial_{i_1, \dots, i_k} \gamma_n = \sum_{i=1}^n c_i \sigma^{(k)}\left(\sum_{j=1}^d \alpha_{ij} x_j + \beta_i\right) \prod_{1 \leq h \leq k} \alpha_{ij_h},$$

where $\sigma^{(k)}$ is the k th derivative of σ . If we let $\bar{\sigma}^{(k)} \triangleq \sup_{z \in \mathbb{R}} |\sigma^{(k)}(z)|$ (supposed to be finite), we obtain

$$|\partial_{i_1, \dots, i_k} \gamma_n| \leq \sum_{i=1}^n |c_i| \bar{\sigma}^{(k)} \prod_{1 \leq h \leq k} |\alpha_{ij_h}|. \quad (4.100)$$

Thus, if the parameters c_i and α_{ij_h} and the value $\bar{\sigma}^{(k)}$ are finite, then (4.97) implies that the terms $V^{(k)}(\gamma_n, i_1, \dots, i_k)$ are finite, too. Hence, by (4.98) we conclude that $V_{HK}(\gamma_n)$ is also finite. Note that for commonly used sigmoidal activation functions we have $\bar{\sigma}^{(k)} = 1$.

5. Radial Basis Function OHL Networks.

From

$$\begin{aligned} \varphi_5(\mathbf{x}) &\triangleq \gamma(\mathbf{x}, \mathbf{w}_n) = \sum_{i=1}^n c_i \exp[-\|\mathbf{x} - \mathbf{t}_i\|^2 / (2s_i^2)] + c \\ &= \sum_{i=1}^n c_i \exp\left[-\sum_{j=1}^d (x_j - t_{ij})^2 / (2s_i^2)\right] + c, \end{aligned}$$

we have

$$\partial_{i_1, \dots, i_k} \gamma_n = \sum_{i=1}^n (-s_i^{-2})^k c_i \exp\left[-\sum_{j=1}^d (x_j - t_{ij})^2 / (2s_i^2)\right] \prod_{1 \leq h \leq k} (x_{j_h} - t_{ij_h}).$$

Thus, we obtain

$$|\partial_{i_1, \dots, i_k} \gamma_n| \leq \sum_{i=1}^n |s_i^{-2k}| |c_i| \prod_{1 \leq h \leq k} |x_{j_h} - t_{ij_h}|.$$

If the parameters c_i and t_{ij_h} are bounded, then the boundedness of $V_{HK}(\gamma_n)$ directly follows. Usually, the centroids t_i are in $[0, 1]^d$, hence $t_{ij} \in [0, 1]$ and we obtain

$$|\partial_{i_1, \dots, i_k} \gamma_n| \leq \sum_{i=1}^n |s_i^{-2k}| |c_i|. \quad (4.101)$$

4.4.6 The Koksma–Hlawka Inequality

The fundamental result by Hlawka [19], stated in the next theorem (see also, e.g., [24, p. 20]), provides, for a function φ of bounded variation in the sense of Hardy and Krause, a relationship between the star discrepancy of the sample sequence $\{\mathbf{x}^L\}$ and the accuracy of the numerical integration of φ . In Sects. 5.1 and 5.2, we shall address in detail the issue of the computation of integrals and expected values. What interests us now is the use of this result in the general context of DLT.

Theorem 4.6 (Koksma–Hlawka (KH) inequality) *If φ has bounded variation $V_{HK}(\varphi)$ on $[0, 1]^d$ in the sense of Hardy and Krause, then the following holds for any sequence $\{\mathbf{x}^L\} \in [0, 1]^d$:*

$$\left| \frac{1}{L} \sum_{l=1}^L \varphi(\mathbf{x}^{(l)}) - \int_{[0,1]^d} \varphi(\mathbf{x}) d\mathbf{x} \right| \leq V_{HK}(\varphi) D_L^*(\{\mathbf{x}^L\}). \quad (4.102)$$

□

The KH inequality enables us to state conditions such that Limit (4.92) is equal to 0; hence, by virtue of Theorem 4.5, Property 4.2 (see Sect. 4.2.2) holds. To get this, the following assumptions are needed.

Assumption 4.2 The sequence $\{\mathbf{x}^L\}$ is such that

$$\lim_{L \rightarrow \infty} D_L^*(\{\mathbf{x}^L\}) = 0.$$

△

Let the function $\varphi(\mathbf{x})$ in (4.102) be given by the loss function

$$\varphi(\mathbf{x}, \mathbf{w}_n) \triangleq v[g(\mathbf{x}), \gamma(\mathbf{x}, \mathbf{w}_n)]. \quad (4.103)$$

Then, the first term in the left-hand-side of Inequality (4.102) turns out to be the empirical risk $R_{\text{emp}}^v(\mathbf{w}_n, \Sigma_L)$ (see (4.91)) and the second one the expected risk $R^v(\mathbf{w}_n)$ (see (4.90)). We now define

$$V'_{HK}(\mathbf{w}_n) \triangleq V_{HK}\{v[g, \varphi(\cdot, \mathbf{w}_n)]\}, \quad n = 1, 2, \dots, \quad (4.104)$$

and introduce another assumption.

Assumption 4.3 The loss function v , the I/O relationship g , and the FSP functions $\gamma(\cdot, \mathbf{w}_n)$ are such that

$$\sup_{\mathbf{w}_n \in W_n} V'_{HK}(\mathbf{w}_n) < \infty.$$

□

Note that, thanks to Lemma 4.1, the assumption above is verified when the loss function v , the function g , and the FSP functions $\gamma(\cdot, \mathbf{w}_n)$ (for all \mathbf{w}_n) belong to $\mathcal{W}_M(\prod_j H_j)$ or $\mathcal{W}_{M,d}(\prod_j H_j)$, each for a particular choice of $\prod_j H_j$ and a particular value $M \leq \bar{M}$, where \bar{M} is finite (for sigmoidal and radial basis functions OHL networks, see, e.g., Sect. 4.4.5).

Then, Inequality (4.102) can be written as follows, enhancing the bound on the difference between the empirical risk and the expected risk, i.e., on the estimation error (see (4.53) with $\mathbf{w} = \mathbf{w}_n$):

$$\mathcal{E}_{\Sigma_L}^{\text{est}, v}(\mathbf{w}_n) = \left| R_{\text{emp}}^v(\mathbf{w}_n, \Sigma_L) - R^v(\mathbf{w}_n) \right| \leq V'_{HK}(\mathbf{w}_n) D_L^*(\{\mathbf{x}^L\}). \quad (4.105)$$

Finally, we can state the following result.

Theorem 4.7 *If Assumptions 4.2 and 4.3 are verified, then Property 4.2 (see Sect. 4.2.2) holds true.* □

Proof Assumption 4.3 enables one to take the supremum with respect to \mathbf{w}_n of both sides of (4.105). Thanks to Assumption 4.2, we can write

$$\sup_{\mathbf{w}_n \in W_n} V'_{HK}(\mathbf{w}_n) \cdot \lim_{L \rightarrow \infty} D_L^*(\{\mathbf{x}^L\}) = 0.$$

Then, the left-hand side of (4.105) goes to 0 when $L \rightarrow \infty$ and Condition (4.92) is verified. Thus, Property 4.2 holds true in virtue of Theorem 4.5. ■

Due to the structure of the KH inequality, the upper bound on the estimation error has a structure simpler than the bounds obtained in SLT (compare, for example, (4.105) with (4.58)). Indeed, the upper bound in (4.105) is given by the product of two factors without common variables. One factor is the star discrepancy of the chosen sequence. As previously stated, a sufficiently uniform spread of the vectors $\mathbf{x}^{(l)}$ in the sequence $\{\mathbf{x}^L\}$ lowers the value of its star discrepancy, hence, the value of the upper bound. Note also that the rate of convergence to zero of the upper bound on the estimation error when $L \rightarrow \infty$ depends on the rate of convergence to zero of the star discrepancy of $\{\mathbf{x}^L\}$.

Once the loss function has been fixed, the second factor (i.e., the variation $V'_{HK}(\mathbf{w}_n)$) depends on the chosen family of FSP functions and on their model complexities. If we compare the bounds in (4.105) and (4.58), we can see that the influence of the estimator γ on the estimation error bounds is described by the VC dimension h in SLT and by the variation $V'_{HK}(\mathbf{w}_n)$ in DLT.

4.4.7 Sample Sets Coming from (t, d) -Sequences

In Inequality (4.105) we pointed out that the rate of convergence to zero of the estimation error $\mathcal{E}_{\Sigma_L}^{\text{est}, v}(\mathbf{w}_n)$, when L goes to infinity, depends on the rate of convergence to zero of the star discrepancy $D_L^*(\{\mathbf{x}^L\})$. Then, in this section, we address (t, d) -sequences, a family of deterministic sequences that are provided with small values of $D_L^*(\{\mathbf{x}^L\})$ and have good convergence properties in the sense that their discrepancies decrease with L almost as $1/L$.

In general, a sequence $\{\mathbf{x}^L\}$ of points in $[0, 1]^d$ is called a *low-discrepancy sequence* if

$$D_L^*(\{\mathbf{x}^L\}) = O(L^{-1}(\ln L)^{d-1}). \quad (4.106)$$

We say that they generate *low-discrepancy sample sets* X_L (see also the definition given, for example, in [24, 30]). Such sequences are used in quasi-Monte Carlo integration methods, computer graphics, simulation, etc. The contents of this section take some basic concepts on the convergence and the construction of low-discrepancy sequences from [24].

Definition 4.7 Fix an integer $b \geq 2$. A subinterval E of $[0, 1]^d$ of the form

$$E = \prod_{i=1}^d [a_i b^{-p_i}, (a_i + 1)b^{-p_i}],$$

where $a_i, p_i \in \mathbb{Z}$, $p_i > 0$, $0 \leq a_i < b^{p_i}$ for $1 \leq i \leq d$, is defined as an “elementary interval in base b .” \triangleleft

Definition 4.8 Let t and m be two integers such that $0 \leq t \leq m$. A “ (t, m, d) -net in base b ” is a sample set P of b^m vectors in $[0, 1]^d$ such that $\#(E, P) = b^t$ for any elementary interval E in base b with $\lambda(E) = b^{t-m}$. \triangleleft

Let the set X_L be a (t, m, d) -net in base b . Note that any elementary interval E in base b of measure b^{t-m} must contain b^t vectors belonging to X_L . Therefore, X_L is endowed with properties of good “uniform spread in $[0, 1]^d$.” More specifically, it is easy to show that smaller values of t provide X_L with stronger uniformity properties (see Remark 4.3 in [24]). We now introduce the concept of (t, d) -sequence in base b that will be useful in the next section to determine upper bounds on the star discrepancy $D_L^*(\{\mathbf{x}^L\})$.

Clearly, if the sample set X_L or the sequence $\{\mathbf{x}^L\}$ is given by a (t, m, d) -net in base b , its cardinality is constrained to be $L = b^m$. In order to remove this limitation, we introduce the “ (t, d) -sequences in base b .”

Definition 4.9 Let $t \geq 0$ be a given integer. A sequence $\mathbf{x}_1, \mathbf{x}_2, \dots$ of vectors in $[0, 1]^d$ is a “ (t, d) -sequence in base b ” if, for all the integers $k \geq 0$ and $m > t$, the sample set $\{\mathbf{x}_i : kb^m + 1 \leq i < (k+1)b^m + 1\}$ is a (t, m, d) -net in base b . \triangleleft

Definitions 4.8 and 4.9 were introduced by Sobol [31] in the case $b = 2$; the general definitions were given by Niederreiter [24]. Two sample sets X_L are shown in Fig. 4.13.

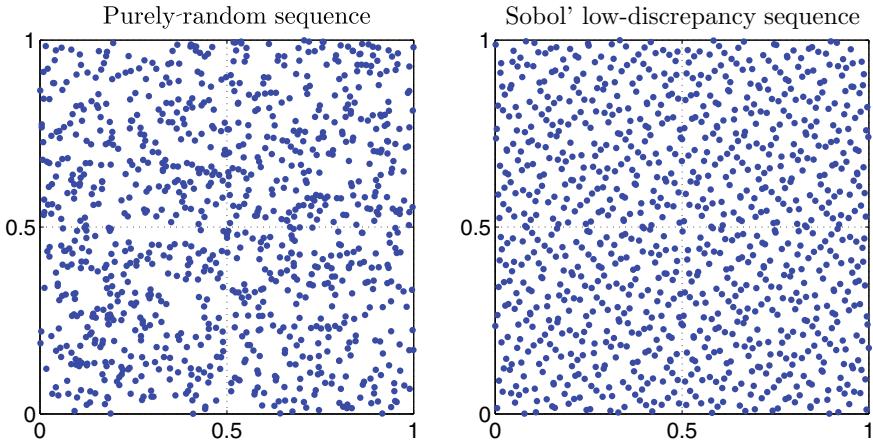


Fig. 4.13 Example of sample sets $X_L \in [0, 1]^2$ drawn by a Monte Carlo technique (left) and by a Sobol's low-discrepancy method (right)

4.4.8 Bounds on Rate of Convergence of the Estimation Error

Let $\{\mathbf{x}^L\}$ be given by the first L elements of a (t, d) -sequence $\mathbf{x}_1, \mathbf{x}_2, \dots$ in base b . In [24], bounds for the star discrepancy $D_L^*(\{\mathbf{x}^L\})$ were provided in order to enable one to determine a sufficient number L of samples so that $D_L^*(\{\mathbf{x}^L\})$ is smaller than a desired value. In particular, the result established by the following theorem is important.

Theorem 4.8 ([24, p. 60]) *The star discrepancy $D_L^*(\{\mathbf{x}^L\})$ of the first L elements of a (t, d) -sequence $\mathbf{x}_1, \mathbf{x}_2, \dots$ in base b verifies the inequality*

$$D_L^*(\{\mathbf{x}^L\}) \leq \frac{1}{L} [C(d, b) b^t (\ln L)^d + O(b^t (\ln L)^{d-1})] \quad \text{for } L \geq 2, \quad (4.107)$$

where

$$C(d, b) = \frac{1}{d} \left(\frac{b-1}{2 \ln b} \right)^d$$

if either $d = 2$ or $b = 2$, $d = 3, 4$; otherwise

$$C(d, b) = \frac{1}{d!} \frac{b-1}{2 \lfloor b/2 \rfloor} \left(\frac{\lfloor b/2 \rfloor}{2 \ln b} \right)^d.$$

□

Let us now assume that the base b is a *prime power* (i.e., an integer power of a prime number) and denote it by q . In this case, one has the following result.

Theorem 4.9 ([24, p. 91]) *For every dimension $d \geq 1$ and every prime power q , there exists a $(T(q, d), d)$ -sequence in base q , where $T(q, d)$ is a suitable integer that depends only on q and d .* \square

The values of $T(q, d)$ are tabulated in [24, p. 93] for $q = 2, 3, 5$ and $1 \leq d \leq 30$.

Then, consider Inequality (4.107) and write it by replacing b with q and t with $T(q, d)$. Note that, in the second term of the right-hand side of (4.107), the constant b^t can be included in the “big O” notation. Thus, we obtain

$$D_L^*(\{\mathbf{x}^L\}) \leq \frac{1}{L} [C(d, q) q^{T(q,d)} (\ln L)^d + O((\ln L)^{d-1})] \quad \text{for } L \geq 2. \quad (4.108)$$

Since we know the exact values of $C(d, q)$ and $T(q, d)$ for any value of q and d , we can choose, for a given dimension d , the integer q that minimizes the expression $C(d, q) q^{T(q,d)}$. It can be shown that the minimizing number $q^\circ(d)$ can be determined within a finite set. Let

$$C^\circ(d) \triangleq \min_q C(d, q) q^{T(q,d)}. \quad (4.109)$$

The values of $q^\circ(d)$ and $C^\circ(d)$ are tabulated in [24, p. 96] for $2 \leq d \leq 20$.

In [24], the asymptotic behavior of $C^\circ(d)$ as $d \rightarrow \infty$ is then addressed, and the following inequality is derived:

$$C^\circ(d) < \frac{1}{d!} \left[\frac{d}{\ln(2d)} \right]^d.$$

Hence, $C^\circ(d) \rightarrow 0$ superexponentially as $d \rightarrow \infty$. Thus, only the second term remains in (4.108) and the star discrepancies of the “optimal” $(T(q^\circ(d), d), d)$ -sequences have the asymptotic bound

$$D_L^*(\{\mathbf{x}^L\}) = O\left(\frac{(\ln L)^{d-1}}{L}\right). \quad (4.110)$$

Consequently, such optimal sequences verify Assumption 4.2 with an almost linear rate of convergence.³

Now, let Assumption 4.3 be verified with

$$\bar{V}_n \triangleq \sup_{\mathbf{w}_n} V'_{HK}(\mathbf{w}_n) < \infty. \quad (4.111)$$

³We refer to $O(1/L)$ and $O(1/L^{1/2})$ as “linear” and “quadratic” rates with respect to L , respectively, for the following reasons that have close but simpler similarities with what reported in Assumptions 2.10, 2.11, 2.12, and 2.13 (L takes the place of n ; the dimension d is absent). Let Q be a quantity dependent on L in such a way that $Q(L) = O(1/L)$ ($Q(L)$ corresponds, e.g., to the left-hand terms of (2.48) and (2.70)). Then, there exists a constant c_1 such that $Q(L) \leq c_1/L$. Let $\varepsilon > 0$. To evaluate the rate at which L has to grow when $\varepsilon \rightarrow 0$, hence when $1/\varepsilon \rightarrow \infty$, in such a way to guarantee that $Q(L) \leq \varepsilon$ holds, we impose $c_1/L \leq \varepsilon$. This provides $L \geq c_1/\varepsilon$, which means that when $1/\varepsilon \rightarrow \infty$, $L \rightarrow \infty$ linearly with respect to $1/\varepsilon$. Analogously, $Q(L) = O(1/L^{1/2})$ implies that there exists c_2 such that $Q(L) \leq c_2/L^{1/2}$ and so, as shown above, we get $L \geq (c_2/\varepsilon)^2$, which expresses that, for $1/\varepsilon \rightarrow \infty$, $L \rightarrow \infty$ quadratically with respect to $1/\varepsilon$.

Then, from (4.105) and (4.110), we obtain

$$\mathcal{E}_{\Sigma_L}^{\text{est},v}(\mathbf{w}_n) = \left| R_{\text{emp}}^v(\mathbf{w}_n, \Sigma_L) - R^v(\mathbf{w}_n) \right| \leq \bar{V}_n D_L^*(\{\mathbf{x}^L\}). \quad (4.112)$$

By applying Proposition 4.1, the following inequality follows from (4.112) (see (4.31)):

$$R^v(\mathbf{w}_{n\Sigma_L}^\circ) - R^v(\mathbf{w}_n^\circ) \leq 2\bar{V}_n D_L^*(\{\mathbf{x}^L\}). \quad (4.113)$$

Taking into account Inequality (4.113) and Definitions (4.90) and (4.91), we can summarize what discussed in this section in the following results:

Theorem 4.10 *If $\mathbf{x}_1, \mathbf{x}_2, \dots$ is a $(T(q^\circ(d), d), d)$ -sequence (generating Σ_L) and (4.111) holds true, then*

$$(i) R^v(\gamma_{n\Sigma_L}^\circ) - R^v(\gamma_n^\circ) = R^v(\mathbf{w}_{n\Sigma_L}^\circ) - R^v(\mathbf{w}_n^\circ) \leq 2\bar{V}_n D_L^*(\{\mathbf{x}^L\}) \\ = O\left(\frac{\bar{V}_n (\ln L)^{d-1}}{L}\right); \quad (4.114)$$

(ii) Property 4.2 (see Sect. 4.2.2) holds true. □

Inequality (4.114) states that the use of “optimized” $(T(q^\circ(d), d), d)$ -sequences enables the estimation error to have a convergence rate to zero of order $O(1/L)$ when $L \rightarrow \infty$ (we ignore the logarithmic factor). This constitutes an important feature of this kind of deterministic sequences, as they allow a faster asymptotic convergence than the random generation of points, which (as we have seen in Sect. 4.3) achieves a rate $O(1/L^{1/2})$. Furthermore, since the bounds considered in this section are generated by deterministic algorithms, their convergence is not subject to a probabilistic confidence.

It must be remarked that the research on low-discrepancy sequences has led, in the past years, to a better understanding of the reasons why such sampling schemes are successful in practice when employed in the context of numerical integration. In particular, it has been found that even better convergence rates can be obtained by careful construction when the integrand is very smooth and there is only mild joint dependence on the possible subsets of the components (for instance, when it belongs to a weighted Sobolev space with fast decaying weights). In this case, the dependence on d of the bounds, which influences negatively the rate at the non-asymptotic level due to the presence of the $(\ln L)^{d-1}$ term, can be reduced significantly. The interested reader is referred, for instance, to [8] (where the concept of *digital nets*, further generalizing the (t, d) -net one, is introduced) and to the references therein for more details.

However, in the rest of the book we shall consider the more general case presented so far, without assuming too strict (and, in the context of optimization and control addressed in the book, mostly unrealistic) regularity conditions on the involved functions.

4.4.9 Estimation of Functions in the Presence of Additive Noise

Let us consider the realistic case in which the output variable y in (4.82) is affected by a random additive noise η . The I/O relationship is then described by (4.6) that we repeat here, i.e.,

$$y = g(\mathbf{x}) + \eta,$$

where $g(\mathbf{x})$ is the unknown function to be estimated and $\eta \in E \subseteq \mathbb{R}$. The deterministic input samples $\mathbf{x}^{(l)}$, $l = 1, \dots, L$, are generated again by a deterministic algorithm. The output samples $y^{(l)}$ are affected by random noises $\eta^{(l)}$.

Following [4], we assume \mathbf{x} to be uniformly distributed on X as in Sect. 4.4.1, and η to be independent of \mathbf{x} , with zero mean and provided with the probability density $p(\eta)$. The case considered in this section corresponds to the situation that we have called “stochastic I/O relationship” in the last column on the right in Fig. 4.2.

As in (4.83), we consider a quadratic loss function. Then, for some family of OHL networks $\gamma(\cdot, \mathbf{w}_n)$ with a given basis cardinality n (or FSP functions with a given model complexity n), Expected Risk (4.17) takes on the following form:

$$\begin{aligned} R(\mathbf{w}_n) &= \int_{X \times E} [g(\mathbf{x}) + \eta - \gamma(\mathbf{x}, \mathbf{w}_n)]^2 p(\eta) d\eta d\mathbf{x} \\ &= \int_X [g(\mathbf{x}) - \gamma(\mathbf{x}, \mathbf{w}_n)]^2 d\mathbf{x} + \int_E \eta^2 p(\eta) d\eta \end{aligned} \quad (4.115)$$

$$+ 2 \int_X [g(\mathbf{x}) - \gamma(\mathbf{x}, \mathbf{w}_n)] d\mathbf{x} \int_E \eta p(\eta) d\eta \quad (4.116)$$

$$= \int_X [g(\mathbf{x}) - \gamma(\mathbf{x}, \mathbf{w}_n)]^2 d\mathbf{x} + \int_E \eta^2 p(\eta) d\eta. \quad (4.117)$$

since η has zero mean.

In a similar way, Empirical Risk (4.18) is given by

$$\begin{aligned} R_{\text{emp}}(\mathbf{w}_n, \Sigma_L) &= \frac{1}{L} \sum_{l=1}^L [g(\mathbf{x}^{(l)}) - \gamma(\mathbf{x}^{(l)}, \mathbf{w}_n)]^2 + \frac{1}{L} \sum_{l=1}^L (\eta^{(l)})^2 \\ &\quad + \frac{2}{L} \sum_{l=1}^L \{[g(\mathbf{x}^{(l)}) - \gamma(\mathbf{x}^{(l)}, \mathbf{w}_n)]\eta^{(l)}\}. \end{aligned} \quad (4.118)$$

The following inequality is derived from (4.117) and (4.118):

$$\begin{aligned}
& \sup_{\mathbf{w}_n \in W_n} |R_{\text{emp}}(\mathbf{w}_n, \Sigma_L) - R(\mathbf{w}_n)| \\
& \leq \sup_{\mathbf{w}_n \in W_n} \left| \frac{1}{L} \sum_{l=1}^L [g(\mathbf{x}^{(l)}) - \gamma(\mathbf{x}^{(l)}, \mathbf{w}_n)]^2 - \int_X [g(\mathbf{x}) - \gamma(\mathbf{x}, \mathbf{w}_n)]^2 d\mathbf{x} \right| \\
& \quad + \left| \frac{1}{L} \sum_{l=1}^L (\eta^{(l)})^2 - \int_E \eta^2 p(\eta) d\eta \right| \\
& \quad + 2 \sup_{\mathbf{w}_n \in W_n} \max_{l=1, \dots, L} |g(\mathbf{x}^{(l)}) - \gamma(\mathbf{x}^{(l)}, \mathbf{w}_n)| \left| \frac{1}{L} \sum_{l=1}^L \eta^{(l)} \right|. \tag{4.119}
\end{aligned}$$

Let us consider the first term in the right-hand side of Inequality (4.119). If the function g and the family of FSP functions $\gamma(\cdot, \mathbf{w}_n)$ have a bounded variation on $[0, 1]^d$ in the sense of Hardy and Krause, by virtue of Lemma 4.1 and Corollary 4.1, the same holds true for the quadratic loss function (compare with (4.103))

$$\varphi(\mathbf{x}, \mathbf{w}_n) = [g(\mathbf{x}) - \gamma(\mathbf{x}, \mathbf{w}_n)]^2.$$

Since Theorem 4.6 can be used and Assumption 4.3 is verified, from Koksma–Hlawka Inequality (4.102) (let $X = [0, 1]^d$) and (4.104) we obtain

$$\sup_{\mathbf{w}_n \in W_n} \left| \frac{1}{L} \sum_{l=1}^L \varphi(\mathbf{x}^{(l)}, \mathbf{w}_n) - \int_{[0,1]^d} \varphi(\mathbf{x}, \mathbf{w}_n) d\mathbf{x} \right| \leq \bar{V}_n D_L^* (\{\mathbf{x}^L\}). \tag{4.120}$$

Then, according to Theorem 4.10, if the sample set Σ_L is generated by a $(T(q^\circ(d), d), d)$ -sequence or, more in general, by a low-discrepancy sequence, then the first term in the right-hand side of (4.119) converges to zero when $L \rightarrow \infty$ with a rate of order $O(1/L)$ (ignoring the logarithmic factor).

As regards the second and the third terms in (4.119), the weak law of large numbers (which guarantees convergence in probability of the average of a random variable to its mean value) implies that both converge to 0 in probability as L tends to ∞ . A rate of convergence can be derived by applying, e.g., *Hoeffding's inequality* [7, 20], which states that, for k i.i.d. random variables v_1, \dots, v_k having zero mean and such that $a_i \leq v_i \leq b_i$ for $i = 1, \dots, k$ and any $\tau \geq 0$, one has

$$P \left\{ \sum_{i=1}^k v_i > \tau \right\} \leq \exp \left(- \frac{2\tau^2}{\sum_{i=1}^k (b_i - a_i)^2} \right).$$

For the second term in (4.119), it can be easily shown that, by applying this inequality with

$$v_i = (\eta^{(i)})^2 - \int_E \eta^2 p(\eta) d\eta,$$

a quadratic rate of convergence is obtained. The same rate can be obtained for the third term by putting $v_i = \eta^{(i)}$.

We can conclude that the use of low-discrepancy sequences preserves the consistency of the ERM principle even in the noisy case. As pointed out in [4], the presence of noise (i.e., the second and the third term in (4.119)) prevents one from maintaining a linear rate of convergence but enables a quadratic rate, which is still not worse than the rates derived in SLT. As we said before, the noisy case dealt with can be considered as a hybrid situation between SLT and DLT. According to [4], it is worth noting that, if the random noise is “small”, then the application of the Bernstein–Chernoff bounds [2] for the two terms on the right-hand side of (4.119) again enables one to obtain an almost linear rate of convergence.

References

1. Alon N, Ben-David S, Cesa-Bianchi N, Haussler D (1997) Scale-sensitive dimensions, uniform convergence, and learnability. *J ACM* 44:615–631
2. Angluin D, Valiant L (1979) Fast probabilistic algorithms for Hamiltonian circuits and matchings. *J Comput Syst Sci* 18:155–193
3. Barron AR (1994) Approximation and estimation bounds for artificial neural networks. *Mach Learn* 14:115–133
4. Cervellera C, Muselli M (2004) Deterministic design for neural network learning: an approach based on discrepancy. *IEEE Trans Neural Netw* 15:533–544
5. Cherkassky V, Mulier F (2007) learning from data: concepts, theory, and methods. Wiley
6. Cucker F, Smale S (2001) On the mathematical foundations of learning. *Bull Am Math Soc* 39:1–49
7. Devroye L, Györfi L, Lugosi G (1997) A probabilistic theory of pattern recognition. Springer, New York
8. Dick J, Pillichshammer F (2010) Digital nets and sequences: discrepancy theory and quasi-Monte Carlo integration. Cambridge University Press
9. Dudley RM (1979) Balls in \mathbb{R}^k do not cut all subsets of $k + 2$ points. *Adv Math* 31:306–308
10. Dudley RM, Giné R, Zinn J (1991) Uniform and universal Glivenko-Cantelli classes. *J Theor Prob* 4:485–510
11. Fang K-T, Wang Y (1994) Number-theoretic methods in statistics. Chapman & Hall
12. Geman S, Bienenstock E, Doursat R (1992) Neural networks and the bias/variance dilemma. *Neural Comput* 4:1–58
13. Girosi F (1995) Approximating error bounds that use VC bounds. In: Proceedings of the international conference on artificial neural networks, pp 295–302
14. Girosi F, Anzellotti G (1992) Rates of convergence of approximation by translates. Technical Report 1288, Artificial Intelligence Laboratory, Massachusetts Institute of Technology
15. Guyon I, Vapnik V, Boser B, Bottou L, Solla S (1992) Capacity control in linear classifiers for pattern recognition. In: Proceedings of the 11th IAPR international conference on pattern recognition, conference B: pattern recognition methodology and systems, vol II, pp 385–388
16. Hastie T, Tibshirani R, Friedman J (2008) The elements of statistical learning. Springer, New York
17. Haussler D (1992) Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Inf Comput* 100:78–150
18. Haykin S (2008) Neural networks and learning systems. Pearson Prentice-Hall
19. Hlawka E (1961) Funktionen von Beschränkter Variation in der Theorie der Gleichverteilung. *Ann Mat Pura Appl* 54:325–333

20. Hoeffding W (1963) Probability inequalities for sums of bounded random variables. *J Am Stat Assoc* 58:13–30
21. Kuipers L, Niederreiter H (1974) Uniform distribution of sequences. Wiley
22. McCaffrey DF, Gallant AR (1994) Convergence rates for single hidden layer feedforward nets. *Neural Netw* 7:147–158
23. Mendelson S (2003) A few notes on statistical learning theory. In: Mendelson S, Smola A (eds) Advanced lectures on machine learning – LNCS 2600, pp 1–40. Springer, Berlin
24. Niederreiter H (1992) Random number generation and Quasi-Monte Carlo methods. SIAM
25. Niyogi P, Girosi F (1994) On the relationship between generalization error, hypothesis complexity, and sample complexity for radial basis functions. Technical report, A.I. Memo No. 1467, C.B.C.L. No. 88, Massachusetts Institute of Technology, <ftp://publications.ai.mit.edu/ai-publications/1000-1499/AIM-1467.ps.Z>
26. Niyogi P, Girosi F (1996) On the relationship between generalization error, hypothesis complexity, and sample complexity for radial basis functions. *Neural Comput* 8:819–842
27. Nussbaum M (1996) On nonparametric estimation of a regression function that is smooth in a domain on \mathbb{R}^k . *Theor Probab Appl* 31:118–125
28. Owen A (2005) Multidimensional variation for quasi-Monte Carlo. In: Fang J, Li G (eds) Contemporary multivariate analysis and experimental design – celebration in honor of Professor Kai-Tai Fang's 65th birthday, pp 49–85. World Scientific
29. Pollard D (1990) Empirical processes: theory and applications. In: NSF-CBMS regional conference series in probability and statistics, vol 2. Institute of Mathematical Statistics and American Statistical Association
30. Schlier C (2004) Discrepancy behaviour in the non-asymptotic regime. *Appl Numer Math* 50:227–238
31. Sobol' IM (1967) The distribution of points in a cube and the approximate evaluation of integrals. *USSR Comput Math Math Phys* 7:86–112
32. Vapnik VN (1982) Estimation of dependences based on empirical data. Springer
33. Vapnik VN (1998) Statistical learning theory. Wiley
34. Vapnik VN (2000) The nature of statistical learning theory, 2nd edn. Springer
35. Vapnik VN, Chervonenkis AY (1971) On the uniform convergence of relative frequencies of events to their probabilities. *Theor Probab Appl* 16:264–280
36. Vapnik VN, Chervonenkis AJ (1991) The necessary and sufficient conditions for consistency in the empirical risk minimization method. *Pattern Recogn Image Anal* 1:283–305
37. Vidyasagar M (1997) A theory of learning and generalization. Springer, Berlin

Chapter 5

Numerical Methods for Integration and Search for Minima



This chapter brings the more methodological part of the book to an end. A few more theoretical concepts will be brought to the reader's attention. In the following chapters, we should have all the instruments to deal with the solution of *Problem P_n in the form (2.11)* – when the cost function $\tilde{F}(\mathbf{w}_n)$ is considered – and to *Problem P_n in the form (2.31)* – when one thinks possible and useful to exploit the particular structure of (2.30), that is, $\tilde{F}(\mathbf{w}_n) = \underset{z}{\mathbb{E}} J(\mathbf{w}_n, z)$. As we explained in Chap. 2, Problem P_n plays a central role in the approximate solution of the extremely wide range of infinite-dimensional optimization (IDO) problems. In particular, we shall focus on optimal control problems, which will be deterministic in Chap. 6 and stochastic in the subsequent chapters.

In this chapter, we shall address (i) the numerical computation of the expected values of functions depending on random variables and (ii) the search for the minima of functions depending on the control variables to be optimized and on the random variables with respect to which one has to compute the expected values. The argument (i) will be considered in Sects. 5.1 (integration) and 5.2 (computation of expected values). Of course, integration includes the computation of the expected value of a function. However, the latter shows peculiar nontrivial aspects that the former does not have. Consequently, the two computations are addressed in different sections. Clearly, the computation of the expected value of functions is essential in the solution of the stochastic optimal control problems that are equivalent to “Problem P_n in the form (2.11).” Section 5.2 is closely connected to the deterministic learning theory (DLT) presented in Chap. 4, where the sequences $\{\mathbf{x}^L\}$ (or the sample sets X_L) obtained by the quasi-Monte Carlo method were described. The properties of these sequences will become very useful in Chaps. 6 and 7, when we shall have to apply dynamic programming (DP) to the solution of nonlinear T -stage optimal control problems of a very general form. The good behavior of the star discrepancy $D_L^*(X_L)$ will enable us to mitigate one of the kinds of curse of dimensionality in DP related to the discretization of the admissible state domains required by the numerical use of

DP. This kind of curse of dimensionality is unavoidable if discretization is performed by using regular grids and the dimension d of the state vector becomes larger and larger.

As regards the search for minima (addressed in Sect. 5.3), we recall that most of the IDO problems described in the book are stated in stochastic frameworks. The extended Ritz method (ERIM) approximates them by nonlinear programming (NLP) problems (i.e., by finite-dimensional optimization (FDO) problems) in which the cost functions take on the form of expected values containing the control variables to be optimized. On this subject, two classical approaches will be considered: (1) NLP techniques given by direct search (i.e., based only on the evaluation of the cost function) and gradient-based descent algorithms. This means that Problem P_n is stated in the form (2.11). We shall just mention a few well-known techniques for which we refer the reader to the literature; (2) *stochastic approximation* (SA) algorithms that enable one to avoid the computation of the expected values. This technique is by far the most used throughout the book to solve NLP problems derived by the ERIM. So, we shall deal with it in more detail. In Sect. 5.4 the search for minima – and in particular for the global minimum – is addressed for problems in which the cost function has a low degree of regularity. This is the case, for instance, of non-differentiable cost functions. Then, once again, Monte Carlo (MC) and quasi-Monte Carlo (quasi-MC) direct search techniques may be effective.

5.1 Numerical Computation of Integrals

As we said in the introduction of this chapter, our interest in the numerical integration of functions stems especially from the fact that, from Chap. 7, we shall have to perform averaging operations on random costs by applying DP and the ERIM to the solution of stochastic optimal control problems. Since we shall assume that the classical linear-quadratic-Gaussian (LQG) assumptions (or any other set of assumptions that allow one to obtain solutions in a closed form) will not apply, we shall have to discretize the values of the random disturbances acting at each decision stage. Therefore, in high-dimensional settings, we shall incur the danger of another kind of curse of dimensionality.

Even though we are not directly interested in integration, the comprehension of its computational aspects makes it easier to understand how to perform averaging operations of functions. Consequently, we shall describe three techniques for the numerical integration of functions, namely, the discretization of the integration domain by regular grids, by MC methods, and by quasi-MC methods.

5.1.1 Integration by Regular Grids

Among the various methods available for the numerical integration of functions by using regular grids in dimension d , we focus on the *trapezoidal rule* for its easy geometrical interpretation. For the sake of simplicity, we shall assume that the integration domain is given by the closed d -dimensional unit cube $[0, 1]^d$. We partition each edge of the cube into m intervals of equal length $1/m$. Other methods use nonuniform integration step sizes. An adaptive algorithm determines the step lengths according to the behavior of the integrand function. It increases the density of the discretized points where the integrand is “less regular.” There is an extensive literature on numerical integration. For the few concepts that are dealt with here, we follow the first pages of [36].

At first, we consider the case $d = 1$ and we assume that the function $f: [0, 1] \rightarrow \mathbb{R}$ to be integrated is continuous. Then, according to the trapezoidal rule, the integral of f on each interval $[i/m, (i+1)/m]$, $i = 0, 1, \dots, m-1$, is approximated by $\{f(i/m) + f((i+1)/m)\}/(2m)$. Therefore, the following approximation is obtained:

$$\int_0^1 f(x)dx \simeq T_m^{(1)}(f) \triangleq \sum_{i=0}^m \Delta_i f(i/m), \quad (5.1)$$

where $\Delta_0 = \Delta_m = 1/(2m)$ and $\Delta_i = 1/m$, $i = 1, \dots, m-1$.

In the case of the trapezoidal rule, if f has a continuous second derivative on $[0, 1]$, it can be shown that the approximation error is given by

$$\left| \int_0^1 f(x)dx - T_m^{(1)}(f) \right| = O\left(\frac{1}{m^2}\right). \quad (5.2)$$

In the multidimensional case $d \geq 2$, a similar result can be obtained. The result is derived by performing the d -dimensional integration through a repeated procedure, in which the one-dimensional trapezoidal rule is applied in each dimension. Assume $f: [0, 1]^d \rightarrow \mathbb{R}$ to be continuous. Then, the estimation of the integral is given by

$$\int_{[0,1]^d} f(\mathbf{x})d\mathbf{x} \simeq T_m^{(d)}(f) \triangleq \sum_{i_1=0}^m \cdots \sum_{i_d=0}^m \Delta_{i_1} \cdots \Delta_{i_d} f\left(\frac{i_1}{m}, \dots, \frac{i_d}{m}\right), \quad (5.3)$$

where the coefficients $\Delta_{i_1}, \dots, \Delta_{i_d}$ are as in (5.1).

Let us assume the partial derivatives $\partial^2 f / \partial x_i^2$ to be continuous on $[0, 1]^d$, $i = 1, \dots, d$. Then, it can be shown that the order of magnitude of the error is the same as in the one-dimensional case, i.e.,

$$\left| \int_{[0,1]^d} f(\mathbf{x})d\mathbf{x} - T_m^{(d)}(f) \right| = O\left(\frac{1}{m^2}\right). \quad (5.4)$$

It is now important to consider the connections among the error (5.4), the dimension d , and the number L of grid points in which f has to be evaluated. The number of points is $L = (m + 1)^d$, hence $m \simeq L^{1/d}$. Then, we have

$$\left| \int_{[0,1]^d} f(\mathbf{x}) d\mathbf{x} - T_m^{(d)}(f) \right| = O(L^{-2/d}). \quad (5.5)$$

Under the assumption that its constant ‘‘hidden’’ in the ‘‘big O ’’-notation does not depend on d , the right-hand side of (5.5) implies a curse of dimensionality as, for a given level of the error, the upper bound on the number L of points increases exponentially with d . A similar issue actually arises also when the hidden constant depends on d , in such a way that reducing the bound by a given percentage of its original value requires an exponential relative increase of L with respect to d . It is worth noting that this phenomenon shows up in a similar way in the numerical computation of d -dimensional integrals based on the successive procedures that apply any one-dimensional integration rule. A thorough discussion on the curse of dimensionality for the numerical integration of functions $f \in \mathcal{C}^k(B)$, where $B \subset \mathbb{R}^d$ is an open set of Lebesgue measure equal to 1 (typically, the interior on the d -dimensional cube $[0, 1]^d$) can be found in [22].

In the remaining part of this section and in the next one, we shall consider alternative methods to the use of regular grids.

5.1.2 Integration by MC Methods

In general, MC methods consider the quantity to be computed in a stochastic context even if the quantity is deterministic. This means that they compute it by random sampling. To introduce this approach, we shall consider, for a function

$$f: B \rightarrow \mathbb{R}, \quad B \subset \mathbb{R}^d, \quad (5.6)$$

the integral

$$\int_B f(\mathbf{x}) d\mathbf{x} \quad (5.7)$$

to be determined in the framework of measure theory and Lebesgue integration. As is well known, if the Riemann integral of a function exists, then so does the Lebesgue integral, and both agree (see, e.g., [18, Sect. 2.1.1]). More precisely, let λ denote the d -dimensional Lebesgue measure and $B \subset \mathbb{R}^d$ be Lebesgue-measurable and such that

$$0 \leq \lambda(B) < \infty.$$

Then, let us consider B as a probability space with probability measure P , where

$$dP \triangleq \frac{d\lambda}{\lambda(B)}. \quad (5.8)$$

Therefore, assuming also that $f \in \mathcal{L}_1(B, \mathbb{R})$, we have

$$\int_B f(\mathbf{x}) d\mathbf{x} = \lambda(B) \int_B f dP = \lambda(B) E(f), \quad (5.9)$$

where $E(f) \triangleq \int_B f dP$.

According to (5.9), the function f is interpreted as a random variable with expected value $E(f)$. It follows that the problem of the approximate computation of an integral is reduced to the approximate computation of an expected value. This means that the *MC estimate of the expected value* $E(f)$ is obtained by generating L independent identically P -distributed random samples $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(L)} \in B$, and letting

$$E(f) \simeq \frac{1}{L} \sum_{i=1}^L f(\mathbf{x}^{(i)}). \quad (5.10)$$

By the strong law of large numbers, when $L \rightarrow \infty$, the right-hand side of (5.10) almost surely (a. s.) converges to $E(f)$ with respect to P , that is,

$$\lim_{L \rightarrow \infty} \frac{1}{L} \sum_{l=1}^L f(\mathbf{x}^{(l)}) = E(f), \quad P - \text{a. s.} \quad (5.11)$$

It is important to determine the probabilistic error of the MC estimate (5.10) and its rate of convergence in Limit (5.11). To this end, we assume that $f \in \mathcal{L}_2(B, \mathbb{R})$ and define the *variance* of f as

$$\sigma^2(f) \triangleq \int_B [f - E(f)]^2 dP. \quad (5.12)$$

The following theorem enables one to determine the error of the MC estimate of $E(f)$. Its proof is given, for example, in [36].

Theorem 5.1 *If $f \in \mathcal{L}_2(B, \mathbb{R})$, then, for every $L \geq 1$ we have*

$$\begin{aligned} & \int_B \cdots \int_B \left[\frac{1}{L} \sum_{l=1}^L f(\mathbf{x}^{(l)}) - E(f) \right]^2 dP(\mathbf{x}^{(1)}) \cdots dP(\mathbf{x}^{(L)}) \\ & = \frac{\sigma^2(f)}{L}. \end{aligned} \quad (5.13)$$

□

From (5.9) and (5.10), it follows that the *MC estimate of the integral* (5.7) is given by

$$\int_B f(\mathbf{x}) d\mathbf{x} \simeq \frac{\lambda(B)}{L} \sum_{l=1}^L f(\mathbf{x}^{(l)}). \quad (5.14)$$

In particular, for $B = [0, 1]^d$ we get

$$\int_{[0,1]^d} f(\mathbf{x}) d\mathbf{x} \simeq \frac{1}{L} \sum_{l=1}^L f(\mathbf{x}^{(l)}),$$

where $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(L)}$ are L i.i.d. random samples drawn from the uniform distribution on $[0, 1]^d$.

If we take the square root of (5.13), its right-hand side becomes $\sigma(f)/\sqrt{L}$. Therefore, Theorem 5.1 gives a very important result that can be stated as follows. *The MC method for numerical integration yields an upper bound on the probabilistic error whose rate of decrease is $O(L^{-1/2})$.* It is important to point out that this rate does not depend on the dimension d of the vector \mathbf{x} . By contrast, we have seen that the classical numerical integration methods, based on the use of regular grids, have a rate $O(L^{-2/d})$ (see (5.5)). According to [36], as regards the rate, the MC method is definitely preferable to the classical integration procedure for large values of d , say $d \geq 5$. However, the bound provided by the MC method is merely probabilistic. Of course, it is advisable to perform as many computations of (5.14) as possible, each with a renewed sampling. From a statistical point of view, this increases the reliability of the overall estimation procedure. It is also worth noting that the efficiency of the MC method for numerical integration can be improved by using various techniques. Such techniques aim at the *variance reduction*, i.e., at decreasing the variance factor in (5.13) [36, Chap. 1].

Of course, the estimate (5.14) is difficult to compute if the measure $\lambda(B)$ is difficult to compute as well. This occurs whenever the form of the integration domain B is “complex”. In this case, we can change the computational procedure as follows. If B is not contained in the unit cube $[0, 1]^d$, we reduce¹ B by applying a suitable change of variables so that the new domain (which is still denoted here by B) is included in $[0, 1]^d$.

¹ As regards this point, a caveat must be stated. Owing to the behavior of the volumes of sets in \mathbb{R}^d with respect to d (see, e.g., [26]), the abovementioned procedure of “reducing” B in such a way that it is contained in the d -dimensional cube might imply an “exponential enlargement” of the domain (see also the remarks in [26, Sect. 18.2]).

Then, the integral becomes

$$\int_B f(\mathbf{x}) d\mathbf{x} = \int_{[0,1]^d} f(\mathbf{x}) \chi_B(\mathbf{x}) d\mathbf{x}, \quad (5.15)$$

where χ_B is the characteristic function of B . Let

$$\tilde{f}(\mathbf{x}) \triangleq f(\mathbf{x}) \lambda_B(\mathbf{x}). \quad (5.16)$$

The right-hand side in (5.15) is now estimated according to (5.14) and the following *MC estimate* is obtained:

$$\int_B f(\mathbf{x}) d\mathbf{x} \simeq \frac{1}{L} \sum_{l=1}^L f(\mathbf{x}^{(l)}) \lambda_B(\mathbf{x}^{(l)}) = \frac{1}{L} \sum_{l=1}^L \tilde{f}(\mathbf{x}^{(l)}). \quad (5.17)$$

Since Estimate (5.17) is derived as (5.14), we conclude that (5.15) is also characterized by a probabilistic error whose rate of decrease is $O(L^{-1/2})$.

Estimates (5.10) and (5.14), which differ only for the presence of the measure $\lambda(B)$ in (5.14), are independent of the dimension d . The usefulness of MC and quasi-MC estimates (which will be described in the next section) will be evident in Chap. 7 where we shall have to compute expected values of cost functions when applying DP for the solution of stochastic optimal control problems. Indeed, as shown in Sect. 5.1.1, the curse of dimensionality cannot be avoided if the expected values are computed by using the classical regular grids.

Remark 5.1 The advantages of the MC method are, however, weakened by some inconveniences.

(i) Estimates (5.10) and (5.14) are random variables, and their probabilistic errors are of order $O(L^{-1/2})$. This may be unacceptable in applications where very reliable numerical results are required.

(ii) Estimates (5.10) and (5.14) do not take into any account the regularity properties of the integrand, which is only supposed to be square-integrable. In other words, the MC error does not achieve any improvement when additional information on the regularity of the integrand is available.

(iii) Finally, the MC method relies on the fact that the samples are realizations of independent random variables. In practice, one can only generate pseudorandom numbers by deterministic algorithms implemented by computers. Indeed, the generation of pseudorandom numbers that can be considered a good representation of random variables is quite a hard task. Appropriate statistical tests for randomness should be passed (see, e.g., [36]).

If we come back to the fundamental result stated by the Koksma–Hlawka inequality (4.102), the possibility to effectively contrast the above-described drawbacks is

quite evident. We shall address this possibility in the next section, where quasi-MC integration methods will be described, together with a discussion on the pros and cons of their use with respect to the classic MC approach.

5.1.3 Integration by Quasi-MC Methods

The basic idea of quasi-MC methods consists in replacing the MC random samples with points obtained by a suitable deterministic algorithm. Consistently with the MC application, this algorithm also aims at generating points distributed as uniformly as possible, i.e., by low-discrepancy sequences. As we have seen in the previous chapter, uniformity is measured by the star discrepancy $D_L^*(\{\boldsymbol{x}^L\})$. In order to be consistent with the concepts and the results addressed in this section, as we did before, we consider the domain $B = [0, 1]^d$ (see also the arguments used to derive Estimate (5.17)).

In analogy with the MC approximation (5.14), we introduce the following quasi-MC estimate:

$$\int_{[0,1]^d} f(\boldsymbol{x}) d\boldsymbol{x} \simeq \frac{1}{L} \sum_{l=1}^L f(\boldsymbol{x}^{(l)}). \quad (5.18)$$

If f has a bounded variation on $[0, 1]^d$ in the sense of Hardy and Krause, Theorem 4.6 holds true. Thus, the integration error of the quasi-MC estimate is bounded as in (4.102), that is,

$$\left| \int_{[0,1]^d} f(\boldsymbol{x}) d\boldsymbol{x} - \frac{1}{L} \sum_{l=1}^L f(\boldsymbol{x}^{(l)}) \right| \leq V_{HK}(f) D_L^*(\{\boldsymbol{x}^L\}). \quad (5.19)$$

Koksma–Hlawka (KH) Inequality (5.19) states that the three drawbacks of the MC estimate of the integral pointed out in Remark 5.1 are mitigated. In fact, (i) if $\{\boldsymbol{x}^L\}$ is generated by a deterministic algorithm, then the star discrepancy $D_L^*(\{\boldsymbol{x}^L\})$ is deterministic as well, thus giving a guaranteed error bound in (5.19); (ii) the regularity properties of the integrand f are taken into account by $V_{HK}(f)$; (iii) the difficulty of generating truly random samples vanishes as $\{\boldsymbol{x}^L\}$ is a deterministic sequence.

Inequality (5.19) provides an insight into the choice of the deterministic sequences $\{\boldsymbol{x}^L\}$. Indeed, the use of low-discrepancy sequences guarantees a convergence rate of the integration error bound of order $O(L^{-1}(\ln L)^{d-1})$ when $L \rightarrow \infty$ (see, e.g., the optimal $(T(q^\circ(d), d), d)$ -sequences in (4.110)), whereas (as we have seen previously) the same error of the MC estimate has a rate of convergence of order $O(L^{-1/2})$.

The abovementioned improvements provided by the use of quasi-MC methods are not peculiar to integration. Indeed, for many numerical problems, which can be solved by MC methods, it is possible to develop quasi-MC techniques that can be considered as their deterministic versions. This occurs for problems in numerical

analysis such as the solution of integral equations. Other examples are given by the solution of systems of equations, boundary value problems, global optimization, etc. For a survey on the applications of the quasi-MC method, we refer to [36, pp. 11, 12] and [8].

It is, however, not to believe that quasi-MC methods always outperform MC ones. In the next section, we shall consider possible limitations of quasi-MC integration with respect to MC integration. Other drawbacks will be discussed in Sect. 5.2, where the numerical computation of the expected values of functions will be addressed. Note also what is reported in [44]: “Recently, we have performed extensive tests of quasi-Monte Carlo integration with integrand whose variation we can compute. These tests confirm (. . .) the well-known fact that practical integration errors are many orders of magnitude smaller than (4.102) would suggest.” Comments on this point are given in [44].

In order to combine the advantages of pure MC and quasi-MC techniques, *randomized quasi-MC sequences* were proposed [37]. In short, the method is based on the fact that the j th component $x_i^{(j)}$ of the i th point \mathbf{x}_i of a (t, d) -sequence in base b (see Definition 4.9) can be written in a “base b expansion” as

$$x_i^{(j)} = \sum_{k=1}^{\infty} a_{ijk} b^{-k},$$

where the terms a_{ijk} are suitable coefficients. It is possible to prove that a random permutation (over k) of these coefficients maintains the basic properties of the low-discrepancy sequence in terms of equidistribution and uniformity, while allowing to compute more accurate estimates of the integration error, typical of the MC approach. It is possible to further improve the almost linear rate of convergence of the deterministic low-discrepancy sequences by exploiting some additional regularity properties of the functions to be integrated, e.g., Lipschitz conditions [38].

5.1.4 Limitations of the Quasi-MC Integration with Respect to the MC One

Theorem 5.1 states that MC integration has a probabilistic error of order $O(L^{-1/2})$. Instead, in quasi-MC methods, low-discrepancy sequences enable one to compute integrals with deterministic upper bounds of order $O(L^{-1}(\ln L)^{d-1})$. However, in practice the superiority of quasi-MC sequences is often not verified. It is shown (see, e.g., [35] and the references therein) that this loss of superiority (in terms of loss of accuracy) is due mainly to two reasons:

1. As we have pointed out after Definition 4.6, the variation $V_{HK}(f)$ of a function f in the sense of Hardy and Krause is bounded if f is sufficiently smooth as specified by (4.99). However, if the integrand f is discontinuous, the KH

inequality (4.102) cannot be applied. Moreover, in [33, 34], it is shown – by computational experiments with Halton, Sobol', and Faure sequences – that if the dimension d increases or if the integrand function is not sufficiently smooth, then the error may be of size $cL^{-\alpha}$, where c is a positive constant and $1/2 \leq \alpha \leq 1$. Then, the bound in (4.108) reverts to the order $O(L^{-1/2})$.

2. The second limitation consists in the following. Despite the fact that the asymptotic bounds given by low-discrepancy sequences are very favorable, the $O(L^{-1}(\ln L)^{d-1})$ behavior turns out to take effect only for large values of L , whereas the actual behavior for “practical” sample sizes L is more similar to the Monte Carlo bounds. This has been verified experimentally (see, e.g., [33, 34, 44]) and is more evident as the dimension d increases. Notice that, for MC methods, Theorem 5.1 shows that the rate of convergence $O(L^{-1/2})$ of Error (5.13) is independent of the dimension d . On the other hand, MC methods allow one to derive, through classical statistical analysis, error bounds for finite, although probabilistic, samples. Concerning quasi-MC methods, finite-sample bounds actually exist (see, e.g, [33]) but they are far too conservative to be actually applied (and, in general, they still depend on the dimension d). Indeed, in the KH inequality, this comes out from the fact that the star discrepancy is usually difficult to compute and that loose bounds on its values are available.

In Chap. 4, a sequence $\{\mathbf{x}^L\}$ of L points has been defined as a “low-discrepancy” one if it verifies Bound (4.106), that is,

$$D_L^*(\{\mathbf{x}^L\}) = O\left(L^{-1}(\ln L)^{d-1}\right).$$

Usually, according to a more accurate treatment (see, e.g., [36]), $\{\mathbf{x}^L\}$ is called a low-discrepancy sequence if its star discrepancy verifies the following inequality:

$$D_L^*(\{\mathbf{x}^L\}) \leq c(d) [L^{-1}(\ln L)^d] + O\left(L^{-1}(\ln L)^{d-1}\right), \quad (5.20)$$

where $c(d)$ is a positive constant that depends on d but not on L (hence, it is not an absolute constant). In [44] it is reported that “it is generally conjectured that the order $O(L^{-1}(\ln L)^d)$ is the best possible for the discrepancy of an infinite sequence.” Notice that this is not in contrast with (4.110), where the first term in the right-hand side of (5.20) was neglected asymptotically, as its associated constant $c(d)$ converges to 0 superexponentially fast with respect to d . The constant $c(d)$ in (5.20) plays an important role to determine the behavior of an infinite sequence. In [44] it is also reported what follows: “Astonishingly, while everybody mentions (5.20), nobody knows where, in practice, the asymptotic regime begins. Is it at $L = 10^5$ or 10^{25} or 10^{100} ? And, strangely enough, there are few, if any, numerical computations of D_L^* for smaller L , i.e., those which are commonly used in integration problems.” Indeed, the form of the bound in (5.20) shows that the behavior of this bound for small L is different from its behavior for large values of L , where the asymptotic regime begins. This can be seen from the fact that the discrepancy bound in (5.20) has a maximum point located approximately at the point $L = e^d$.

In [44], results from extensive tests on quasi-MC computations of integrals (with integrands whose variations are computable) are reported. Halton, Niederreiter, Niederreiter–Xing, and Sobol’ sequences were considered. The following main results are summarized. In the region where the discrepancy bound in (5.20) attains its maximum, D_L^* takes on the behavior of a power. This power profile decreases very slowly for increasing values of L . D_L^* does not decrease monotonically but fluctuates with respect to an average trend showing local minima and maxima. The tests on quasi-MC integration not only confirmed the well-known fact that the “true” integration errors are many orders of magnitude smaller than is indicated by the bound in (5.20) but showed also that D_L^* does not behave as the constant $c(d)$ predicts. Essentially, [44] suggests not to avoid necessarily low-discrepancy sequences that are not considered “optimal” in the literature. What is important is to resort to repeated integrations with different low-discrepancy sequences in order to take confidence in the results.

As a last note, it has to be remarked that most of the research in the past few years has been focused on investigating conditions on the integrand, independently on its Hardy and Krause variation, that can guarantee better bounds than the ones seen so far, which are affected by the $(\ln L)^{d-1}$ term. This has focused the attention on the so-called “weighted spaces,” i.e., spaces of functions in which different subsets of coordinates are weighted differently, depending on the variability of the function in those particular directions. Weighted Sobolev spaces with fast decaying weights are examples of such spaces.

5.2 Numerical Computation of Expected Values

We begin this section by observing that the numerical computation of the expected value

$$E(f) = \int_B f(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} \quad (5.21)$$

via the integration of the function $f(\mathbf{x}) p(\mathbf{x})$ through regular grids is clearly to discard unless the dimension of \mathbf{x} is suitably small. This is due to the same reasons discussed in Sect. 5.1.1.

As regards the computation of $E(f)$ by the MC approach, we have seen in Sect. 5.1.2 that the estimate of the expected value is given by (5.10). Of course, the random vector \mathbf{x} is generally no longer uniformly distributed on B . Then, the samples $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(L)}$ have to be generated according to the probability density $p(\mathbf{x})$. In this connection, it is well known that, concerning the practical issue of generating on a computer (pseudo-) random sequences distributed according to $p(\mathbf{x})$, there are many consolidated methods to obtain the most common distributions, such as the normal,

the beta, the gamma, and so on. The *accept–reject algorithm* (see, e.g., [11, 43]) can be applied to obtain i.i.d. sequences drawn according to any desired distribution, starting from another one that is easier to generate (e.g., the uniform or the normal one). This being stated, Convergence (5.11) still holds, as well as Theorem 5.1 with the variance defined by (5.12). Another possibility is to employ a *Markov chain Monte Carlo technique* (see, e.g., [31] and the references cited therein) based on evolving special Markov chains in such a way that the limiting distribution of the chain is the target distribution.

The approach based on quasi-MC methods is considered in the next section.

5.2.1 Computation of Expected Values by Quasi-MC Methods

Let us examine the possibility of estimating $E(f)$ by using a quasi-MC approach. To this end, we let

$$\tilde{f}(\mathbf{x}) \triangleq f(\mathbf{x})p(\mathbf{x})$$

and $B = [0, 1]^d$. Then, we have

$$E(f) = \int_{[0,1]^d} f(\mathbf{x})p(\mathbf{x})d\mathbf{x} = \int_{[0,1]^d} \tilde{f}(\mathbf{x})d\mathbf{x}.$$

Accordingly, the estimate of the expected value of f is given by

$$E(f) \simeq \frac{1}{L} \sum_{l=1}^L f(\mathbf{x}^{(l)})p(\mathbf{x}^{(l)}) = \frac{1}{L} \sum_{l=1}^L \tilde{f}(\mathbf{x}^{(l)}), \quad (5.22)$$

where $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(L)} \in [0, 1]^d$ is a sequence generated by a deterministic algorithm. Now, the following assumption is needed.

Assumption 5.1 The function $f(\mathbf{x})$ and the probability density $p(\mathbf{x})$ belong to $\mathcal{W}_M([0, 1]^d)$. \triangleleft

By Corollary 4.1, under Assumption 5.1, it is possible to state that the product $f \cdot p$ has a bounded variation $V_{HK}(f \cdot p)$ in the sense of Hardy and Krause. Hence, by virtue of Theorem 4.6, we can write an inequality similar to (5.19) for any sequence $\{\mathbf{x}^L\}$, that is,

$$\left| E(f) - \frac{1}{L} \sum_{l=1}^L f(\mathbf{x}^{(l)})p(\mathbf{x}^{(l)}) \right| \leq V_{HK}(f \cdot p) D_L^*(\{\mathbf{x}^L\}). \quad (5.23)$$

Therefore, if $\{\mathbf{x}^L\}$ is a low-discrepancy sequence, then the quasi-MC estimate of $E(f)$ has the same advantages over the MC estimate (5.10) as those of the quasi-MC estimate of integrals over the MC one. This means that the rate of convergence to zero of the error of the quasi-MC estimate of expected values is of order $O(1/L)$ (except for a logarithmic factor) instead of $O(L^{-1/2})$.

Despite its formal correctness (concerning, in particular, the asymptotic rates of convergence), the approach we just described is quite naive and risks not to be effective in practice. This can be illustrated by the following very simple example.

Example 5.1 Consider the function $f(x_1, x_2) = \sin(50x_1 x_2)$ and the density

$$p(x_1, x_2) = 0.8\mathcal{N}(\boldsymbol{\mu}_1, \Sigma_1) + 0.2\mathcal{N}(\boldsymbol{\mu}_2, \Sigma_2)$$

given by a mixture of two Gaussians, with $\boldsymbol{\mu}_1 = [0.3 \ 0.5]^\top$, $\boldsymbol{\mu}_2 = [0.8 \ 0.2]^\top$, and

$$\Sigma_1 = \begin{bmatrix} 10^{-2} & 10^{-2} \\ 10^{-2} & 2 \cdot 10^{-2} \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 2 \cdot 10^{-3} & 10^{-3} \\ 10^{-3} & 2 \cdot 10^{-2} \end{bmatrix}.$$

Figure 5.1a and b depicts the function f and the density p in the unit cube $[0, 1]^2$, respectively. The product $f \cdot p$ in the same domain is shown in Fig. 5.1c. \triangleleft

The simple Example 5.1 should convince us of the inefficiency of sampling the points from a uniform distribution. Indeed, looking at Fig. 5.1c we can see that, even if the function f has a very complex behavior over the whole domain, if one spreads the points in such a way that the whole input space is uniformly sampled, then about 80% of them would be “wasted” in regions where the integrand (i.e., the product $f \cdot p$) is 0 or close to 0, thus not contributing to the numerical estimation of the integral.

To overcome this possible waste, it is suitable to get back to a context in which the points are generated not deterministically, as in quasi-MC methods, but according to a probability distribution, hence, using an MC approach. In this case, $E(f)$ is evaluated by the usual empirical mean (see Sect. 5.1.2) with the points $\mathbf{x}^{(l)}$ coming from a random sequence drawn according to the density p . From a practical point of view, by proceeding in such a way one is sure that no points are wasted in regions where the integrand has small weight (according to p). Alternatively, one may consider the improvements described in the next section.

5.2.2 Improving MC and Quasi-MC Estimates of Integrals and Expected Values

According to (5.13), the expected square errors of the MC estimates of the expected values and integrals of functions are related to the variance $\sigma^2(f)$, hence, to the sampling probability density $p(\mathbf{x})$. Then, the expected square errors can be reduced

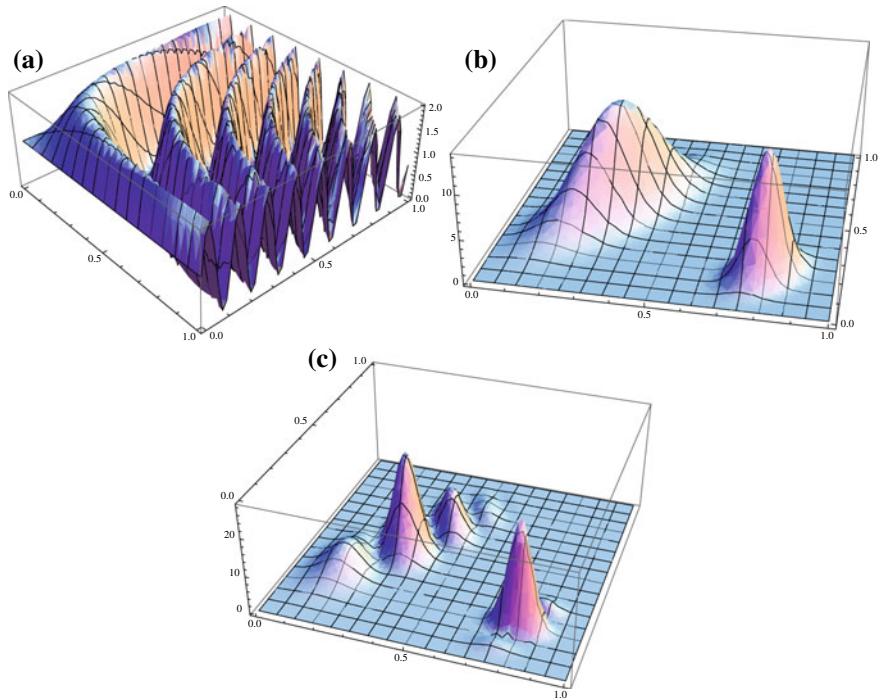


Fig. 5.1 **a** function f ; **b** density p ; **c** product of f and p

if $\sigma^2(f)$ decreases. A common technique for choosing sampling densities that achieve variance reduction is the so-called *importance sampling*.

Let us focus the attention on the estimate of the expected value $E(f)$. Such an estimate is obviously comprehensive of the estimate of the integral of f . The importance sampling technique is based on the simple fact that the expected value $E(f)$ has infinite alternative representations of the form

$$E(f) = \int_{[0,1]^d} f(\mathbf{x}) \frac{p(\mathbf{x})}{q(\mathbf{x})} q(\mathbf{x}) d\mathbf{x},$$

where q is any probability density function defined on $[0, 1]^d$ and with support large enough to contain that of p .

We can now consider an alternative version of the empirical estimate of the expected value having the form

$$E(f) \simeq \frac{1}{L} \sum_{l=1}^L \frac{p(\mathbf{x}^{(l)})}{q(\mathbf{x}^{(l)})} f(\mathbf{x}^{(l)}) = \frac{1}{L} \sum_{l=1}^L \omega_l f(\mathbf{x}^{(l)}), \quad (5.24)$$

where $\omega_l \triangleq p(\mathbf{x}^{(l)})/q(\mathbf{x}^{(l)})$. The vectors $\mathbf{x}^{(l)}$ are now drawn according to the probability density q . The coefficients ω_l are called *importance weights*.

Estimator (5.24) converges to the integral $E(f)$ for the same reasons as the usual MC estimate. There are two reasons to use the form (5.24) instead of the usual one. The first is that q may be “simpler” than p , and consequently samples may be generated more easily according to q . The second reason is that, if q is suitably chosen, it can be proved (see, for example, [11, 31, 43]) that the variance of the estimation error can be reduced significantly. This is particularly true when q is chosen in such a way that $f \cdot p/q$ is nearly constant and with finite variance.

Notice that, rather than the form (5.24), the following one is preferable:

$$E(f) \simeq \frac{\sum_{l=1}^L \omega_l f(\mathbf{x}^{(l)})}{\sum_{l=1}^L \omega_l}, \quad (5.25)$$

since it can be proved that often, in spite of the introduction of a small bias, its mean squared error is lower (see, e.g., [43]).

It is also worth noting that nothing prevents one from using a low-discrepancy sequence for the vectors $\mathbf{x}^{(l)}$ instead of a random one in a context of importance sampling. This can be done by using the accept–reject techniques mentioned at the beginning of the section to make the points of the low-discrepancy sequence distributed according to q .

Note that the accept–reject sampling technique may lose its effectiveness in a quasi-MC framework if compared with the MC one. Indeed, this technique integrates discontinuous functions. The discontinuity comes from the use of a characteristic function deriving from the accept–reject decision. This was pointed out in [33]. A smoothed version of the standard accept–reject technique is proposed in [53], where numerical experiments show the improvements of the *extended smoothed accept–reject method* over the standard accept–reject one.

Finally, it is worth mentioning more recent approaches that aim at directly generating low-discrepancy sequences characterized by an arbitrary distribution, without relying on the acceptance–rejection method. The reader may refer to [21], where a detailed discussion is provided also proposing a method that is able to cope with functions without finite variation, such as functions with singularities on the boundary of the domain.

5.3 Minimization Techniques for the Solution of Problem P_n

This section will provide a more detailed analysis of an issue already introduced in Sects. 2.3 and 2.3, i.e., the solution of the NLP Problem P_n both in the forms (2.11) and (2.31). We restate it here.

Problem P_n. Find

$$\mathbf{w}_n^{\circ} = \underset{\mathbf{w}_n \in W_n}{\operatorname{argmin}} \tilde{F}(\mathbf{w}_n) = \underset{\mathbf{w}_n \in W_n}{\operatorname{argmin}} \int J(\mathbf{w}_n, z) p(z) dz. \quad (5.26)$$

△

The necessity of computing the integral in (5.26) is the main reason why in this chapter we examine some techniques for the solution of Problem P_n. Indeed, as stated previously, this integral generally cannot be computed analytically. In Sects. 5.3.1 and 5.3.2, we shall consider classical NLP algorithms. In Sect. 5.3.3 and in the following ones, we present a stochastic approximation method that enables us to avoid the computation of integrals. In the following, Assumption 1.2 holds true, i.e., it is supposed that the probability density $p(z)$ is known.

5.3.1 Direct Search Methods

Let Assumption 2.4 be verified. Then, the cost function $J(\mathbf{w}_n, z)$ can be computed “exactly” (or “in a sufficiently accurate way”) for all \mathbf{w}_n and z . We discard the use of the Hessian matrix of $J(\mathbf{w}_n, z)$ as it typically involves too heavy computations. We also assume not to use the gradient $\nabla_{\mathbf{w}_n} J(\mathbf{w}_n, z)$. This may be due to a choice, to computational difficulties or, simply, to the fact that $J(\mathbf{w}_n, z)$ has discontinuous first derivatives. Thus, in order to solve Problem P_n, we have to resort to the so-called *direct search methods*, i.e., algorithms that only make use of the values of the cost function $\tilde{F}(\mathbf{w}_n)$. Algorithms of this type are the *coordinate descent methods* (which search along the set of coordinate directions), *Rosenbrock’s method*, *Powell’s method*, the *pattern search algorithm* by Hooke and Jeeves, the various types of *simplex methods*, and others. For these methods as well as for the gradient-based ones described later in the chapter, see, for example, [1, 3, 7, 27]. Another direct method is a *random search* based on the evaluation of the cost function at the samples generated in the admissible domain by MC techniques. The deterministic analogue of the random search is the *quasi-random search* that uses quasi-MC techniques (see, e.g., [28, 36]). Both methods will be briefly described at the end of the chapter.

In general, the integral in (5.26) (hence, the function \tilde{F}) cannot be computed analytically. Thus, one requires the numerical techniques described in Sects. 5.1 and 5.2, where we have addressed MC and quasi-MC methods. Such methods provide estimates of the form

$$\tilde{F}(\mathbf{w}_n) \simeq \frac{1}{L} \sum_{l=1}^L J(\mathbf{w}_n, z_l), \quad (5.27)$$

where L is the number of (exact) evaluations of the function $J(\mathbf{w}_n, z)$ at the sample vectors z_1, \dots, z_L . In the MC approach, these vectors are chosen randomly according to the probability density $p(z)$ whereas, in the quasi-MC context, they are generated by some suitable deterministic algorithm. As it was shown in Sect. 5.1,

Estimate (5.27) approximates the cost $\tilde{F}(\mathbf{w}_n)$ arbitrarily well provided that L is sufficiently large. More specifically, under suitable assumptions, the error between the “true” value of $\tilde{F}(\mathbf{w}_n)$ and its estimate (5.27) decreases at the rate $1/\sqrt{L}$ or even more rapidly.

5.3.2 Gradient-Based Methods

Let us suppose that Assumption 2.5 is verified and suppose that we want to solve Problem P_n by a gradient-based method. Clearly, we need again to compute the expected values in a numerical form. We point out that if $J(\mathbf{w}_n, \mathbf{z})$ is a function of class \mathcal{C}^1 with respect to \mathbf{w}_n and \mathbf{z} , then, under suitable hypotheses specified in Theorem 5.2, $\tilde{F}(\mathbf{w}_n)$ is a function of class \mathcal{C}^1 as well and we can write

$$\nabla_{\mathbf{w}_n} \tilde{F}(\mathbf{w}_n) = \nabla_{\mathbf{w}_n} \int J(\mathbf{w}_n, \mathbf{z}) p(\mathbf{z}) d\mathbf{z} = \nabla_{\mathbf{w}_n} \underset{\mathbf{z}}{\mathbb{E}}[J(\mathbf{w}_n, \mathbf{z})] \quad (5.28a)$$

$$= \int \nabla_{\mathbf{w}_n} J(\mathbf{w}_n, \mathbf{z}) p(\mathbf{z}) d\mathbf{z} = \underset{\mathbf{z}}{\mathbb{E}}[\nabla_{\mathbf{w}_n} J(\mathbf{w}_n, \mathbf{z})]. \quad (5.28b)$$

If (5.28) holds true, then we can interchange the operations of computing the gradient and the integral. $\nabla_{\mathbf{w}_n} J(\mathbf{w}_n, \mathbf{z})$ is called *stochastic gradient* because it depends on the random vector \mathbf{z} . The interchange of the gradient and the integral in (5.28) forms the basis for the stochastic approximation algorithms addressed in the next sections. The conditions under which the interchange of integration and differentiation is valid are formally stated in the theorem below. The theorem is given in [48, p. 509] and its proof in [12].

Consider the integral

$$\int_X h(\boldsymbol{\vartheta}, \mathbf{x}) d\mathbf{x}, \quad (5.29)$$

where X is the integration domain, the vector $\boldsymbol{\vartheta}$ belongs to a set Θ and $h: \Theta \times X \rightarrow \mathbb{R}$ is a differentiable function with respect to $\boldsymbol{\vartheta}$ for any $\boldsymbol{\vartheta} \in \Theta$ and $\mathbf{x} \in X$.

Theorem 5.2 *Let Θ be an open set and h and $\nabla_{\boldsymbol{\vartheta}} h$ be continuous on $\Theta \times X$. Suppose that there exist nonnegative Lebesgue-integrable functions $q_0(\mathbf{x})$ and $q_1(\mathbf{x})$ such that*

$$|h(\boldsymbol{\vartheta}, \mathbf{x})| \leq q_0(\mathbf{x})$$

and

$$|\nabla_{\boldsymbol{\vartheta}} h(\boldsymbol{\vartheta}, \mathbf{x})| \leq q_1(\mathbf{x}),$$

for all $(\boldsymbol{\vartheta}, \mathbf{x}) \in \Theta \times X$. Suppose also that

$$\int_X q_0(\mathbf{x}) d\mathbf{x} < \infty$$

and

$$\int_X q_1(\mathbf{x}) d\mathbf{x} < \infty.$$

Then, the interchange of differentiation and integration is valid, i.e.,

$$\nabla_{\vartheta} \int_X h(\vartheta, \mathbf{x}) d\mathbf{x} = \int_X \nabla_{\vartheta} h(\vartheta, \mathbf{x}) d\mathbf{x}.$$

□

Theorem 5.2 is a consequence of Lebesgue's dominated convergence theorem, which gives conditions under which a limit can be brought inside an integral [18].

In order to concentrate on the peculiarities of the structure of Problem P_n with the cost expressed in the form (5.26), we assume this problem to be unconstrained, that is, $W_n = \mathbb{R}^{\mathcal{N}(n)}$. Among the various gradient-based algorithms, two important classes of techniques are given by the *conjugate direction methods* and the methods of the type

$$\mathbf{w}_n(k+1) = \mathbf{w}_n(k) - \alpha_k S_k \nabla \tilde{F}[\mathbf{w}_n(k)], \quad k = 0, 1, \dots, \quad (5.30)$$

where $\alpha_k > 0$ is a stepsize to be properly selected so that the cost function is decreased at each iteration. The algorithm starts from an initial guess point $\mathbf{w}_n(0)$, and S_k is a symmetric positive-definite matrix (this guarantees a descent direction). Typical gradient-based algorithms of the form (5.30) range from the *steepest descent method* (for which $S_k = I$) to the *quasi-Newton methods* where S_k is adjusted from one iteration to the next in order that the direction

$$-S_k \nabla \tilde{F}[\mathbf{w}_n(k)]$$

tends to approximate Newton's method direction

$$-HF[\mathbf{w}_n(k)]^{-1} \nabla \tilde{F}[\mathbf{w}_n(k)]. \quad (5.31)$$

The progressive approximation of the Hessian matrix $HF[\mathbf{w}_n(k)]$ by S_k is obtained by using two successive iterates $\mathbf{w}_n(k)$ and $\mathbf{w}_n(k+1)$ together with the corresponding gradients $\nabla \tilde{F}[\mathbf{w}_n(k)]$ and $\nabla \tilde{F}[\mathbf{w}_n(k+1)]$. Among quasi-Newton methods, we mention the *Davidon–Fletcher–Powell algorithm* and the *Broyden–Fletcher–Goldfarb–Shanno algorithm*.

If Problem P_n is constrained (i.e., if W_n is a proper subset of $\mathbb{R}^{\mathcal{N}(n)}$), then it can be solved, e.g., by the *feasible direction* and the *conditional-gradient methods*, the *gradient-projection method*, and the *two-metric projection method* [3, Chap. 2]. Another approach, which takes into account the structure of the equality and/or

inequality constraints defining the set over which the optimization has to be performed, is based on the theory of Lagrange multipliers [3, Chap. 3]. This leads to the *barrier and interior point methods*, to the *penalty and augmented Lagrangian methods* and the associated *sequential quadratic programming*, and to the *Lagrangian and primal–dual interior point methods* [3, Chap. 4].

Remark 5.2 It is important to point out that the constraint set W_n on the “free” parameter vector \mathbf{w}_n has more a theoretical than a practical meaning. Indeed, W_n is, throughout the book, a very convenient tool to state the FDO or NLP problems to which the basic IDO Problem P has been reduced. Unfortunately, in general, W_n is a very hard set to compute. Consequently, we shall come back on the issue of constraints when we have to deal with the numerical examples of IDO problems. In such circumstances, it should be easier to examine how to behave in the presence of constraints. \triangleleft

Clearly, if the interchange property stated by (5.28) is valid, then the MC and quasi-MC estimates of the integrals in (5.28a) and (5.28b) provide the same kind of approximation of the gradient, that is,

$$\nabla \tilde{F}[\mathbf{w}_n(k)] \simeq \nabla_{\mathbf{w}_n} \frac{1}{L} \sum_{l=1}^L J[\mathbf{w}_n(k), \mathbf{z}_l(k)] = \frac{1}{L} \sum_{l=1}^L \nabla_{\mathbf{w}_n} J[\mathbf{w}_n(k), \mathbf{z}_l(k)]. \quad (5.32)$$

The vectors $\nabla_{\mathbf{w}_n} J[\mathbf{w}_n(k), \mathbf{z}_l(k)]$ are obtained by computing the gradient $\nabla_{\mathbf{w}_n} J[\mathbf{w}_n(k), \mathbf{z}(k)]$ in correspondence with the vectors of the sample set $\{\mathbf{z}_l(k)\}_{l=1}^L$ generated at the iteration k according to the previously explained MC and quasi-MC rules.

If we do not change the sample set at any iteration k , we obtain an approximate solution of Problem P_n that depends on the set $Z_L \triangleq \{\mathbf{z}_l, l = 1, \dots, L\}$ (see (4.1)) or the sequence $\{\mathbf{z}^L\} \triangleq \{\mathbf{z}_l\}_{l=1}^L$ (see (4.2)), where the vectors are no more indexed by k . Problem P_n is then solved by a deterministic gradient-based algorithm. For example, if we consider the simple steepest descent method, the use of the fixed-sample set yields the iterative procedure

$$\mathbf{w}_n(k+1) = \mathbf{w}_n(k) - \alpha_k \frac{1}{L} \sum_{l=1}^L \nabla_{\mathbf{w}_n} J[\mathbf{w}_n(k), \mathbf{z}_l], \quad k = 0, 1, \dots. \quad (5.33)$$

This way of using the data is called *batch processing mode* as an algorithm like (5.33) passes repeatedly through the same set $\{\mathbf{z}_l(k)\}_{l=1}^L$ and the gradients in (5.33) are recomputed at iteration k with \mathbf{w}_n changing at each step of the iterating procedure.

An opposite situation occurs when only one random vector $\mathbf{z}(k)$ is considered at iteration k . The related iterative procedure, obtained by setting $L = 1$ in (5.33), is called *sequential* or *by-pattern mode* (the latter term is used also in the context of neural networks). It gives rise to the *stochastic gradient algorithm* that is described in some detail in Sects. 5.3.4, 5.3.8, and later on. This algorithm is the most frequently used to solve Problem P_n throughout the book.

Finally, it is worth noting what follows. It may occur that $J(\mathbf{w}_n, z)$ is a function of class \mathcal{C}^1 with respect to \mathbf{w}_n and z and can be evaluated exactly but that the gradient $\nabla_{\mathbf{w}_n} J(\mathbf{w}_n, z)$ is given by a very complicated expression. In this case, we may use a direct search method. In many cases, however, it may be preferable to resort to gradient-based algorithms with the unavailable derivatives approximated by *finite differences* (see, for example, [3, p. 159]). If $\nabla_{\mathbf{w}_n} J(\mathbf{w}_n, z)$ can be computed in a “sufficiently accurate way,” then we can say that Assumption 2.4 is verified. Usually, commercial packages for optimization offer the possibility to provide the gradient function or, in alternative, they calculate the gradient themselves by some finite-difference technique.

5.3.3 Stochastic Approximation: Robbins–Monro Algorithm for the Solution of Nonlinear Root-Finding Problems

The drawbacks of the batch methods presented in the previous section are clearly due to the necessity of computing integrals approximately. In this section and in the following ones, we consider another method which enables one to minimize the cost function by Stochastic Gradient Algorithm (2.37), and avoid the approximate computation of integrals. The new technique is a particular application of *stochastic approximation*. This method aims at solving nonlinear root-finding problems in the presence of noisy disturbances and is generally called *Robbins–Monro algorithm*, in honor of the two people who first proposed it.

Root-finding SA provides a general framework for the analysis of convergence of many stochastic algorithms that, at first sight, seem to have nothing to do with root-finding methods. Following [48], we mention recursive least-squares and least mean squares algorithms, nonlinear parameter estimation methods, simulated annealing, evolutionary computation, simulation-based optimization, reinforcement learning, and more. We refer, for instance, to [48] for a description of such methods.

In order to introduce Robbins–Monro’s idea, in [30, p. 3] the following problem is stated: find the root θ^* of the equation

$$g(\theta) = 0, \quad (5.34)$$

where $g: \mathbb{R} \rightarrow \mathbb{R}$ is an unknown nonlinear function. We assume g to be increasing in a neighborhood of θ^* (if it was decreasing, we would change its sign). All this justifies the minus sign on the right-hand side in Algorithms (5.40) and (5.43) presented later on. We suppose that noisy observations of $g(\theta)$

$$y = g(\theta) + \eta \quad (5.35)$$

can be obtained in correspondence with any selected value of θ , where η is a noise term.

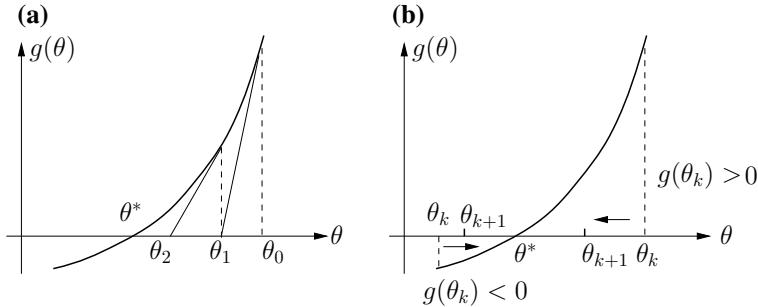


Fig. 5.2 **a** Steps of Algorithm (5.36). **b** Steps of Algorithm (5.37) (g is assumed to be increasing)

Of course, if the function $g(\theta)$ was known and continuously differentiable, we could use the classical Newton's procedure. Let g be bounded in the neighborhood of θ^* with (strictly) positive derivative g' . Then, the procedure generates an estimate θ_k obtained recursively by

$$\theta_{k+1} = \theta_k - \frac{1}{g'(\theta_k)} g(\theta_k), \quad k = 0, 1, \dots, \quad (5.36)$$

starting from an initial guess θ_0 (see Fig. 5.2). If θ_0 belongs to a small neighborhood of θ^* , then θ_k converges to θ^* for $k \rightarrow \infty$.

An alternative algorithm is the following:

$$\theta_{k+1} = \theta_k - \alpha g(\theta_k), \quad k = 0, 1, \dots, \quad (5.37)$$

where $\alpha > 0$ is a constant stepsize chosen sufficiently small so as to guarantee convergence if θ_0 is sufficiently close to θ^* (see Fig. 5.2b). Note that the sign of $g(\theta_k)$ (together with the sign of the derivative $g'(\theta_k)$ in the case of (5.36)) specifies whether θ_{k+1} has to increase or decrease, thus approaching the root θ^* . Of course, Algorithm (5.37) is simpler than (5.36) as it does not use the derivative (g is not required to be differentiable) but in general its convergence is slower.

As the functions $g(\theta)$ and $g'(\theta)$ are unknown, one cannot use Algorithms (5.36) and (5.37). It may seem quite natural to resort again to Procedure (5.37) by replacing $g(\theta_k)$ with a “good” estimate obtained by averaging a sufficiently large number L of observations $y_k = g(\theta_k) + \eta_k$. This means that, at iteration k , one takes L observations $y_k^{(1)}, \dots, y_k^{(L)}$ and computes the sample mean (see (5.35))

$$g_L(\theta_k) \triangleq \frac{1}{L} \sum_{l=1}^L y_k^{(l)} = g(\theta_k) + \frac{1}{L} \sum_{l=1}^L \eta_k^{(l)}. \quad (5.38)$$

If $E(\eta) = 0$, then $g_L(\theta_k)$ is an unbiased estimate of $g(\theta_k)$ and it can be used to replace $g(\theta_k)$ in Algorithm (5.37), thus getting

$$\theta_{k+1} = \theta_k - \alpha_k g_L(\theta_k), \quad k = 0, 1, \dots . \quad (5.39)$$

The idea of Robbins and Monro [42] consists in avoiding the computation of averages at any step and simply using a single noisy observation y_k (let $L = 1$ in (5.38)). Therefore, Algorithm (5.37) is replaced by

$$\theta_{k+1} = \theta_k - \alpha_k y_k, \quad k = 0, 1, \dots , \quad (5.40)$$

where the stepsize $\alpha_k > 0$ has to decrease suitably as we shall better specify. Indeed, Robbins and Monro recognized that computing a “good” average at any iteration k is a wasteful use of observations. The average can be obtained *across iterations*, instead, while new observations are acquired.

The same reasoning carried out for Eq. (5.34) can be repeated for a vector $\theta \in \mathbb{R}^p$, when one aims at finding a root θ^* of

$$g(\theta) = \mathbf{0}, \quad (5.41)$$

where $g: \mathbb{R}^p \rightarrow \mathbb{R}^p$. Equation (5.41) is a classic algebraic system of p equations and p unknowns. Each component of g is supposed to be increasing like in the scalar case; if some component is decreasing, its sign has to be changed. Again, we assume that the function $g(\theta)$ is unknown and that noisy observations of $g(\theta)$ can be taken at any value of θ . Such observations are given by (see (5.35))

$$y_k = g(\theta_k) + \eta_k, \quad k = 0, 1, \dots . \quad (5.42)$$

So, Algorithm (5.40) becomes

$$\theta_{k+1} = \theta_k - \alpha_k y_k, \quad k = 0, 1, \dots . \quad (5.43)$$

Remark 5.3 Clearly, Batch Algorithm (5.33) requires the data set Z_L (or the sequence $\{z^L\}$) to be known a priori. Of course, the number L of samples has to be sufficiently large so that the summation in the right-hand sides of (5.32) and (5.33) constitutes a good approximation of the cost $\tilde{F}(\mathbf{w}_n)$ (see (5.27)). If new data are acquired after the instant L , the vector \mathbf{w}_n (which obviously takes the place of θ) must be recomputed from the beginning. Algorithm (5.43) and the algorithms based on it (we shall consider them in the next sections), on the contrary, are performed step by step as soon as new observations (5.42) become available. Therefore, recursive algorithms of type (5.43) are typically executed online and are well suited in possibly time-varying adaptive contexts (suitable extensions of Problem P_n have then to be stated). This is one of the reasons why these algorithms will be mostly used in the numerical examples presented in the book. \triangleleft

Remark 5.4 In solving Problem P_n in the presence of constraints, one can apply the algorithms for constrained optimization mentioned in Sects. 5.3.1 and 5.3.2. Indeed, the cost function is specified on the right-hand side of (5.33) and we consider that

Assumptions 2.4 and 2.5 are verified. Of course, as regards the SA techniques presented in the following sections, things are more complex if there is any constraint. [48, p. 98] addresses this issue as follows. Replace the vector \mathbf{w}_n with $\boldsymbol{\theta}_k$ and the constraint set W_n with Θ . Then, Algorithm (5.43) takes on the form

$$\boldsymbol{\theta}_{k+1} = \Psi_\Theta(\boldsymbol{\theta}_k - \alpha_k \mathbf{y}_k), \quad k = 0, 1, \dots, \quad (5.44)$$

where Ψ_Θ is a “suitable” mapping, chosen by the user, that projects any vector, falling outside Θ , into the constraint set. Algorithm (5.44) has only a formal importance as all the difficulties, which affect constrained optimization, are even more complex in SA unless Θ is a “simple set” (this is, e.g., the case of a hypercube, where the components of $\boldsymbol{\theta}$ have independent upper and lower bounds). Indeed, as reported in [48], the majority of practical and theoretical results refer to the unconstrained version of (5.43). \triangleleft

Note that the noises $\boldsymbol{\eta}_0, \boldsymbol{\eta}_1, \dots$ may have general statistical properties. For example, their density functions may depend on k and $\boldsymbol{\theta}_k$ (i.e., $p_k(\boldsymbol{\eta}_k | \boldsymbol{\theta}_k)$). They may also be mutually dependent and/or characterized by nonidentical densities. However, in many cases addressed in the following sections, they are i.i.d. zero mean random vectors.

After Robbins and Monro’s seminal paper [42], a large number of works followed developing the concept of SA. One of the first extensive studies is reported in [50]. See [2, 30] for an in-depth analysis from both theoretical and application standpoints, and [9] for numerical solving techniques and case studies. See also [48] for a review on stochastic search methods in optimization.

In the next section, we describe a general stochastic algorithm that embeds Procedure (5.43).

5.3.4 *Robbins–Monro Algorithm and the Stochastic Gradient Method are Particular Cases of a General Stochastic Algorithm*

In this section, we address a very general stochastic algorithm of the form

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \alpha_k \mathbf{s}_k, \quad k = 0, 1, \dots, \quad (5.45)$$

where $-\mathbf{s}_k$ is a certain random direction of motion in the space of the vectors $\boldsymbol{\theta} \in \mathbb{R}^p$, which will be specified case-by-case. Algorithm (5.45) includes Robbins–Monro’s recursive procedure and the *stochastic gradient algorithm* (2.37) aimed at solving Problem P_n. Clearly, these two algorithms are strictly connected to each other. This can be seen by writing the necessary optimality conditions for a differentiable function $\tilde{F}(\boldsymbol{\theta})$ in the form of a system of equations, i.e., (see (5.41)),

$$\mathbf{g}(\boldsymbol{\theta}) \triangleq \nabla \tilde{F}(\boldsymbol{\theta}) = \mathbf{0}. \quad (5.46)$$

If the values of the function $\mathbf{g}(\boldsymbol{\theta})$ are known through the noisy observations (5.42), then we can solve System (5.41) by Robbins–Monro’s SA procedure (5.43), which specializes Algorithm (5.45) with

$$\mathbf{s}_k = \mathbf{y}_k = \mathbf{g}(\boldsymbol{\theta}_k) + \boldsymbol{\eta}_k. \quad (5.47)$$

As regards the minimization of the cost function $\tilde{F}(\boldsymbol{\theta})$, we consider “Problem P_n in the forms (2.11) and (2.31),” where the cost function is given by $\tilde{F}(\mathbf{w}_n)$ and $E_z[\tilde{J}(\mathbf{w}_n, z)]$, respectively (see (2.30)). Let $\boldsymbol{\theta}_n = \mathbf{w}_n$ and drop the subscript n . Then, the necessary condition for optimality is given by²

$$\mathbf{g}(\mathbf{w}) = \nabla \tilde{F}(\mathbf{w}) = \nabla_{\mathbf{w}} E_z[\tilde{J}(\mathbf{w}, z)] = \mathbf{0}. \quad (5.48)$$

As pointed out in this chapter and in Sect. 2.3, computing analytically (and even numerically) the expected value $E_z[J(\mathbf{w}, z)]$ is in general quite a difficult task. Thus, the root-finding algorithm

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \alpha_k \mathbf{g}[\mathbf{w}(k)], \quad k = 0, 1, \dots, \quad (5.49)$$

cannot be applied. However, taking into account the fact that the *stochastic gradient* $\nabla_{\mathbf{w}} J(\mathbf{w}, z)$ is an unbiased estimate of the *deterministic gradient* $\nabla \tilde{F}(\mathbf{w})$ (see (2.35)), we resort to the *stochastic gradient algorithm* (2.37) that we rewrite here

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \alpha_k \nabla_{\mathbf{w}} \tilde{J}[\mathbf{w}(k), z(k)], \quad k = 0, 1, \dots. \quad (5.50)$$

Apparently, the noise $\boldsymbol{\eta}_k$ may seem not to be present in Algorithm (5.50). However, we must keep in mind that (5.50) is an approximation of (5.49). Then, in order to understand the meaning of $\boldsymbol{\eta}_k$, we write

$$\begin{aligned} \mathbf{w}(k+1) &= \mathbf{w}(k) - \alpha_k \nabla_{\mathbf{w}} \tilde{J}[\mathbf{w}(k), z(k)] + \alpha_k \nabla \tilde{F}[\mathbf{w}(k)] - \alpha_k \nabla \tilde{F}[\mathbf{w}(k)] \\ &= \mathbf{w}(k) - \alpha_k \nabla \tilde{F}[\mathbf{w}(k)] + \boldsymbol{\eta}_k, \quad k = 0, 1, \dots, \end{aligned} \quad (5.51)$$

where

$$\boldsymbol{\eta}_k \triangleq \alpha_k \left\{ \nabla \tilde{F}[\mathbf{w}(k)] - \nabla_{\mathbf{w}} \tilde{J}[\mathbf{w}(k), z(k)] \right\}. \quad (5.52)$$

²To keep clear the notation, in the following examples we denote by $\boldsymbol{\theta}_k$ the variables related to the various SA problems and by $\mathbf{w}(k)$ the iterates of the gradient algorithms somehow connected with Problem P_n . Of course, the variables $\mathbf{w}(k)$ are special cases of the variables $\boldsymbol{\theta}_k$.

Definition (5.52) shows that the random variable η_k is originated from the approximation of the deterministic gradient $\nabla \tilde{F}[\mathbf{w}(k)]$ by the stochastic gradient $\nabla_{\mathbf{w}} \tilde{J}[\mathbf{w}(k), z(k)]$.

Note that, by using the notation

$$\nabla \tilde{F}[\mathbf{w}(k)] = \underset{z}{\mathbb{E}} \{\nabla_{\mathbf{w}} J[\mathbf{w}(k), z(k)]\}, \quad (5.53)$$

we have assumed that the expected value is not conditioned by the values of \mathbf{w} , z at the preceding iterations and by the stepsize. Such values would make up the “history” of Algorithm (5.50) at the iteration k . In the next section, we shall consider such assumption (which is rather realistic, in practice) in some detail.

Again, we can say that Stochastic Gradient Algorithm (5.50) is a particular case of the general SA algorithm (5.45) with

$$\mathbf{s}_k = \nabla_{\mathbf{w}} \tilde{J}[\mathbf{w}(k), z(k)].$$

Other examples of problems solvable by algorithms of type (5.45) can be found in [4, Sect. 4.2.1] and [48, Sect. 4.2]. Note that sometimes the term “stochastic gradient algorithm” denotes a procedure that is slightly different from Algorithm (5.50). For instance, Example 4.2 in [4] describes the algorithm

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \alpha_k \left\{ \nabla h[\mathbf{w}(k)] + \boldsymbol{\eta}_k \right\}, \quad k = 0, 1, \dots, \quad (5.54)$$

aimed at minimizing the cost function $h(\mathbf{w})$. Of course, (5.54) is a particular case of (5.45) with

$$\mathbf{s}_k = \nabla h[\mathbf{w}(k)] + \boldsymbol{\eta}_k.$$

Unlike Algorithm (5.51), Algorithm (5.54) shows “true” random noises $\boldsymbol{\eta}_k$ of the gradient of the cost $h(\mathbf{w})$.

Under suitable assumptions, Algorithms (5.43), (5.50), and (5.54) enjoy the same convergence properties as (5.45). Such properties are described in the next section.

5.3.5 Convergence of the General Stochastic Algorithm

The conditions for convergence of Stochastic Algorithm (5.45) given in the following examples are illustrated in [4]. Other similar conditions are considered in the literature; see, for example, [48, Sect. 4.3.2]. Note also that the SA convergence conditions are *sufficient* conditions. Moreover, many practical applications of SA give satisfactory results even if some conditions are weakened.

It is important to point out that in [4] the basic root-finding Robbins–Monro algorithm aims at solving the vector equation

$$\tilde{g}(\theta) = \theta \quad (5.55)$$

(where $\tilde{g}: \mathbb{R}^p \rightarrow \mathbb{R}^p$) instead of (5.41) that we report here for convenience:

$$g(\theta) = \mathbf{0}. \quad (5.56)$$

Clearly, (5.55) and (5.56) are equivalent if one defines

$$g(\theta) \triangleq -\tilde{g}(\theta) + \theta.$$

If (5.56) is dealt with, then the noisy measures are given by (5.42) and, in the SA algorithm (5.45), the vector s_k takes on the form

$$s_k = g(\theta_k) + \eta_k, \quad k = 0, 1, \dots.$$

Instead, if (5.55) is considered, we have

$$s_k = -\tilde{g}(\theta_k) + \theta_k + \eta_k = g(\theta_k) + \eta_k, \quad k = 0, 1, \dots.$$

Systems (5.55) or (5.56) are chosen depending on the problem at hand.

Let us now address Algorithm (5.45) in a deterministic context, i.e., with the vector s_k generated by a deterministic law. More specifically, we consider a NLP problem in which a differentiable cost function $h(\theta)$ has to be minimized. Then, $-s_k$ is a *descent direction*, i.e., it has to verify the condition

$$-\nabla h(\theta_k)^\top s_k < 0 \quad (5.57)$$

for any θ_k such that $\nabla h(\theta_k)^\top \neq \mathbf{0}$. Condition (5.57) guarantees that $h(\theta_{k+1}) < h(\theta_k)$ if α_k is sufficiently small.

A common technique for establishing the convergence of a recursive algorithm of the type (5.45) to some limit vector θ^* consists in measuring the “distance” between an iterate θ_k and θ^* by some *Lyapunov* or *potential function*. Denote this function by $C(\theta)$. In general, $C(\theta)$ has an “instrumental” meaning. In a NLP problem, it may coincide with the cost function to be minimized. For NLP problems with non-differentiable cost functions, a smooth function $C(\theta)$ can be used. Let $C(\theta)$ be continuously differentiable. When s_k is a random vector, it would be too restrictive to require that Condition (5.57) is verified for any random direction s_k . Instead, it is reasonable to require that

$$\nabla C(\theta_k)^\top E(s_k | \mathcal{F}_k) > 0, \quad k = 0, 1, \dots, \quad (5.58)$$

where the vector \mathcal{F}_k denotes the “history” of Algorithm (5.45) at iteration k , i.e., the set of all the information acquired until iteration k that is useful to determine the expected value of s_k . If all the past variables of (5.45) have to be stored, then a complete history is given by

$$\mathcal{F}_k \triangleq \text{col} (\boldsymbol{\theta}_0, \dots, \boldsymbol{\theta}_k, s_0, \dots, s_{k-1}, \alpha_0, \dots, \alpha_k) . \quad (5.59)$$

Condition (5.58) means that on average s_k should form an acute angle with the gradient $\nabla C(\boldsymbol{\theta}_k)$ or, equivalently, that $-s_k$ should be on average a direction of decrease in the cost $C(\boldsymbol{\theta})$.

We now state the following theorem (see [4, Proposition 4.1]), which gives sufficient conditions for the convergence of Algorithm (5.45).

Theorem 5.3 *Let the function $C(\boldsymbol{\theta})$ satisfy the following properties:*

- (a) $C(\boldsymbol{\theta}) \geq 0, \forall \boldsymbol{\theta} \in \mathbb{R}^p$.
- (b) (Lipschitz continuity of $\nabla C(\boldsymbol{\theta})$) $C(\boldsymbol{\theta})$ is continuously differentiable and there is some Lipschitz constant L such that

$$|\nabla C(\boldsymbol{\theta}) - \nabla C(\boldsymbol{\theta}')| \leq L |\boldsymbol{\theta} - \boldsymbol{\theta}'|, \quad \forall \boldsymbol{\theta}, \boldsymbol{\theta}' \in \mathbb{R}^p . \quad (5.60)$$

- (c) (Pseudogradient property) There exists a positive constant c such that

$$\nabla C(\boldsymbol{\theta}_k)^\top E(s_k | \mathcal{F}_k) \geq c |\nabla C(\boldsymbol{\theta}_k)|^2, \quad k = 0, 1, \dots . \quad (5.61)$$

- (d) There exist positive constants K_1, K_2 such that

$$E(|s_k|^2 | \mathcal{F}_k) \leq K_1 + K_2 |\nabla C(\boldsymbol{\theta}_k)|^2, \quad k = 0, 1, \dots . \quad (5.62)$$

Assume that the stepsizes α_k are nonnegative and satisfy the conditions

(e)

$$\sum_{k=0}^{\infty} \alpha_k = \infty . \quad (5.63)$$

(f)

$$\sum_{k=0}^{\infty} \alpha_k^2 < \infty . \quad (5.64)$$

Then the following holds with probability 1:

- (i) The sequence $\{C(\boldsymbol{\theta}_k)\}_{k=0}^{\infty}$ converges.
- (ii) $\lim_{k \rightarrow \infty} \nabla C(\boldsymbol{\theta}_k) = \mathbf{0}$.
- (iii) Every limit point $\boldsymbol{\theta}^*$ of the sequence $\{\boldsymbol{\theta}_k\}_{k=0}^{\infty}$ is a stationary point of $C(\boldsymbol{\theta})$, i.e., $\nabla C(\boldsymbol{\theta}^*) = \mathbf{0}$.

□

Different sets of conditions for the a.s. convergence of the SA iterates are given by various authors. See, e.g., [48, pp. 106, 107] and [30, Chaps. 5 and 6].

Note that the convergence of the sequence $\{C(\theta_k)\}_{k=0}^{\infty}$ (see item (i) of Theorem 5.3) does not imply the convergence or the boundedness of the sequence $\{\theta_k\}_{k=0}^{\infty}$. Of course, if the potential function $C(\theta)$ has compact sublevel sets, then $\{\theta_k\}_{k=0}^{\infty}$ is bounded. Moreover, if $C(\theta)$ has a single stationary point θ^* and $\{\theta_k\}_{k=0}^{\infty}$ has limit points, then Item (iii) states that θ^* is the only limit point of $\{\theta_k\}_{k=0}^{\infty}$. Then, $\lim_{k \rightarrow \infty} \theta_k = \theta^*$ with probability 1.

Assumption (b) of Theorem 5.3 is a smoothness condition on the function $C(\theta)$. In particular, it is verified if $C(\theta)$ is twice differentiable and its second derivatives are globally bounded by some constant. Condition (5.61) is stronger than Condition (5.58). Likewise in the case of deterministic gradient-based algorithms, it is a technical condition for demonstrating the convergence properties of Algorithm (5.45). Assumption (d) imposes a bound to the mean square magnitude of s_k . In general, Assumptions (b), (c), and (d) can be considered as extensions to the stochastic framework of similar conditions, which establish the convergence of deterministic gradient-based algorithms (see Propositions 3.4 and 3.5 in [4]).

Assumptions (e) and (f) are consistent with intuition. Actually, Condition (5.64) implies that

$$\lim_{k \rightarrow \infty} \alpha_k = 0. \quad (5.65)$$

Condition (5.65) is needed for the convergence of Algorithm (5.45) as, at the limit point θ^* , s_k may be nonzero even if $\nabla C(\theta^*)$ is zero (see (5.62)). On the other hand, Condition (5.63) should prevent α_k from becoming too small too early, thus preventing Algorithm (5.45) to make further useful progress. For example, if the norm of s_k is bounded by a scalar b and if

$$\sum_{k=0}^{\infty} \alpha_k \leq B < \infty,$$

we have

$$\theta_k = \theta_0 - \sum_{i=0}^{k-1} \alpha_i s_i$$

and

$$|\theta_k - \theta_0| \leq \sum_{i=0}^{k-1} \alpha_i |s_i| \leq bB, \quad k = 1, 2, \dots.$$

Then, θ_k is confined to a ball of center θ_0 and radius bB . If the stationary point θ^* is outside this ball, then the algorithm will never reach θ^* . This motivates Assumption (e).

In order to motivate Assumption (f), we consider a simple problem inspired by [4, Example 4.1, p. 136]. Let v_k , $k = 0, 1, \dots$, be i.i.d. samples of a scalar random variable v with mean $E(v)$ that we want to estimate on the basis of the variables v_k ; v has unitary variance. This is the same as solving the equation $E(v) = \theta$ or, equivalently,

$$g(\theta) \triangleq -E(v) + \theta = 0.$$

Of course, Algorithm (5.36) cannot be used since $E(v)$ is unknown. However, the problem can be considered in Robbins–Monro’s framework by introducing noisy measures of $g(\theta)$, that is,

$$y_k = E(v) - v_k - E(v) + \theta_k = g(\theta_k) + \eta_k, \quad k = 0, 1, \dots,$$

where

$$\eta_k = E(v) - v_k.$$

Then, we can resort to Algorithm (5.43) and write

$$\theta_{k+1} = \theta_k - \alpha_k y_k = \theta_k - \alpha_k (\theta_k - v_k), \quad k = 0, 1, \dots. \quad (5.66)$$

From (5.66), it can be seen that for θ_k to converge to $E(v)$, $\alpha_k [E(v) - v_k]$ has to converge to zero. Assumption (f) is a sufficient condition for the latter convergence since we can write

$$E \left\{ \sum_{k=0}^{\infty} \alpha_k^2 [E(v) - v_k]^2 \right\} = \sum_{k=0}^{\infty} \alpha_k^2 E \{ [E(v) - v_k]^2 \} = \sum_{k=0}^{\infty} \alpha_k^2 < \infty, \quad (5.67)$$

where we have exploited the fact that v has a unitary variance.

It is worth noting that in (5.67) the interchange of the infinite summation and the expectation is a consequence of the monotone convergence theorem (see, e.g., [48]). According to (5.67), the first term is finite. It follows that the random variable $\sum_{k=0}^{\infty} \alpha_k^2 [E(v) - v_k]^2$ is finite with probability 1; otherwise, its expected value would be infinite. Thus, $\alpha_k [E(v) - v_k]$ converges to zero with probability 1.

5.3.6 Convergence of the Root-Finding Robbins–Monro Stochastic Approximation Algorithm

As we said in Sect. 5.3.3, Robbins–Monro’s idea consists in giving up obtaining a wasteful estimate of $g(\theta_k)$ provided by the sample mean (5.38) with a suitably large value of L , and in simply using Algorithm (5.43).

Conditions for the convergence of SA Algorithm (5.42) are given in [48, Sect. 4.3.2]. However, to remain in the context of Theorem 5.3, we follow [4, p. 146]. Then, we replace System (5.41) with

$$\tilde{g}(\theta) = \theta, \quad (5.68)$$

where

$$\tilde{g}(\boldsymbol{\theta}) \triangleq -g(\boldsymbol{\theta}) + \boldsymbol{\theta}. \quad (5.69)$$

Now, we assume a root $\boldsymbol{\theta}^*$ of (5.55) or (5.56) to exist that satisfies the condition

$$|\tilde{g}(\boldsymbol{\theta}) - \boldsymbol{\theta}^*| \leq \beta |\boldsymbol{\theta} - \boldsymbol{\theta}^*|, \quad (5.70)$$

for any $\boldsymbol{\theta} \in \mathbb{R}^p$; β is a constant with $0 \leq \beta < 1$.

Let us introduce the potential function

$$C(\boldsymbol{\theta}) \triangleq \frac{1}{2} |\boldsymbol{\theta} - \boldsymbol{\theta}^*|^2,$$

which obviously satisfies Assumption (a) of Theorem 5.3. We have

$$\nabla C(\boldsymbol{\theta}) = \boldsymbol{\theta} - \boldsymbol{\theta}^*. \quad (5.71)$$

Hence, Assumption (b) of Theorem 5.3 is satisfied with $L = 1$.

The noisy measures are given by (see (5.47))

$$\mathbf{y}_k = g(\boldsymbol{\theta}_k) + \boldsymbol{\eta}_k = -\tilde{g}(\boldsymbol{\theta}_k) + \boldsymbol{\theta}_k + \boldsymbol{\eta}_k, \quad k = 0, 1, \dots. \quad (5.72)$$

Assume that

$$\mathbb{E}(\boldsymbol{\eta}_k | \mathcal{F}_k) = \mathbf{0}, \quad k = 0, 1, \dots. \quad (5.73)$$

Therefore, from (5.72) and (5.73), we have

$$\mathbb{E}(\mathbf{s}_k | \mathcal{F}_k) = g(\boldsymbol{\theta}_k) = -\tilde{g}(\boldsymbol{\theta}_k) + \boldsymbol{\theta}_k. \quad (5.74)$$

From the Cauchy–Schwartz inequality and (5.70), it follows that

$$[\tilde{g}(\boldsymbol{\theta}_k) - \boldsymbol{\theta}^*]^\top (\boldsymbol{\theta}_k - \boldsymbol{\theta}^*) \leq |\tilde{g}(\boldsymbol{\theta}_k) - \boldsymbol{\theta}^*| \cdot |\boldsymbol{\theta}_k - \boldsymbol{\theta}^*| \leq \beta |\boldsymbol{\theta}_k - \boldsymbol{\theta}^*|^2. \quad (5.75)$$

Subtracting $(\boldsymbol{\theta}_k - \boldsymbol{\theta}^*)^\top (\boldsymbol{\theta}_k - \boldsymbol{\theta}^*)$ from both sides of (5.75) yields

$$[\tilde{g}(\boldsymbol{\theta}_k) - \boldsymbol{\theta}_k]^\top (\boldsymbol{\theta}_k - \boldsymbol{\theta}^*) \leq -(1 - \beta) |\boldsymbol{\theta}_k - \boldsymbol{\theta}^*|^2. \quad (5.76)$$

Using (5.71) and (5.74), Inequality (5.76) becomes

$$-\mathbb{E}(\mathbf{s}_k | \mathcal{F}_k)^\top \nabla C(\boldsymbol{\theta}_k) \leq -(1 - \beta) |\boldsymbol{\theta}_k - \boldsymbol{\theta}^*|^2.$$

Hence,

$$\mathbb{E}(\mathbf{s}_k | \mathcal{F}_k)^\top \nabla C(\boldsymbol{\theta}_k) \geq c |\boldsymbol{\theta}_k - \boldsymbol{\theta}^*|^2,$$

which is Assumption (c) of Theorem 5.3 with $c = (1 - \beta)$.

Let us now consider Assumption (d). From (5.72) and (5.73), we can write

$$\mathrm{E}(|\mathbf{s}_k|^2 | \mathcal{F}_k) = |\mathbf{g}(\boldsymbol{\theta}_k)|^2 + \mathrm{E}(|\boldsymbol{\eta}_k|^2 | \mathcal{F}_k). \quad (5.77)$$

Using the triangle inequality and (5.70), we have

$$|\mathbf{g}(\boldsymbol{\theta})| = |- \tilde{\mathbf{g}}(\boldsymbol{\theta}) + \boldsymbol{\theta}| \leq |- \tilde{\mathbf{g}}(\boldsymbol{\theta}) + \boldsymbol{\theta}^*| + |\boldsymbol{\theta} - \boldsymbol{\theta}^*| \leq \beta |\boldsymbol{\theta} - \boldsymbol{\theta}^*| + |\boldsymbol{\theta} - \boldsymbol{\theta}^*|.$$

Then,

$$|\mathbf{g}(\boldsymbol{\theta}_k)|^2 \leq (1 + \beta)^2 |\boldsymbol{\theta}_k - \boldsymbol{\theta}^*|^2. \quad (5.78)$$

Let us assume that

$$\mathrm{E}(|\boldsymbol{\eta}_k|^2 | \mathcal{F}_k) \leq a + b |\boldsymbol{\theta}_k - \boldsymbol{\theta}^*|^2 \quad (5.79)$$

for some positive scalars a and b . From (5.71), (5.77), (5.78), and (5.79), we obtain

$$\mathrm{E}(|\mathbf{s}_k|^2 | \mathcal{F}_k) \leq a + [b + (1 + \beta)^2] |\nabla C(\boldsymbol{\theta}_k)|^2,$$

which is Assumption (d) of Theorem 5.3 with $K_1 = a$ and $K_2 = b + (1 + \beta)^2$. Concluding, if also Assumptions (e) and (f) are verified, then Theorem 5.3 can be applied to this case.

5.3.7 Choice of the Stepsize Sequence

Clearly, the choice of the rule that generates the stepsize sequence $\{\alpha_k\}_{k=0}^\infty$ is a basic issue for the performance of the SA algorithm. In this section, we report some concepts taken from [48, Sect. 4.4], which also provides several references on this subject.

So far we have considered the convergence of SA algorithms but nothing has been said about the speed with which the iterate $\boldsymbol{\theta}_k$ approaches $\boldsymbol{\theta}^*$. In this case, the probability distribution of the iterates plays an important role. Under appropriate regularity conditions, it can be shown that

$$k^{a/2} (\boldsymbol{\theta}_k - \boldsymbol{\theta}^*) \xrightarrow{\text{dist}} \mathcal{N}(\mathbf{0}, \Sigma) \quad \text{as } k \rightarrow \infty, \quad (5.80)$$

where $\xrightarrow{\text{dist}}$ means “converges in distribution”; Σ is some covariance matrix that depends on the sequence $\{\alpha_k\}_{k=0}^\infty$ and on the Jacobian matrix of the function \mathbf{g} . The scalar $a > 0$ describes the decay rate of the stepsize sequence. An example of this sequence is given by

$$\alpha_k = \frac{c}{(k+1)^a}, \quad k = 0, 1, \dots, \quad (5.81)$$

where $c > 0$. Note that Sequence (5.81) satisfies Assumptions (e) and (f) of Theorem 5.3 for $1/2 < a \leq 1$.

Convergence (5.80) has a simple interpretation: as k goes to infinity, the probability distribution of $\boldsymbol{\theta}_k$ approximates better and better a normal distribution with mean $\boldsymbol{\theta}^*$ and covariance matrix Σ/k^a . Moreover, the term $k^{a/2}$ in (5.80) implies that $|\boldsymbol{\theta}_k - \boldsymbol{\theta}^*|$ decreases, in a stochastic sense, at a rate proportional to $k^{-a/2}$, hence, $\boldsymbol{\theta}_k$ approaches $\boldsymbol{\theta}^*$ at this rate. If we adopt Sequence (5.81), we can see that its rate of decrease is maximized for $a = 1$. Then, the maximum convergence rate of the Robbins–Monro SA algorithm is $O(1/\sqrt{k})$ in an appropriate stochastic sense.

It can be shown (minimizing a norm of Σ ; see, e.g., [2]) that the convergence rate of the Robbins–Monro algorithm (5.43) is *asymptotically* optimized if the scalar stepsize α_k is replaced by the *matrix gain*

$$A_k \triangleq \frac{1}{k+1} H(\boldsymbol{\theta}^*)^{-1}, \quad k = 0, 1, \dots, \quad (5.82)$$

where $H(\boldsymbol{\theta}) \triangleq \partial g(\boldsymbol{\theta})/\partial\boldsymbol{\theta}$ is the Jacobian matrix of $g(\boldsymbol{\theta})$. Then, SA algorithm (5.43) achieves the maximum rate of convergence if it has the form

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \frac{1}{k+1} H(\boldsymbol{\theta}^*)^{-1} \mathbf{y}_k, \quad k = 0, 1, \dots. \quad (5.83)$$

Note that, in minimizing a twice-differentiable cost function $\tilde{F}(\boldsymbol{\theta})$ (see “Problem P_n in the form (2.11)”) via the Robbins–Monro algorithm, we have $g(\boldsymbol{\theta}) = \nabla \tilde{F}(\boldsymbol{\theta})$ (see (5.48)). Then, $H(\boldsymbol{\theta})$ becomes the Jacobian matrix of the gradient $\nabla \tilde{F}(\boldsymbol{\theta})$, that is,

$$H(\boldsymbol{\theta}) = \frac{\partial}{\partial \boldsymbol{\theta}} \nabla \tilde{F}(\boldsymbol{\theta}), \quad (5.84)$$

which is the Hessian matrix of the cost $\tilde{F}(\boldsymbol{\theta})$. It follows that the matrix gain in (5.82) can be viewed as the stochastic version of the deterministic gain $H(\boldsymbol{\theta})^{-1}$ multiplying the gradient $\nabla \tilde{F}(\boldsymbol{\theta})$ in the Newton–Raphson method (5.31). The decay term $1/(k+1)$ appears only in (5.82) for the purpose of damping the noise effects.

Of course, the matrix gain in (5.82) has only a theoretical meaning since in practice one does not know $\boldsymbol{\theta}^*$ and has merely noisy measures of $g(\boldsymbol{\theta})$ or $\nabla \tilde{F}(\boldsymbol{\theta})$ for given values of $\boldsymbol{\theta}$ and not of their Jacobian matrices. Moreover, the gain A_k refers to an asymptotic result. However, (5.82) provides an ideal reference model aimed at improving the convergence rate of the SA algorithms. Some methods are characterized by the property of adaptively estimating the stepsize α_k (or the matrix gain A_k) on the basis of recent information acquired during the search process. For a survey on such adaptive algorithms, we refer the reader to [48, Sect. 4.5.2] and the works cited therein.

Coming back to Sequence (5.81), despite the asymptotic optimality for $a = 1$, the unavoidable fact that the SA algorithms cannot use but a finite number of samples may imply that values $a < 1$ yield better results. Theoretical arguments for this choice are

given, for example, in [30, Sect. 10.2]. An intuitive reason is that a slower decrease of α_k gives larger values of the stepsize for large values of k . Then, the algorithm moves in bigger steps toward θ^* .

The stepsizes generated by (5.81) can incur the following drawback: choosing a large value of c provides non-negligible stepsizes in the first iterations of the algorithm. However, the denominator is still small and this may cause an unstable initial behavior of the algorithm. On the other hand, choosing a small value of c provides a stable behavior in the early iterations but a sluggish performance for large values of k . Such a drawback can be mitigated by adding a *stability constant* $b > 0$ in the denominator of (5.81), which becomes

$$\alpha_k = \frac{c}{(k + 1 + b)^a}, \quad k = 0, 1, \dots.$$

Clearly, b stabilizes the algorithm in the early iterations, thus enabling one to choose not too small values of c . When k increases, the constant b becomes negligible with respect to k , while the stepsizes remain non-negligible.

Let us consider b as the equivalent of a certain number of iterations. For values $1/2 < \alpha \leq 1$, [48, p. 114] suggests to choose b such that it is approximately 5 to 10 percent of the total number of iterations that are expected in the search process (recall that we are in a practical context of finite-sample search).

Another practical procedure to avoid a possible sluggish behavior of the algorithms is to periodically restart them. This means that one starts from an initial guess θ_0 . Then, one periodically restarts the algorithm by letting $\theta_0 = \theta_k$ (k is the index of the last iteration of each period) and resetting the stepsize sequence at α_0 .

In the numerical examples reported later in the book, we shall use the following stepsizes in the stochastic gradient algorithms:

$$\alpha_k = \frac{c_1}{c_2 + k}, \quad k = 0, 1, \dots, \tag{5.85}$$

where $c_1, c_2 > 0$. Thus $c_1 = c$, $c_2 = b + 1$, and $a = 1$. We determined c_1 and c_2 by sometimes tedious numerical experimentation. Probably, values $a < 1$ and other variants could improve the performances of our algorithms. Actually, optimizing their effectiveness was not among our main goals.

5.3.8 Convergence of the Stochastic Gradient Algorithm

Let us now address the stochastic gradient algorithm (2.37) (or (5.50), or (5.51)) and verify under which conditions this algorithm satisfies the assumptions of Theorem 5.3 with θ_k replaced by $w(k)$. As pointed out at the end of Sect. 5.3.4, Algorithm (5.54) is sometimes also called “stochastic gradient algorithm.” In the book, we keep the term “stochastic gradient algorithm” for Method (5.50) or (5.51). Clearly, Algorithms (5.51) and (5.54) are equivalent. Indeed, it is sufficient to define

$$\tilde{\boldsymbol{\eta}}_k = -\frac{1}{\alpha_k} \boldsymbol{\eta}_k$$

and (5.51) takes on the form of (5.54), i.e.,

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \alpha_k \left\{ \nabla \tilde{F}[\mathbf{w}(k)] + \tilde{\boldsymbol{\eta}}_k \right\}, \quad k = 0, 1, \dots. \quad (5.86)$$

Of course, if the α_k are chosen deterministically, then $E(\boldsymbol{\eta}_k) = E(\tilde{\boldsymbol{\eta}}_k) = \mathbf{0}$ (see (5.52)).

The convergence of Algorithm (5.86) is proved in [4, p. 142] under suitable assumptions. Then, we exploit this demonstration to prove the convergence of Algorithms (5.50) and (5.51) as well. To simplify the computations without violating the generality of the treatment, we assume that $\mathbf{z}(k)$, $k = 0, 1, \dots$, in Algorithm (5.50) and $\boldsymbol{\eta}(k)$ and $\tilde{\boldsymbol{\eta}}(k)$, $k = 0, 1, \dots$, in Algorithms (5.51) and (5.86), respectively, are mutually independent random vectors generated on the basis of their probability density functions. Moreover, we assume that the stepsizes α_k , $k = 0, 1, \dots$, are computed a priori via a given deterministic formula. Then, Algorithms (5.50), (5.51), and (5.86) can be viewed as discrete-time dynamic systems with $\mathbf{w}(k)$ as state vector driven by the random variables $\mathbf{z}(k)$, $\boldsymbol{\eta}(k)$, and $\tilde{\boldsymbol{\eta}}(k)$, respectively. Clearly, the “history” of such systems is simply reduced to $\mathcal{F}_k = \mathbf{w}(k)$ and can be omitted in the equations of this section. This will be the most frequent situation throughout the book.

We take $\tilde{F}(\mathbf{w})$ as a potential function, i.e., $C(\mathbf{w}) = \tilde{F}(\mathbf{w})$. We assume that $\tilde{F}(\mathbf{w})$ is nonnegative and that $\nabla \tilde{F}(\mathbf{w})$ is Lipschitz-continuous. Then, Assumptions (a) and (b) of Theorem 5.3 are verified.

We further assume that positive constants C and D can be found such that

$$E(|\tilde{\boldsymbol{\eta}}_k|^2) \leq C + D |\nabla \tilde{F}[\mathbf{w}(k)]|^2. \quad (5.87)$$

From (5.86) we get

$$\mathbf{s}_k = \nabla \tilde{F}[\mathbf{w}(k)] + \boldsymbol{\eta}_k. \quad (5.88)$$

Then, taking into account that $E(\boldsymbol{\eta}_k) = \mathbf{0}$, we obtain

$$\nabla \tilde{F}[\mathbf{w}(k)]^\top E(\mathbf{s}_k) = \nabla \tilde{F}[\mathbf{w}(k)]^\top E \left\{ \nabla \tilde{F}[\mathbf{w}(k)] + \boldsymbol{\eta}_k \right\} = |\nabla \tilde{F}[\mathbf{w}(k)]|^2,$$

and Assumption (c) of Theorem 5.3 is satisfied with $c = 1$.

Using (5.87) and $E(\boldsymbol{\eta}_k) = \mathbf{0}$, Equation (5.88) provides

$$\begin{aligned} \mathbb{E}(|s(k)|^2) &= \left| \nabla \tilde{F}[\mathbf{w}(k)] \right|^2 + \mathbb{E}\left(\left| \tilde{\eta}_k \right|^2\right) + 2 \nabla \tilde{F}[\mathbf{w}(k)]^\top \cdot \mathbb{E}(\tilde{\eta}_k) \\ &\leq \left| \nabla \tilde{F}[\mathbf{w}(k)] \right|^2 + C + D \left| \nabla \tilde{F}[\mathbf{w}(k)] \right|^2, \end{aligned}$$

and Assumption (d) of Theorem 5.3 is verified with $K_1 = C$ and $K_2 = 1 + D$. If Assumptions (e) and (f) are also verified, then it follows that statements (i), (ii), and (iii) of Theorem 5.3 hold true.

5.3.8.1 Momentum

The rate of convergence of the stochastic gradient algorithm may take advantage from the use of the so-called *momentum*, which is useful for smoothing the individual gradients in (5.50). At the very beginning, the literature proposed to add the momentum term to steepest descent algorithms (see (5.30) with $\alpha(k) = \alpha$ and $S(k) = I$) with the form $\beta[\mathbf{w}(k) - \mathbf{w}(k-1)]$, where $0 \leq \beta < 1$. With this variation, the steepest descent algorithm is called *heavy ball method*. The addition of this term to the right-hand side of (5.50) gives

$$\begin{aligned} \mathbf{w}(1) &= \mathbf{w}(0) - \alpha_0 \nabla_{\mathbf{w}} \tilde{J}[\mathbf{w}(0), \mathbf{z}(0)] \\ \mathbf{w}(k+1) &= \mathbf{w}(k) - \alpha_k \nabla_{\mathbf{w}} \tilde{J}[\mathbf{w}(k), \mathbf{z}(k)] + \beta[\mathbf{w}(k) - \mathbf{w}(k-1)], \quad k = 1, 2, \dots. \end{aligned} \tag{5.89}$$

It can be seen that by moving from $\mathbf{w}(k)$ to $\mathbf{w}(k+1)$ one takes into account not only the gradient in $\mathbf{w}(k)$ but also the difference $\mathbf{w}(k) - \mathbf{w}(k-1)$ weighted by the coefficient β . More precisely, the change is influenced by the previous gradients even though in a decreasing way owing to the presence of the powers of β . This can be explained by writing (5.89) in the form

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \sum_{i=0}^k \beta^{k-i} \alpha_i \nabla_{\mathbf{w}} \tilde{J}[\mathbf{w}(i), \mathbf{z}(i)], \quad k = 0, 1, \dots. \tag{5.90}$$

Roughly speaking, the momentum term has the effect of accelerating the progress of the algorithm whenever the more recent gradients are “aligned” along the same direction. It slows down this progress, however, if the gradients are “opposite” to each other. Of course, the gradients are also influenced by the random vectors $\mathbf{z}(i)$. However, when the algorithm moves in a “narrow valley,” the momentum encourages the motion of $\mathbf{w}(k)$ to follow the direction of the valley and it smooths out the gradients that are “orthogonal” to such a direction. Furthermore, the momentum term should improve the behavior of the stochastic gradient algorithm when Problems P_n have a cost surface that is alternatively very flat and very steep. This occurs very frequently in solving Problems P_n if their costs are expressed in terms of neural networks.

This is generally reported about the addition of the momentum term to the deterministic steepest descent method, when no random vectors $z(k)$ appear in (5.89) (we have followed the comments given in [4, p. 104]). Similar considerations may hold true for the case of the stochastic gradient algorithm. For the use of the momentum in the specific case of this algorithm, see [48, p. 140].

5.3.9 Global Optimization via Stochastic Approximation

The term “global optimization” refers to the problem with the ambitious aim of minimizing or maximizing globally a function for which possibly no information is available about its structure, such as convexity, knowledge of Lipschitz properties, or other general structural properties. As such, the problem is extremely general and inherently intractable.

Among comprehensive surveys and reference books on global optimization, we refer the reader to [24, 32, 56] and to the Handbook of Global Optimization [23, 39]. The paper [40] contains a concise annotated review of the most common methodologies and software tools, together with a discussion of test problems; [14] presents an overview of the research progress in deterministic global optimization during the decade 1998–2008. For a more complete treatment of deterministic global optimization, the reader is referred to [13, 25, 51]. Of particular interest is the combination of global optimization techniques with surrogate optimization [20], for problems in which each single evaluation of the objective function is computationally expensive. In these cases, the objective function is replaced by a more easily computable “surrogate one” (e.g., an RBF network approximation). Additional computational savings can be obtained by parallelization. In the next paragraphs, we shall briefly discuss different approaches to global optimization methods and some of their major features. Then, we shall briefly address one of them, based on SA.

Roughly speaking, algorithms for global optimization can be classified into deterministic and stochastic. *Deterministic algorithms* (see, e.g., [24] for an overview) are characterized by the fact that they provide a guarantee for finding (asymptotically) a global minimum. It is known that several global optimization problems are NP-hard [52] if approached by deterministic algorithms. Indeed, it turns out that the associated computational burden can become unacceptably large for high-dimensional problems and complicated functions to be minimized.

Stochastic algorithms, instead, exploit some kind of randomness, usually in the form of a pseudorandom number generator [55]. In contrast to deterministic methods, which typically guarantee asymptotic convergence to the optimum, stochastic algorithms ensure convergence in a probabilistic sense. They are motivated by the search of a trade-off between computational effort and strength of the optimality guarantee. Basically, one relaxes the requirement for deterministic convergence to the weaker claim of converging merely in a probabilistic sense, so as to gain in computational feasibility. In other words, random search algorithms sacrifice the deterministic guarantee of optimality to quickly find a good solution. Some authors

also refer to such algorithms as *metaheuristics*. They include simulated annealing, tabu (taboo) search, genetic algorithms, evolutionary programming, particle swarm optimization, ant colony optimization, cross-entropy, stochastic approximation, multistart and clustering algorithms, shake-and-bake methods, and deformation methods.

There is a large practical interest in stochastic algorithms, which are extensively applied to a variety of global optimization problems, particularly in high dimensions and without special model structure; see, for example, [5, 6, 19, 39, 41, 48, 49, 55, 56]. Indeed, stochastic algorithms have been shown to have a potential to solve large-scale problems quite efficiently, as it is not possible for deterministic algorithms. Roughly speaking, stochastic algorithms are often capable of providing good approximate solutions quickly and easily. Moreover, they are becoming popular among practitioners, thanks to the fact that they are also quite easy to implement. The other side of the coin is that they typically have to be customized to the specific problem at hand via trial-and-error.

Different types of classifications are available for stochastic global optimization algorithms. In [46], a classification of so-called *two-phase methods* is proposed. The first phase (*exploration*) is a global search in the domain where the optimization has to be performed; the second (*exploitation*) is a local search instead. As an example, multistart methods consist in a local search algorithm (e.g., a deterministic gradient search) initialized from starting points distributed in the whole domain (typically, in a uniform way). In [57], a distinction is made between *instance-based* and *model-based* stochastic algorithms. The former generate new candidate points in the domain on the basis of the current point or of the current population of points, whereas model-based methods exploit a sampling distribution and update its parameters. Among instance-based algorithms, we mention simulated annealing, genetic algorithms, and tabu search; ant colony optimization, stochastic gradient search, and the cross-entropy method, on the contrary, are model-based algorithms.

In [45], the following distinction is made: algorithms that *assume a stochastic model* versus algorithms that *are stochastic in nature*. In the first case, the function to be minimized is considered a sample path of some stochastic process and the information on such a function is sequentially updated via Bayes' rule. To the second class belong algorithms in which the observations of the objective functions are obtained by generating random points in the domain according to a suitable sampling. The performances depend on such a sampling procedure. In the last part of this section, we shall briefly investigate an algorithm of this kind.

In general, as outlined in [45], stochastic algorithms for global optimization are made of three major steps: sampling, optimization, and a stopping criterion. In the *sampling step*, pseudorandom points are generated in the domain and the associated function values are computed. The number of points generated in the sampling step can be either fixed or adaptively determined. In the *optimization step*, a local optimization algorithm is applied in correspondence with the sampled points. The local search can be performed starting either from a selected subset of sampled points or from each of them. The *stopping criterion* is particularly critical, as it strongly influences the performance. It must stop the algorithm when the global optimum has been detected or when it is not convenient to proceed further (i.e, the improvement in

quality would not compensate for the additional computational effort). Examples of stopping criterions are fixed number of steps/function evaluations; no improvement detected during the last few iterations; the a posteriori estimate that the expected convenience in continuing the algorithm is below a certain threshold; etc. The points can be drawn in various ways, e.g., i.i.d. random vectors (typically, uniformly distributed in the domain); according to a distribution with support in a neighborhood of the current point; on the basis of a distribution that depends on previously generated points (and possibly associated function evaluations); etc. Different algorithms are obtained depending on the abovementioned choices within each step.

In the remaining of this section, we briefly consider global optimization algorithms based on SA, following [48, Sect. 8.4].

Let us start from the general SA algorithm (5.45), reported here for convenience:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \alpha_k \mathbf{s}_k, \quad k = 0, 1, \dots . \quad (5.91)$$

Suppose that a global optimum \mathbf{w}° of \tilde{F} over \mathbb{R}^p exists. Specialization of (5.91) to optimization problems via SA gives (5.51) (we let $\mathbf{w}(k) = \boldsymbol{\theta}_k$), i.e.,

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \alpha_k \nabla \tilde{F}[\mathbf{w}(k)] + \boldsymbol{\eta}_k, \quad k = 0, 1, \dots , \quad (5.92)$$

or, equivalently (see (5.86))

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \alpha_k \nabla \tilde{F}[\mathbf{w}(k)] - \alpha_k \tilde{\boldsymbol{\eta}}_k, \quad k = 0, 1, \dots . \quad (5.93)$$

To achieve (stochastic) global optimization capabilities, the basic idea discussed in this section consists in modifying the algorithm (5.93) as follows:

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \alpha_k \nabla \tilde{F}[\mathbf{w}(k)] - \alpha_k \tilde{\boldsymbol{\eta}}_k + \beta_k \boldsymbol{\zeta}_k, \quad k = 0, 1, \dots , \quad (5.94)$$

where $\boldsymbol{\zeta}_k$ is an artificial random input and β_k a coefficient that decays to zero. Typically but not necessarily, $\{\boldsymbol{\zeta}_k\}$ is an i.i.d. Gaussian sequence $\mathcal{N}(\mathbf{0}, I)$. The additional term $\beta_k \boldsymbol{\zeta}_k$ has to make the algorithm escape from local minima and possibly reach a global minimum point (of course, not in a deterministic sense).

It is natural to expect that, to this end, for $k \rightarrow \infty$ the convergence $\beta_k \rightarrow 0$ has to be sufficiently fast. However, the rate of decrease to zero should not be too high, otherwise for large values of k Algorithm (5.94) resembles (5.92) and remains unable to escape from local minima. Several authors investigated the conditions for various forms of stochastic convergence of the sequence $\{\mathbf{w}(k)\}$ to a global minimum point \mathbf{w}° , with different choices for the sequences $\{\alpha_k\}$ and $\{\beta_k\}$; see, e.g., [10, 15–17, 29]. We report here four possibilities, taken from [48, Sect. 8.4], where $\alpha > 0$, $\beta > 0$, $B > 0$, and $A \geq 0$ are parameters to be chosen:

1. $\alpha_k = \alpha/\ln(k+1)$, $\beta_k = \alpha_k$ [29];
2. $\alpha_k = \alpha/(k+1+A)$, $\beta_k = \beta/\sqrt{(k+1)\ln(k+1)}$ [15, 16];
3. $\alpha_k = \alpha/(k+1+A)^\gamma$, $\beta_k = \beta/[(k+1)^{\gamma/2}\ln(k+1)]$, $0 < \gamma < 1$, [10].

$$4. \alpha_k = \alpha/(k+1+A)^\gamma, \quad \beta_k = \beta/\sqrt{(k+1)^\gamma \ln[(k+1)^{1-\gamma} + B]}, \quad 0 < \gamma < 1 \quad [54].$$

Various convergence results to an element of the set of global minimum points have been derived together with suitable conditions on the random input sequence $\{\zeta_k\}$ and the conditions already discussed in Sect. 5.3.5 for local SA algorithms. For instance, [15, 16] proved the convergence in probability (note that such kind of convergence is weaker than the almost-sure convergence – to local minima – provided by Theorem 5.3).

It has to be remarked that the rate of convergence of Algorithm (5.94) is typically very slow (see, e.g., [16, 54]). For example, in [54] the rate $O(1/\sqrt{\ln k})$ was derived with the choices for $\{\alpha_k\}$ and $\{\beta_k\}$ in item (5.3.9)) above. This has to be contrasted with the order $O(1/k^\eta)$, $0 < \eta \leq 1/2$ holding for local SA algorithms. In [48, Sect. 8.4], however, it is pointed out that such slow rates are asymptotic results and that, in practice, better finite-sample accuracies may often hold.

5.4 Monte Carlo and Quasi-Monte Carlo Search for Minima

Search for minima of functions, and especially global search for minima, is another basic problem to which MC and quasi-MC techniques can be applied. Consider a domain $B \subseteq \mathbb{R}^d$ and a cost function $h: B \rightarrow \mathbb{R}$ and let

$$\mathbf{x}^\circ \triangleq \operatorname{argmin}_{\mathbf{x} \in B} h(\mathbf{x}). \quad (5.95)$$

The MC and quasi-MC techniques are particularly useful whenever the cost surface of the function h has a low degree of regularity. This is the case, for example, with non-differentiable functions, where gradient-based methods cannot be used and *direct search methods* are required instead. The MC and quasi-MC techniques will be briefly described in the next two sections, respectively.

5.4.1 Monte Carlo Random Search

We draw some concepts from the wide treatments reported in [48, Sect. 2.2] and [36, Chap. 6]. Using the same notation as in Chap. 4, we consider a finite sequence $\{\mathbf{x}^L\}$ formed by the first L vectors of a given realization of a sequence $\{\mathbf{x}^{(l)}\}_{l=1}^\infty$ of i.i.d. random vectors generated according to the probability distribution P that we choose on B . Usually, a uniform probability density is used. Then, we solve the problem

$$\text{find } \mathbf{x}_L^\circ = \operatorname{argmin}_{1 \leq l \leq L} h(\mathbf{x}^{(l)}). \quad (5.96)$$

A minimizing vector \mathbf{x}_L° can be determined by a batch approach or by a recursive procedure. Let us consider the *batch approach* first. All the vectors $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(L)}$ are laid down and \mathbf{x}_L° is determined by simply taking the vector that yields the minimum value of the cost.

Define $h^\circ \triangleq h(\mathbf{x}^\circ)$ and $h_L^\circ \triangleq h(\mathbf{x}_L^\circ)$. The sequence $\{h_L^\circ\}_{L=1}^\infty$ converges to h° under the conditions stated below (see Theorem 6.1 and its proof in [36, p. 148]).

Theorem 5.4 *If h is continuous on B and the probability distribution P is such that $P(A) > 0$ for any non-empty open subset A of B , then*

$$\lim_{L \rightarrow \infty} h(\mathbf{x}_L^\circ) = h^\circ$$

a.s. with respect to P .

□

In the *recursive approach*, the cost function is evaluated as soon as a new vector $\mathbf{x}^{(l)}$ is generated. Then, $h(\mathbf{x}^{(l)})$ is computed and compared with the best value of the cost function that has been previously found. The following simple procedure describes such an approach (see, e.g., [48, p. 38]).

Procedure 5.1 (“simple blind” random search)

- Choose an initial value $\mathbf{x}^{(1)} \in B$ drawn according to the chosen probability distribution P (as we said before, usually uniform on B). Compute $h(\mathbf{x}^{(1)})$. Set $\hat{\mathbf{x}}^\circ(1) = \mathbf{x}^{(1)}$ ($\hat{\mathbf{x}}^\circ(l)$ is an estimate, at iteration l , of the minimizer \mathbf{x}° defined in (5.95)). Set $l = 1$.
- (*) Generate a “new” independent vector $\mathbf{x}^{(l+1)} \in B$ according to P .
- If $h(\mathbf{x}^{(l+1)}) < h[\hat{\mathbf{x}}^\circ(l)]$, set $\hat{\mathbf{x}}^\circ(l+1) \leftarrow \mathbf{x}^{(l+1)}$. Otherwise, set $\hat{\mathbf{x}}^\circ(l+1) \leftarrow \hat{\mathbf{x}}^\circ(l)$.
 - If some termination criterion is satisfied or a prefixed maximum number L of evaluations has been reached, then stop. Otherwise, set $l \leftarrow l + 1$ and go to (*). ▷

Clearly, both the batch approach and the recursive Procedure 5.1 are the simplest search techniques that can be conceived. In [48, p. 38], Procedure 5.1 is called “blind search” because it does not exploit the information that may have been obtained up to the current iteration. Note that the procedure can be used, at least in principle, for every function defined on continuous or discrete sets of vectors \mathbf{x} . One has simply to use continuous or discrete sampling distributions, respectively. Likewise, if \mathbf{x} takes both continuous and discrete values, then an appropriate mix of continuous and discrete sampling distributions can be exploited.

Note also that the recursive procedure is “open” in the sense that it can go on step by step until the user is satisfied with the current estimate $\hat{\mathbf{x}}^\circ(l)$. Of course, this is allowed unless a bound L is imposed on the number of evaluations. Theorem 2.1 stated in [48, p. 40] is very important in this case (see also [48, p. 40, 41] for its proof and examples). It provides conditions for the convergence $\hat{\mathbf{x}}^\circ(l) \rightarrow \mathbf{x}^\circ$ as $l \rightarrow \infty$. Informally, it states that such convergence is achieved if the cost function can take

the value $h(\mathbf{x}^\circ)$ or values near $h(\mathbf{x}^\circ)$ with nonzero probability. The probability is the one chosen for generating the samples $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots$.

Of course, it is also of interest to determine the rate of convergence of Procedure 5.1. This rate can be expressed in terms of the number of iterations of the procedure (i.e., the number of evaluations of the function h) necessary to reach a *satisfactory region* $N(\mathbf{x}^\circ)$ that represents some acceptable neighborhood of \mathbf{x}° . Following [48, p. 42, 43], assume that a maximum number L of iterations has been fixed a priori, and suppose that Procedure 5.1 terminates at the first iteration l^* at which $\hat{\mathbf{x}}^\circ(l)$ enters $N(\mathbf{x}^\circ)$ (if $l^* \leq L$), otherwise at the iteration L . If $l^* < L$, we set $\hat{\mathbf{x}}^\circ(L) = \dots = \hat{\mathbf{x}}^\circ(l^* + 1) = \hat{\mathbf{x}}^\circ(l^*)$.

Since the L samples are drawn independently, the probability of never entering $N(\mathbf{x}^\circ)$ is

$$\prod_{l=1}^L [1 - P\{\mathbf{x}^{(l)} \in N(\mathbf{x}^\circ)\}].$$

Consequently, the probability that an iterate $\mathbf{x}^{(l)}$, $l = 1, \dots, L$, enters $N(\mathbf{x}^\circ)$ within L iterations is given by

$$P\{\hat{\mathbf{x}}^\circ(L) \in N(\mathbf{x}^\circ)\} = 1 - \prod_{l=1}^L [1 - P\{\mathbf{x}^{(l)} \in N(\mathbf{x}^\circ)\}]. \quad (5.97)$$

Let us now assume that the probability distribution P is uniform on B . So, we let

$$P^* \triangleq P\{\mathbf{x}^{(l)} \in N(\mathbf{x}^\circ)\}, \quad l = 1, \dots, L. \quad (5.98)$$

Define also

$$P\{\hat{\mathbf{x}}^\circ(L) \in N(\mathbf{x}^\circ)\} \triangleq 1 - \varrho, \quad (5.99)$$

where $\varrho \in (0, 1)$ is a small positive scalar. Thus, substitution of (5.98) and (5.99) into (5.97) yields

$$1 - \varrho = 1 - (1 - P^*)^L,$$

hence

$$L = \left\lceil \frac{\ln \varrho}{\ln(1 - P^*)} \right\rceil \quad (5.100)$$

gives the minimal number of iterations of Procedure 5.1 guaranteeing that the estimate $\hat{\mathbf{x}}^\circ(L)$ enters $N(\mathbf{x}^\circ)$ with probability at least $1 - \varrho$.

Unfortunately, the expression (5.100) shows that L may grow very rapidly as the dimension d of \mathbf{x} gets large, thus making Procedure 5.1 inefficient. This can be seen by letting $B \triangleq [0, 1]^d$, $\mathbf{x}^\circ \triangleq \text{col}(x_1^\circ, \dots, x_d^\circ)$, $0 < \varepsilon < \frac{1}{2}$, and $\varepsilon < x_i^\circ < 1 - \varepsilon$, $i = 1, \dots, d$. Let the satisfactory region be given by the d -fold Cartesian product

$$N(\mathbf{x}^\circ) = [x_1^\circ - \varepsilon, x_1^\circ + \varepsilon] \times \dots \times [x_d^\circ - \varepsilon, x_d^\circ + \varepsilon].$$

Table 5.1 Minimal number of iterations L ensuring that an iterate of Procedure 5.1 enters the satisfaction region $N(\mathbf{x}^\circ)$ with probability 0.9 (i.e., $\rho = 0.1$). The number L is determined for $B = [0, 1]^d$, $\varepsilon = 0.04$ (i.e., $P^* = 0.08^d$) and some values of the dimension d

d	1	2	5	10
L	28	359	$7.5 \cdot 10^5$	$2.1 \cdot 10^{11}$

Since P is uniform on B , P^* is the volume of $N(\mathbf{x}^\circ)$, that is, $P^* = (2\varepsilon)^d$, and

$$L = \left\lceil \frac{\ln \rho}{\ln[1 - (2\varepsilon)^d]} \right\rceil.$$

Then, the decay of P^* as d increases implies (for a fixed probability $1 - \varrho$) an explosive growth of the minimal number L of cost evaluations (see [48]). The example reported in Table 5.1 illustrates this statement.

More sophisticated recursive algorithms were proposed, in which the random sampling is made dependent on the position of the estimate that has achieved the lowest value of the cost function h . In this way, the search is guided by the information previously obtained on the shape of the cost function in the environment surrounding the current estimate. The methods based on this concept are called *localized algorithms*. Such algorithms do not necessarily always perform better than the simpler blind search Procedure 5.1, as this depends on the specific problem dealt with. In practice, however, the localized algorithms usually have greater practical efficiency. Two of them are described in [48, Sect. 2.2.2], where examples are reported for a comparison with Procedure 5.1. It is worth noting that the convergence conditions for localized algorithms are in general more restrictive than those for Procedure 5.1.

5.4.2 Quasi-Monte Carlo Search

Also in the application of quasi-MC methods to the solution of Problem (5.96), both the *batch approach* and the *recursive approach* described in Sect. 5.4.1 can be applied. An advantage of quasi-MC methods is that low-discrepancy sequences cover the sample space more “uniformly” than the samples generated by an MC method, which tend to form clusters (see Fig. 4.13). As a result, such clusters leave possibly large “empty” areas on which they provide no information on the local behavior of the cost function h . If new points are added randomly (i.e., according to MC techniques), they do not necessarily cover such areas. In the quasi-MC framework instead, the successive points of a low-discrepancy sequence “take into account” the positions of their predecessors in the sequence. So, they are able to fill the gaps previously left.

This motivates, in general, the superiority of the quasi-MC methods over the MC ones in the search for minima. For example, in [28] we read: “It has been recognized

through theory and practice that a variety of uniformly distributed sequences provide more accurate results than purely random samples of points.”

The performance of quasi-MC methods for global optimization is analyzed, among others, in [36, Chap. 6] or, with a lighter treatment, in [28]. Let \mathbf{x}_L° be defined as in Problem (5.96), where the points $\mathbf{x}^{(l)}$ are now generated by a quasi-MC algorithm. Then, the following result holds, which is the deterministic counterpart of Theorem 5.4 (see [36, Chap. 6], [47]).

Theorem 5.5 *If h is continuous on B and if the sequence $\{\mathbf{x}^{(l)}\}_{l=1}^\infty$ is dense in B , then*

$$\lim_{L \rightarrow \infty} h(\mathbf{x}_L^\circ) = h^\circ.$$

□

In order to investigate rates of convergence, we suppose that the set B in Problem (5.96) is $[0, 1]^d$. We also assume that the cost function h is Lipschitz-continuous, with Lipschitz constant $C > 0$, that is,

$$|h(\mathbf{x}) - h(\mathbf{y})| \leq C|\mathbf{x} - \mathbf{y}| \quad (5.101)$$

for every $\mathbf{x}, \mathbf{y} \in [0, 1]^d$. Now, we consider the sequence $\{\mathbf{x}^L\} = \{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(L)}\}$. The following concept is needed.

Definition 5.1 Let $\{\mathbf{x}^L\}$ be a sequence belonging to $[0, 1]^d$. The “dispersion” of the sequence $\{\mathbf{x}^L\}$ (or of the sample set X_L) is defined as

$$d_L(\{\mathbf{x}^L\}) \triangleq \sup_{\mathbf{x} \in [0, 1]^d} \min_{l=1, \dots, L} |\mathbf{x} - \mathbf{x}^{(l)}|. \quad (5.102)$$

△

In (5.102) we used the Euclidean norm, as in most applications. Another possibility consists in using the maximum norm

$$|\mathbf{x} - \mathbf{x}^{(l)}|_\infty = \max_{1 \leq i \leq d} |x_i - x_i^{(l)}|.$$

From (5.101) and (5.102), one gets

$$|h(\mathbf{x}_L^\circ) - h(\mathbf{x}^\circ)| \leq C d_L(\{\mathbf{x}^L\}), \quad (5.103)$$

where

$$\mathbf{x}_L^\circ \triangleq \arg \min_{\mathbf{x}^{(l)} \in \{\mathbf{x}^L\}} h(\mathbf{x}^{(l)}).$$

Therefore, the smaller the dispersion of a sequence $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(L)}\}$, the smaller the upper bound on the error in (5.103). It is worth noting that the dispersion $d_L(\{\mathbf{x}^L\})$

is related to the star discrepancy $D_L^*(X_L)$, in the sense that it can be proved that low-discrepancy sequences have also small dispersion. The vice versa is not generally true (see, [36, Theorem 6.6]). Even better upper bounds on $|h(\mathbf{x}_L^\circ) - h(\mathbf{x}^\circ)|$ can be obtained under the assumption that the function h “strongly” depends only on $s < d$ components of the vector \mathbf{x} , whereas the dependence on the other components is “weak”. Then, in many practical applications, d can be substituted by an “effective dimension s ,” which may happen to be much smaller than d (see, e.g., [28, Sect. 2]).

Following the discussion contained in [28], we report some advantages of quasi-MC methods (at least in their simplest forms described in this section) with respect to other global optimization methods. In their most general forms, quasi-MC methods (as well as MC ones) do not require assumptions about the structure of Problem (5.95). In particular, as already mentioned, they can be used for any class of cost functions (for instance, non-differentiable ones). Moreover, inequality constraints can be explicitly taken into account, and the feasible region can be non-convex and even disconnected. Instead, the equality constraints cannot be taken into account and the optimization problem has to be transformed into an unconstrained one.

The quasi-MC search method described so far may be called *crude search*. Indeed, its rate of convergence is in general very slow and, consequently, its usefulness is in practice quite limited. All this is essentially due to the fact that most of the time is spent to compute the cost function in points that are far from the minimum point \mathbf{x}° . Note that both MC and quasi-MC methods belong to the class of the so-called *nonadaptive* algorithms, in which the numerical process depends only on the current iteration step and not on the previously computed ones. Note that adaptivity can improve the efficiency of a crude search. For example, adaptivity can be performed as follows. Fewer points $\mathbf{x}^{(l)}$ are considered in the regions where the cost function h varies slowly, whereas h is evaluated in a larger number of points in the regions where the cost function oscillates strongly. An initial crude search can provide information on the oscillations of h .

Other methods that mitigate the previously mentioned loss of time in points far from the minimum point \mathbf{x}° are the so-called *multistage* algorithms. In their simplest form, multistage methods are the two-phase methods mentioned in Sect. 5.3.9, which work this way. There are two phases. In the first one (called *global phase*), MC or quasi-MC techniques are used to solve the optimization problem up to a given tolerance. Then, in the second phase (named *local phase*), a local minimization search is performed, starting from the most promising solutions obtained in the first phase. Depending on the nature of the cost function, such a local minimization search can be performed by classical NLP methods or by further sampling in small neighborhoods of the most promising solutions when NLP algorithms are difficult to apply or cannot be applied at all (e.g., in the case of gradient-based methods with non-differentiable cost functions). Alternatively, a small number of nearby points is generated and the local minimization is applied to all such points. The point with the lowest value of h is taken as global minimum point.

A drawback of this multistarting technique is that the descent algorithms may lead to the same local minima while starting from different initial sample points. In this case, an additional phase may be useful. Only the sample points whose cost values

are sufficiently small are chosen, and those belonging to the region of attraction of the same local minimizer are grouped in the same *cluster*. Hence, the local phase is performed only once for each cluster. If the clusters are chosen properly, then the clustering phase reduces the number of local searches. Moreover, this technique does not affect the search for the global minimum. As regards the definition of clusters, all sample points within a *critical distance* are assigned to the same cluster. Various critical distances have been described in [28]. In general, as in other problems where both MC and quasi-MC techniques are applied, significant improvements in the efficiency of the algorithms are obtained by simply replacing random samples with low-discrepancy sequences. A few algorithms were developed on the basis of the clustering concept. The interested reader may refer, for instance, to [28] and the works cited therein. In that paper, several numerical examples of locating global minima in problems of increasing complexity were also presented.

References

1. Bazaraa MS, Sherali HD, Shetty CM (2006) Nonlinear programming: theory and algorithms, 3rd edn. Wiley
2. Benveniste A, Métivier M, Priouret P (1990) Adaptive algorithms and stochastic approximation. Springer, Berlin
3. Bertsekas DP (1999) Nonlinear programming, 2nd edn. Athena Scientific
4. Bertsekas DP, Tsitsiklis JN (1996) Neuro-dynamic programming. Athena Scientific
5. Blum C, Roli A (2003) Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput Surv 35:268–308
6. Boender CGE, Romeijn HE (1995) Stochastic methods. In: Horst R, Pardalos PM (eds) Handbook of global optimization. Kluwer, pp 829–869
7. Conn AR, Scheinberg K, Vicente LN (2009) Introduction to derivative-free optimization. MPS-SIAM Series on Optimization
8. Dick J, Pillichshammer F (2010) Digital nets and sequences: discrepancy theory and quasi-Monte Carlo integration. Cambridge University Press
9. Ermoliev Yu, Wets RJ-B (eds) (1980) Numerical techniques for stochastic optimization. Springer, Berlin
10. Fang H, Gong G, Qian M (1997) Annealing of iterative stochastic schemes. SIAM J Control Optim 35:1886–1907
11. Fishman G (1996) Monte Carlo: concepts, algorithms, and applications. Springer, New York
12. Fleming W (1977) Functions of several variables. Springer, New York
13. Floudas CA (2013) Deterministic global optimization: theory, methods and applications. Non-convex Optimization and its Applications, vol 37. Springer-Science + Business Media, Dordrecht
14. Floudas CA, Gounaris CE (2009) A review of recent advances in global optimization. J Global Optim 45:3–38
15. Gelfand S, Mitter SK (1991) Simulated annealing type algorithms for multivariate optimization. Algorithmica 6:419–436
16. Gelfand S, Mitter SK (1993) Metropolis-type annealing algorithms for global optimization in \mathbb{R}^d . SIAM J Control Optim 31:111–131
17. Geman S, Hwang CR (1986) Diffusions for global optimization. SIAM J Control Optim 24:1031–1043
18. Giaquinta M, Modica G (2009) Mathematical analysis: an introduction to functions of several variables. Birkhäuser

19. Glover F, Kochenberger GA (2003) Handbook of metaheuristics, vol 57. International Series in Operations Research & Management Science. Kluwer
20. Haftka RT, Villanueva D, Chaudhuri A (2016) Parallel surrogate-assisted global optimization with expensive functions—a survey. *Struct Multidiscip Optim* 54:3–13
21. Hartinger J, Kainhofer R (2006) Non-uniform low-discrepancy sequence generation and integration of singular integrands. In: Niederreiter H, Talay D (eds) Monte Carlo and Quasi-Monte Carlo methods 2004. Springer, Berlin, pp 163–179
22. Hinrichs A, Novak E, Ullrich M, Woźniakowski H (2014) The curse of dimensionality for numerical integration of smooth functions II. *J Complex* 30:117–143
23. Horst R, Pardalos PM (eds) (1995) Handbook of global optimization. Kluwer
24. Horst R, Tuy H (1996) Global optimization: deterministic approaches, 3rd edn. Springer, Berlin
25. Horst R, Tuy H (2013) Global optimization: deterministic approaches, 3rd edn. Springer-Science + Business Media
26. Kainen PC (1997) Utilizing geometric anomalies of high dimension: when complexity makes computation easier. In: Warwick K, Karni M (eds) Compute-intensive methods in control and signal processing. The curse of dimensionality, Birkhäuser, pp 283–294
27. Kolda TG, Lewis RM, Torczon V (2003) Optimization by direct search: new perspectives on some classical and modern methods. *SIAM Rev* 45:385–482
28. Kucherenko S (2006) Applications of Quasi Monte Carlo methods in global optimization. In: Liberti L, Maculan C (eds) Global optimization: from theory to implementation, vol 84. Springer, Boston, pp 111–133
29. Kushner HJ (1987) Asymptotic global behavior for stochastic approximation and diffusions with slowly decreasing noise effects: global minimization via Monte Carlo. *SIAM J Appl Math* 47:169–185
30. Kushner HJ, Yin GG (2003) Stochastic approximation and recursive algorithms, 2nd edn. Springer, New York
31. Liu JS (2001) Monte Carlo strategies in scientific computing. Springer, New York
32. Locatelli M, Schoen F (2013) Global optimization: theory, algorithms, and applications. SIAM, Mos-SIAM Series on Optimization
33. Morokoff WJ, Caflisch RE (1994) Quasi-Monte Carlo integration. *J Comput Phys* 122:218–230
34. Morokoff WJ, Caflisch RE (1994) Quasi-random sequences and their discrepancies. *SIAM J Sci Comput* 15:1251–1279
35. Moskowitz B, Caflisch RE (1996) Smoothness and dimension reduction in quasi-Monte Carlo methods. *Math Comput Model* 23:37–54
36. Niederreiter H (1992) Random number generation and Quasi-Monte Carlo methods. SIAM
37. Owen A (1995) Randomly permuted (t, m, s) -nets and (t, s) sequences. In: Niederreiter H, Shiu PJ-S (eds) Monte Carlo and Quasi-Monte Carlo methods in scientific computing (Lecture Notes in Statistics, vol 106). Springer, New York, pp 299–317
38. Owen A (1997) Scrambled net variance for integrals of smooth functions. *Ann Stat* 25:1541–1562
39. Pardalos PM, Romeijn HE (eds) (2002) Handbook of global optimization , vol 2. Kluwer
40. Pintér JD (2002) Global optimization: software, test problems, and applications. In: Romeijn HE, Pardalos PM (eds) Handbook of global optimization, vol 2 (Chap 15). Kluwer, pp 515–569
41. Rardin RL (1998) Optimization in operations research. Prentice Hall
42. Robbins H, Monro S (1951) A stochastic approximation method. *Ann Math Stat* 22:400–407
43. Robert CP, Casella G (2004) Monte Carlo statistical methods, 2nd edn. Springer, New York
44. Schlier C (2004) Discrepancy behaviour in the non-asymptotic regime. *Appl Numer Math* 50:227–238
45. Schoen F (1991) Stochastic techniques for global optimization: a survey of recent advances. *J Global Optim* 1:207–228
46. Schoen F (2002) Two-phase methods for global optimization. In: Pardalos PM, Romeijn HE (eds) Handbook of global optimization, vol 2. Kluwer, pp 151–177
47. Sobol' IM (1992) An efficient approach to multicriteria optimum design problems. *Surv Math Ind* 1:259–281

48. Spall JC (2003) Introduction to stochastic search and optimization: estimation, simulation and control. Wiley
49. Törn A, Žilinskas A (1989) Global optimization. Springer, Berlin
50. Tsyplkin YZ (1971) Adaptation and learning in automatic systems. Academic Press
51. Tuy H (2016) Convex analysis and global optimization. Springer
52. Vavasis SA (1995) Complexity issues in global optimization: a survey. In: Encyclopedia of optimization, pp 27–41. Kluwer,
53. Wang X (2000) Improving the rejection sampling method in quasi-Monte Carlo methods. *J Comput Appl Math* 114:231–246
54. Yin G (1999) Rates of convergence for a class of global stochastic optimization algorithms. *SIAM J Optim* 10:99–120
55. Zabinsky ZB (2003) Stochastic adaptive search for global optimization. Kluwer
56. Zhigljavsky A, Žilinskas (2008) A Stochastic global optimization. Springer, New York
57. Zlochin M, Birattari M, Meuleau N, Dorigo M (2004) Model-based search for combinatorial optimization: a critical survey. *Ann Oper Res* 131:373–395

Chapter 6

Deterministic Optimal Control over a Finite Horizon



In this chapter, we address discrete-time deterministic problems where optimal controls have to be generated at a finite number T of time instants or decision stages. No random variables influence either the dynamic system or the cost function. The optimal control law consists of a sequence of optimal control vectors that can be computed offline and applied online in an open-loop form. Indeed, the state trajectory can be predicted a priori, so that, at least in principle, there is no necessity of observing the state vector.

The deterministic problem is defined as Problem C1 and stated in Sect. 6.1. We formulate it for two main reasons. First, it is of intrinsic practical importance. Second, by focusing on Problem C1 we shall have a rather simple framework where we can introduce and investigate one of the two important instruments that we shall extensively use to solve the more complex T -stage stochastic optimal control Problems C2 and C3 addressed in Chaps. 7 and 8, respectively. These instruments are given by dynamic programming (DP) and the extended Ritz method (ERIM). We describe the basic concepts for understanding DP in Sect. 6.2, where we present a simple but constructive example of a T -stage decision problem. In Sect. 6.3, we extend these concepts to solve Problem C1. As this problem is stated in quite a general form (e.g., we do not assume the classical *linear-quadratic* (LQ) hypotheses to be verified), its optimal solution cannot be found in an analytical form. Therefore, in Sect. 6.4, we resort to two main approximations consisting of the discretization of the state space into suitable grids at each decision stage, and in approximating the optimal cost-to-go and control functions by fixed-structure parametrized (FSP) functions. The resulting numerical DP procedure is called *approximate dynamic programming* (ADP).

The generation of the optimal controls in the forward phase of DP can be performed in various ways. We address this issue in some detail in Sect. 6.5. The use of FSP functions and the related generation of samples for the construction of grids in the state space require the computation of generalization, approximation, and estimation

errors. These concepts and tools, that were presented in Chap. 4, are applied in Sect. 6.6. In particular, we establish error bounds, with special reference to the use of quasi-Monte Carlo (quasi-MC) techniques.

Finally, in Sect. 6.7, we point out that Problem C1 can also be viewed as a nonlinear programming (NLP) problem. Therefore, once the gradient of the cost function is computed, it can be used to solve Problem C1 via a two-point-boundary-value problem and by applying gradient-based NLP descent techniques. The same section also examines the case in which the terminal time of the process control has to be determined.

6.1 Statement of Problem C1

We consider the following dynamic system (generally nonlinear):

$$\mathbf{x}_{t+1} = \mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t), \quad t = 0, 1, \dots, T - 1, \quad (6.1)$$

where $\mathbf{x}_0 = \hat{\mathbf{x}}$ is a given initial state. The state vector \mathbf{x}_t may be constrained to belong to a nonempty set X_t , that is,

$$\mathbf{x}_t \in X_t \subseteq \mathbb{R}^d, \quad t = 1, \dots, T. \quad (6.2)$$

The control vector \mathbf{u}_t may be constrained to take values in a nonempty set $U_t(\mathbf{x}_t)$ that may depend on the decision stage t and on the current state \mathbf{x}_t , i.e.,

$$\mathbf{u}_t \in U_t(\mathbf{x}_t) \subseteq \mathbb{R}^m, \quad t = 0, 1, \dots, T - 1. \quad (6.3)$$

We assume that the cost function has an additive form and is given by

$$J = \sum_{t=0}^{T-1} h_t(\mathbf{x}_t, \mathbf{u}_t) + h_T(\mathbf{x}_T), \quad (6.4)$$

where $h_t(\mathbf{x}_t, \mathbf{u}_t)$ are transition costs to be paid at stage t and $h_T(\mathbf{x}_T)$ is a terminal cost incurred at the end of the process. In this and in the next chapters, unless otherwise stated (e.g., in the example presented in Sect. 6.2), all the variables – i.e., state, control, disturbances (in the case of the problems considered in the next chapters), etc. – will take on values in continuous sets. Then, we can state the deterministic T -stage optimal control problem as follows.

Problem C1. *Find the sequence of optimal control vectors $\mathbf{u}_0^\circ, \mathbf{u}_1^\circ, \dots, \mathbf{u}_{T-1}^\circ$ that minimizes the cost J , subject to Constraints (6.1), (6.2), and (6.3).* \triangleleft

Problem C1 is the deterministic version of Problem 1.4, as the random sequence of disturbances $\{\xi_0, \xi_1, \dots, \xi_{T-1}\}$ is absent. The feedback on the state vector is not necessary as the state trajectory $\{\mathbf{x}_1, \dots, \mathbf{x}_T\}$ can be computed a priori, once the

initial state \mathbf{x}_0 and the sequence of control vectors $\{\mathbf{u}_0, \mathbf{u}_1, \dots, \mathbf{u}_{T-1}\}$ have been fixed. Therefore, no advantage results from implementing closed-loop control functions $\mathbf{u}_t = \mu_t(\mathbf{x}_t)$ (see (1.23)) (provided that one is really sure that no disturbance will affect the dynamic system). It is worth noting that (in general) Problem C1 is a linear or nonlinear programming problem (i.e., a finite-dimensional optimization (FDO) problem) as the unknown is given by the finite-dimensional vector $\mathbf{u} \triangleq \text{col}(\mathbf{u}_0, \mathbf{u}_1, \dots, \mathbf{u}_{T-1})$, while the optimal control problem stated in Sect. 1.5.3 is an infinite-dimensional optimization (IDO) or functional optimization problem.

It is important to note that in the statement of Problem C1 the duration of the decision problem does not appear at all as a variable to be determined. This may be reasonable in some cases but not in others. The presence or not of the duration depends on the origins of Problem C1 and, in general, on the T -stage decision problems considered. In Sect. 1.1.1, we said that, in economic and management problems, T is usually a fixed integer and decisions are taken periodically. In this case, the duration of the decision process is of no interest. However, if Problem C1 has its origin in a framework described by physical laws, the time may play an important role whenever one has to consider Problem C1 as the approximation of a continuous-time optimal control problem. This occurs when the process duration (i.e., the final time t_f if the initial time is 0) has to be optimized together with the sequence of controls. Think, for instance, of minimum-time problems.

To fix our ideas, let us consider a simplified version of Problem 1.2. Terminal region (1.13) and Constraints (1.15) are omitted. However, the final time t_f is free and has to be determined together with the control law, so as to minimize the cost J . Problem 1.2 is then modified as follows.

Problem 6.1 *Find the optimal control law $\mathbf{u}^\circ(t)$, $t_0 \leq t \leq t_f$ and the optimal final time t_f° that minimize the cost*

$$J = \int_0^{t_f} h_c[\mathbf{x}(t), \mathbf{u}(t), t] dt + h_f[\mathbf{x}(t_f), t_f], \quad (6.5)$$

and verify State Equation (1.12). \triangleleft

In Sect. 6.7.2, we address a discrete-time approximation of Problem 6.1. We consider this approximation with a *fixed* final time t_f as a particular case of Problem C1. Consistently with this, we call the discrete-time approximation of Problem 6.1 with *free* final time t_f *Problem C1 t_f* .

6.2 A Constructive Example for Understanding the Basic Concepts of Dynamic Programming

Let us give an example that is particularly simple, but still meets all the requirements for understanding the basic mechanisms of the DP procedure for solving quite general T -stage decision problems. We search for the minimum-cost route in the oriented graph of Fig. 6.1.

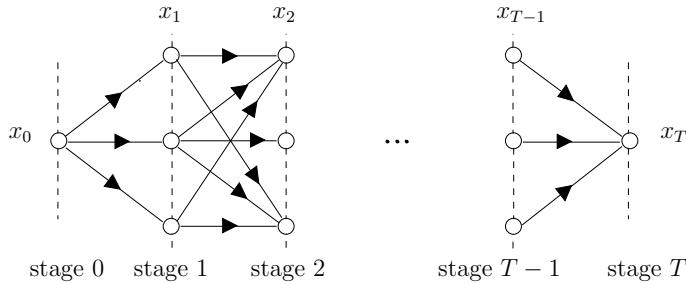
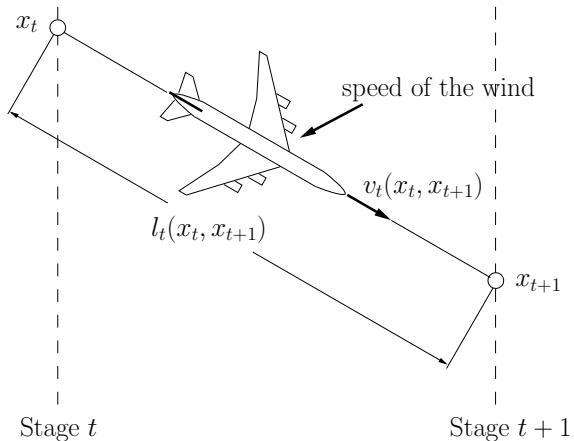


Fig. 6.1 An example of oriented graph in DP

To fix our ideas, let us assume that the routes in the graph correspond to the possible routes of an airplane starting from point x_0 and arriving at point x_T . Let us also assume that we aim at determining the minimum-time trajectory (or trajectories) of the airplane in the graph. For brevity, in the following paragraphs, the term “minimum-time” will frequently be replaced by the term “optimal”. Then, the graph is plotted by dividing the two-dimensional surface between x_0 and x_T into a rectangular grid: the vertical lines of the grid, apart from the ones associated with the stages $t = 0, T$, correspond to the decision stages $t = 0, 1, \dots, T - 1$, and the graph nodes located on each vertical line correspond to q possible discretized route points. Moreover, let us denote each point x_t of the t th stage by an integer $x_t \in \{1, 2, \dots, q\}$ for $1 \leq t \leq T - 1$. Then, at each node x_t of the t th stage, a decision must be made on the next node to be reached, i.e., x_{t+1} . Let us assume that from each node x_t it is possible to reach any next node x_{t+1} . This means neglecting the dynamics of the airplane. Obviously, at the stage $T - 1$, no decision has to be made. It is worth noting that considering the graph on a two-dimensional surface means neglecting the flying height of the airplane. This is reasonable, if the takeoff and landing times are negligible, as compared with the total flight time, and if the airplane’s route is held at a fixed constant flying height.

Of course, if the meteorological situation between the nodes x_0 and x_T did not affect the airplane’s speed, the minimum-time trajectory would coincide with the orthodromic arc linking x_0 to x_T . However, the speed is affected by the meteorological conditions, particularly by the presence of wind. Let us assume that the characteristics of the wind (speed and direction) are deterministic, time-invariant, and known (as a result of the weather forecast) at each point covered by the flight. Moreover, let us suppose that the pilot has fixed a certain constant engine thrust (corresponding, in the absence of wind, to a given cruising speed), and that the autopilot keeps the airplane exactly on the orthodromic arc (x_t, x_{t+1}) joining x_t to x_{t+1} . Given the mean characteristics of the wind on such an arc, it is then possible to calculate the airplane’s speed, $v_t(x_t, x_{t+1})$ on the arc itself (in general, this speed differs from the cruising speed). Then, if one denotes the length of the arc (x_t, x_{t+1}) by $l_t(x_t, x_{t+1})$ (see Fig. 6.2), the time $h_t(x_t, x_{t+1})$ required to cover the arc is given by

Fig. 6.2 The airplane flies the arc joining the node x_t to the node x_{t+1}



$$h_t(x_t, x_{t+1}) = \frac{l_t(x_t, x_{t+1})}{v_t(x_t, x_{t+1})}.$$

The problem lies in determining the route $x_0, x_1^\circ, \dots, x_{T-1}^\circ, x_T$ yielding the minimum total time. In other words, one has to solve the problem

$$\min_{x_1, x_2, \dots, x_{T-1}} \sum_{t=0}^{T-1} h_t(x_t, x_{t+1}). \quad (6.6)$$

A quite similar problem can be found in [17].

Let J° be the minimum total time obtained by solving Minimization (6.6). It is important to point out that the optimal routing problem for the airplane is nothing but a special case of Problem C1. Indeed, if we denote by u_t the decision to be made at node x_t (i.e., the next node x_{t+1}), we can write

$$x_{t+1} = u_t, \quad t = 0, 1, \dots, T - 1. \quad (6.7)$$

Then, x_t turns out to be a scalar discrete state variable, u_t a control variable, and (6.7) a state equation. (Note that, in the optimal routing problem for the airplane, the controls u_t are generated at the “geometrical” decision stage t , whereas in Problem C1 they are generated at the “temporal” stage t ; this difference, however, has no conceptual significance.) The cost to be minimized in (6.6) can be rewritten in the form of Cost (6.4) (clearly, $h_T(x_T) = 0$), that is,

$$J = \sum_{t=0}^{T-1} h_t(x_t, u_t).$$

As we shall see, all the conclusions (based on DP) that we shall draw from the optimal routing problem apply directly to Problem C1. In this sense, the introductory example dealt with has a constructive meaning.

Before presenting the DP approach, we note that an optimal solution of Problem (6.6) might be found by choosing one trajectory at a time among all possible trajectories, comparing the time required by this trajectory with the shortest time previously calculated, and hence determining the trajectory (or the trajectories) that achieves the minimum time.

The obvious drawback of this procedure, which we might call “brute-force” enumeration procedure, concerns the number of trajectories to be considered: in general, this number is too large. To fix our ideas, let us assume that 20 decisional stages ($T = 21$) are given, and that, at each stage, apart from the stages $t = 0$ and $t = T = 21$, there are 10 possible route nodes ($q = 10$). Therefore, the number of trajectories to be examined turns out to be equal to 10^{20} . Let us also suppose that the computation time taken by an elementary calculation (i.e., determination of the time required by a given trajectory and comparison of this time with the shortest one previously determined) is equal to $1 \mu s$. It follows that the time required to consider all the trajectories is $10^{20} \cdot 10^{-6} s \simeq 3.17 \cdot 10^6$ years!

A method that allows us to avoid so expensive computational approach involves the following procedure. First of all, we define the “minimum time-to-go functions” $J_t^\circ(x_t)$ that are given by the minimum times required by the airplane for reaching the arrival node x_T starting from a generic node $x_t \in \{1, 2, \dots, 10\}$ for $1 \leq t \leq T - 1$, and x_0 for $t = 0$. We have

$$J_{T-1}^\circ(x_{T-1}) \triangleq h_{T-1}(x_{T-1}, x_T), \quad (6.8)$$

$$J_t^\circ(x_t) \triangleq \min_{x_{t+1}, \dots, x_{T-1}} \sum_{k=t}^{T-1} h_k(x_k, x_{k+1}), \quad t = 0, 1, \dots, T - 2.$$

Let us now consider the stage $t = T - 2$ and, for each x_{T-2} , let us calculate the minimum times

$$J_{T-2}^\circ(x_{T-2}) = \min_{x_{T-1}} [h_{T-2}(x_{T-2}, x_{T-1}) + J_{T-1}^\circ(x_{T-1})]. \quad (6.9)$$

We postpone to the end of this section the discussion of the equivalence of the expressions of $J_{T-2}^\circ(x_{T-2})$ obtained, respectively, by (6.8) and (6.9). Note that the solution of Problem (6.9) also provides an optimal control function of the type $x_{T-1}^\circ = u_{T-2}^\circ = \mu_{T-2}^\circ(x_{T-2})$, and hence the next node to be reached starting from each node x_{T-2} .

The procedure goes on in the same way, thus calculating “backwards” the minimum times-to-go

$$\begin{aligned} J_{T-1}^{\circ}(x_{T-1}) &= h_{T-1}(x_{T-1}, x_T), \\ J_t^{\circ}(x_t) &= \min_{x_{t+1}} [h_t(x_t, x_{t+1}) + J_{t+1}^{\circ}(x_{t+1})], \quad t = 0, 1, \dots, T-2. \end{aligned} \tag{6.10}$$

The optimal control functions are derived in the same way as for the stage $T - 2$. They take on the form

$$x_{t+1}^{\circ} = u_t^{\circ} = \mu_t^{\circ}(x_t), \quad t = 0, 1, \dots, T-1. \tag{6.11}$$

Recursive Equation (6.10) describes the DP computational procedure. Clearly, at the stage $t = 0$, we obtain the duration of the airplane's minimum-time trajectory, that is,

$$J^{\circ} = J_0^{\circ}(x_0).$$

Once the functions $\mu_t^{\circ}(x_t)$ and the times $J_t^{\circ}(x_t)$ have been computed for any state x_t , the "backward" phase of the DP procedure is concluded, and we can derive the minimum-time trajectory from the "forward" phase of DP as follows. At the starting node x_0 , we determine the first node to be reached by using the optimal control function $\mu_0^{\circ}(x_0)$, that is,

$$x_1^{\circ} = \mu_0^{\circ}(x_0).$$

At the state x_1° , we compute

$$x_2^{\circ} = \mu_1^{\circ}(x_1^{\circ}),$$

and the same mechanism continues until the minimum-time trajectory $x_0, x_1^{\circ}, \dots, x_{T-1}^{\circ}, x_T$ is obtained. It is important to note that after the backward phase of the DP procedure has been completed, a *closed-loop control law* is derived. Actually, as shown in Fig. 6.3, the optimal control (i.e., the next node to be reached) is specified as a function of both the stage and the state, not just as a function of the stage, as would occur for an *open-loop control law*.

The importance of a closed-loop solution is evident in the case of deviations from the original minimum-time trajectory. In our example, these deviations might take place if, at some instant of the flight, unforeseen wind conditions moved the airplane to a node not belonging to the minimum-time trajectory (in this case, the assumptions on the deterministic environment that led us to state the minimum-time trajectory problem as Problem C1 would be disproved). The availability of a closed-loop optimal control law enables us to generate a control sequence that is the really optimal one for the remaining stages (of course, the discretization of the vertical segments corresponding to the decisional stages in the grid of Fig. 6.1 should be sufficiently fine so that we may say that the wind has moved the airplane to another node $\tilde{x}_t \neq x_t^{\circ}$, which is not on the minimum-time trajectory but is practically coincident with a node of the grid). On the other hand, if the control law was open-loop, the remainder of the trajectory (i.e., $\tilde{x}_t, x_{t+1}^{\circ}, \dots, x_{T-1}^{\circ}, x_T$) would be suboptimal, unless further computations

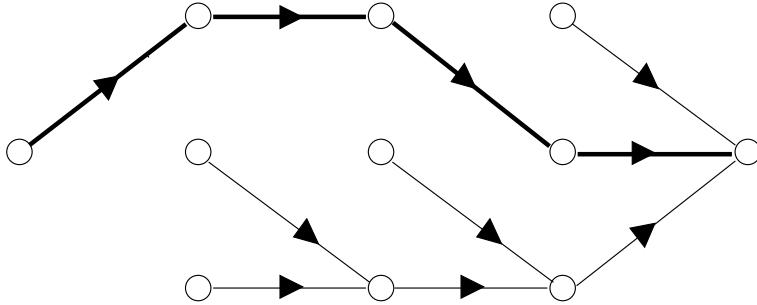


Fig. 6.3 The minimum-time control law in closed-loop form. The arcs indicate the optimal decisions at each node; the thicker arcs indicate the open-loop optimal decisions and the minimum-time trajectory

are performed online to determine a new partial minimum-time trajectory starting from \tilde{x}_t , (“reoptimization”).

The DP recursive equation (6.10) is the operational translation of Bellman’s *principle of optimality* (see [1–3]), which, with reference to our example, can be stated as follows: Whatever the sequence of decisions that led from x_0 to $\hat{x}_t \in \{1, \dots, q\}$, $t = 1, \dots, T-1$ (\hat{x}_t is not required to coincide with a node x_t on the optimal trajectory), a necessary condition for the trajectory $\hat{x}_t, \hat{x}_{t+1}, \dots, \hat{x}_{T-1}, x_T$ to be a portion of the optimal trajectory is that the said trajectory should be in itself an optimal solution of the following partial optimization problem:

$$J_t^\circ(x_t) = \min_{x_{t+1}, \dots, x_{T-1}} \sum_{k=t}^{T-1} h_k(x_k, x_{k+1}). \quad (6.12)$$

The above definition can be expressed in more concise terms as follows: “The second part of an optimal trajectory must be in itself an optimal trajectory.”

The principle of optimality can also be justified as follows. Let us assume that the trajectory $x_0, x_1^\circ, \dots, x_t^\circ, x_{t+1}^\circ, \dots, x_{T-1}^\circ, x_T$ is optimal (suppose, for simplicity, that there is only one optimal solution). Let us also assume that Minimization (6.12) provides a different trajectory portion in the final part (i.e., $x_t^\circ, x_{t+1}^*, \dots, x_{T-1}^*, x_T$) characterized by a route time from x_t° to x_T that is shorter than the corresponding partial route time of the optimal trajectory. Then, the new trajectory $x_0, x_1^\circ, \dots, x_t^\circ, x_{t+1}^*, \dots, x_{T-1}^*, x_T$ is better than the optimal one and this is a contradiction.

In the present example, one might apply the principle of optimality in order to obtain a “forward” resolving procedure (i.e., considering the partial minimum time necessary to reach x_t starting from x_0) instead of the “backward” resolving method previously described. Concerning the applicability of the “forward” and “backward” methods, see, for instance, [13]. For a better understanding of the seemingly simple concept of the principle of optimality, we refer to a variety of examples reported, e.g., in [14].

Considering how things have been presented, one may feel that Equations (6.10) have been derived on the basis of a somehow “heuristic” principle just based on the principle of optimality. So, one may prefer to obtain Equation (6.10) by some “formal” demonstration (indeed, in more complex optimal decision problems, some care must be taken in deriving the DP procedure directly from the principle of optimality). To this end, let us consider Minimum Time-to-Go Functions (6.8) at the starting point x_0 and split the summation as follows:

$$J^\circ = J_0^\circ(x_0) = \min_{x_1, \dots, x_{T-1}} \left[h_0(x_0, x_1) + \sum_{k=1}^{T-1} h_k(x_k, x_{k+1}) \right]. \quad (6.13)$$

Next, the minimization in (6.13) is also split into two parts:

$$J_0^\circ(x_0) = \min_{x_1} \min_{x_2, \dots, x_{T-1}} \left[h_0(x_0, x_1) + \sum_{k=1}^{T-1} h_k(x_k, x_{k+1}) \right].$$

As $h_0(x_0, x_1)$ depends only on x_1 and not on x_2, \dots, x_{T-1} , we can write

$$J_0^\circ(x_0) = \min_{x_1} \left[h_0(x_0, x_1) + \min_{x_2, \dots, x_{T-1}} \sum_{k=1}^{T-1} h_k(x_k, x_{k+1}) \right].$$

Recalling definition (6.8) of $J_t^\circ(x_t)$ for $t = 1$, we have

$$J_0^\circ(x_0) = \min_{x_1} [h_0(x_0, x_1) + J_1^\circ(x_1)],$$

and hence we have obtained DP Equation (6.10) for $t = 0$. It is evident that, if we repeat the same reasoning for $J_1^\circ(x_1)$, then we obtain (6.10) for $t = 1$. By following the same procedure, we can “demonstrate” DP Equation (6.10) also for $t = 1, 2, \dots, T - 2$.

Let us now consider what are the advantages of applying DP to the solution of the minimum-time routing problem (in terms of computation time). The implementation of (6.10) involves two operations at any node x_t : (i) the addition of the time $h_t(x_t, x_{t+1})$ to the minimum time-to-go $J_{t+1}^\circ(x_{t+1})$ and (ii) the comparison of this sum with the shortest time-to-go previously obtained for the other nodes x_{t+1} .

Suppose that the two operations above take a total computation time equal to $1\mu s$. This assumption is surely pessimistic if compared with the time required by one step of the brute-force procedure, which consists of the sum of 20 numbers $h_t(x_t, x_{t+1})$ plus a comparison. Using DP, the elementary computations are performed for each of the 10 nodes x_t of every stage $t = 1, \dots, T - 2 = 19$ and for each of the 10 nodes x_{t+1} that can be reached starting from x_t . At the stage $t = 0$, only one starting point is to be considered, whereas at the stage $t = T - 1 = 20$, no computations are needed. The total computation time is then equal to $\simeq (19 \cdot 10 + 1 \cdot 1) \cdot 10 \cdot 1\mu s$.

$10^{-6}s = 1.91\text{ ms}$. Note that the computation time increases linearly with the number T of stages, and not exponentially as in the brute-force enumeration procedure.

The example described in this section might seem to be the “winning” algorithm for T -stage optimal control problems. Unfortunately, the clear advantage of applying this algorithm rests on the fact that a scalar state equation is used in the example. When the state dimension increases, hard computational requirements may prevent one from using DP practically. We shall discuss this point in the next sections.

6.3 Solution of Problem C1 by Dynamic Programming: The Exact Approach

The considerations related to the example presented in the previous section can be easily extended to the optimization of the control law for dynamic systems and cost functions not subject to random variables, that is, they can be extended to Problem C1. As we said, the minimum-time routing problem is a particular case of Problem C1. This problem may correspond to one of the many decision processes for which decisions are made at different time instants, e.g., at fixed dates in some management problem. Alternatively, Problem C1 may approximate a continuous-time optimal control problem by a discrete-time one. In the latter case, (6.1) approximates State Equation (1.12), and (6.4) approximates Integral Cost (1.14). If the control actions are generated by digital devices, then the decision stages correspond to the sampling instants.

In this section, we do not address the numerical issues connected with the solution of Problem C1 by DP. These numerical aspects essentially regard the sampling of the state space and will be considered in Sect. 6.4 and in those that follow it. As no discretization errors are involved, in this section, we deal with the “exact” application of DP, whereas in the next ones we shall address ADP.

After obvious modifications, the application of the principle of optimality can be stated as it has been done in the previous section, leading directly to a recursive relationship of the same type as (6.10). We have then to define the “minimum cost-to-go functions” $J_t^\circ(\mathbf{x}_t)$ corresponding to the “minimum time-to-go functions” (6.8). Before that, however, we must specify the domains on which the functions $J_t^\circ(\mathbf{x}_t)$ take their values. Let us denote these sets by \bar{X}_t , $t = 1, \dots, T - 1$. From a practical point of view, it is difficult to imagine that Constraints (6.2) and (6.3) are not present. If these constraints are not explicitly given, practical considerations (physical, economical, etc.) can be used to delimit these variables at least qualitatively.

Then, the sets \bar{X}_t are determined recursively as follows (of course, \bar{X}_0 simply reduces to the starting point $\mathbf{x}_0 = \hat{\mathbf{x}}$):

$$\begin{aligned} \bar{X}_{t+1} &= X_{t+1} \cap \{\mathbf{x}_{t+1} \in \mathbb{R}^n : \mathbf{x}_{t+1} = \mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t), \mathbf{x}_t \in \bar{X}_t, \mathbf{u}_t \in U_t(\mathbf{x}_t)\}, \\ t &= 0, 1, \dots, T - 1, \end{aligned} \quad (6.14a)$$

$$\bar{X}_0 = \{\hat{\mathbf{x}}\}. \quad (6.14b)$$

Of course, for a certain set \bar{X}_{t+1} not to be empty, a nonempty subset of X_{t+1} must be reachable from \bar{X}_t under the action of controls $\mathbf{u}_t \in U_t(\mathbf{x}_t)$ for some $\mathbf{x}_t \in \bar{X}_t$. It is almost needless to say that, in general, the recursive procedure outlined by (6.14) cannot be carried out analytically, unless the state equation (6.1) and the shape of the sets X_t and $U_t(\mathbf{x}_t)$ are sufficiently simple. This is the case, for example, when the dynamic system is linear and the constraint sets X_t and $U_t(\mathbf{x}_t)$ are polyhedra (see [5] for a comprehensive treatment on this issue). However, it is reasonable to assume that some numerical technique may be conceived in order to implement (6.14) in an approximate way. In this case, one should not think that the determination of the sets of the “tube” $\{\bar{X}_1, \bar{X}_2, \dots, \bar{X}_T\}$ is too critical a matter. Even if one gets a quite rough estimate of the sets X_t , it can “enlarge” them properly. This does not cause errors in the DP procedure but a mere waste of time since the optimal costs-to-go will also be computed in points that will never be visited by the state trajectory. We shall be more clear when we will address the discretization of the sets \bar{X}_t .

This said, the minimum or optimal cost-to-go functions are defined as

$$\begin{aligned} J_T^\circ(\mathbf{x}_T) &\triangleq h_T(\mathbf{x}_T), \quad \mathbf{x}_T \in \bar{X}_T, \\ J_t^\circ(\mathbf{x}_t) &\triangleq \min_{\mathbf{u}_k \in U_k(\mathbf{x}_k), f_k(\mathbf{x}_k, \mathbf{u}_k) \in X_{k+1}} \left[\sum_{k=t}^{T-1} h_k(\mathbf{x}_k, \mathbf{u}_k) + h_T(\mathbf{x}_T) \right], \\ &\quad \mathbf{x}_t \in \bar{X}_t, \quad t = 0, 1, \dots, T-1. \end{aligned} \quad (6.15)$$

In particular, $J_0^\circ(\mathbf{x}_0)$ corresponds to the total minimum cost.

As the controls $\mathbf{u}_1, \dots, \mathbf{u}_{T-1}$ do not affect the cost $h_0(\mathbf{x}_0, \mathbf{u}_0)$ of the first transition, we can use Definition (6.15)¹ and write

$$\begin{aligned} J_0^\circ(\mathbf{x}_0) &= \min_{\mathbf{u}_0, \dots, \mathbf{u}_{T-1}} \left[\sum_{k=0}^{T-1} h_k(\mathbf{x}_k, \mathbf{u}_k) + h_T(\mathbf{x}_T) \right] \\ &= \min_{\mathbf{u}_0} \left\{ h_0(\mathbf{x}_0, \mathbf{u}_0) + \min_{\mathbf{u}_1, \dots, \mathbf{u}_{T-1}} \left[\sum_{k=0}^{T-1} h_k(\mathbf{x}_k, \mathbf{u}_k) + h_T(\mathbf{x}_T) \right] \right\} \\ &= \min_{\mathbf{u}_0} [h_0(\mathbf{x}_0, \mathbf{u}_0) + J_1^\circ(\mathbf{x}_1)]. \end{aligned}$$

By carrying out the same procedure for $t = 1, \dots, T-1$, we obtain the following recursive equation for DP:

$$J_T^\circ(\mathbf{x}_T) = h_T(\mathbf{x}_T), \quad \mathbf{x}_T \in \bar{X}_T, \quad (6.16a)$$

$$\begin{aligned} J_t^\circ(\mathbf{x}_t) &= \min_{\mathbf{u}_t \in U_t(\mathbf{x}_t), f_t(\mathbf{x}_t, \mathbf{u}_t) \in X_{t+1}} [h_t(\mathbf{x}_t, \mathbf{u}_t) + J_{t+1}^\circ(\mathbf{x}_{t+1})], \\ &\quad \mathbf{x}_t \in \bar{X}_t, \quad t = T-1, T-2, \dots, 0. \end{aligned} \quad (6.16b)$$

¹Sometimes, for notational simplicity, we shall omit the constraints on the control vectors.

Then, the determination of the sequence of optimal controls $\mathbf{u}_0^\circ, \mathbf{u}_1^\circ, \dots, \mathbf{u}_{T-1}^\circ$ can be performed in the following two phases:

1. In the *backward* phase of the DP procedure, at the stages $t = T - 1, T - 2, \dots, 0$, the optimization problems defined by Equation (6.16b) are solved. More specifically, at stage $T - 1$, the minimum cost-to-go function $J_{T-1}^\circ(\mathbf{x}_{T-1})$, $\mathbf{x}_{T-1} \in \bar{X}_{T-1}$ is calculated together with the optimal control function $\mathbf{u}_{T-1}^\circ = \mu_{T-1}^\circ(\mathbf{x}_{T-1})$, $\mathbf{x}_{T-1} \in \bar{X}_{T-1}$. Then, the knowledge of $J_{T-1}^\circ(\mathbf{x}_{T-1})$ allows the determination of $J_{T-2}^\circ(\mathbf{x}_{T-2})$ and $\mathbf{u}_{T-2}^\circ = \mu_{T-2}^\circ(\mathbf{x}_{T-2})$. When the recursive procedure has been completed, the minimum value of the cost (6.4), the first optimal control vector \mathbf{u}_0° , and the optimal control functions

$$\mathbf{u}_1^\circ = \mu_1^\circ(\mathbf{x}_1), \dots, \mathbf{u}_{T-1}^\circ = \mu_{T-1}^\circ(\mathbf{x}_{T-1}) \quad (6.17)$$

are available (but not yet the optimal control vectors).

2. For the initial state $\mathbf{x}_0 = \hat{\mathbf{x}}$, the first optimal control \mathbf{u}_0° is computed. The vector $\mathbf{x}_1^\circ = f_0(\hat{\mathbf{x}}, \mathbf{u}_0^\circ)$ of the optimal trajectory is determined by using State Equation (6.1), and the second optimal control $\mathbf{u}_1^\circ = \mu_1^\circ(\mathbf{x}_1^\circ)$ is calculated. Then, by proceeding analogously, the sequence of the optimal control vectors $\mathbf{u}_0^\circ, \mathbf{u}_1^\circ, \dots, \mathbf{u}_{T-1}^\circ$ is determined, thus concluding the *forward* phase of the DP procedure.

Remark 6.1 An important variant of Problem C1 arises whenever the decision-maker (DM) does not know the initial value of \mathbf{x}_0 before the decision process starts. It only knows that \mathbf{x}_0 belongs to a given region $X_0 \subseteq \mathbb{R}^d$. As soon as the process starts, the DM can observe perfectly \mathbf{x}_0 at the instant $t = 0$. It may occur that the DM has not enough time to compute the optimal control \mathbf{u}_0° for the specific value of \mathbf{x}_0 it has observed. This happens if the time for computing \mathbf{u}_0° is not “short” enough as compared with the length of the time interval Δt between two subsequent stages. Then, at time $t = 0$ the DM should have at its disposal an optimal control function

$$\mathbf{u}_0^\circ = \mu_0^\circ(\mathbf{x}_0), \quad \forall \mathbf{x}_0 \in X_0 \quad (6.18)$$

(computed *offline* or *a priori*) that generates \mathbf{u}_0° for any possible initial state. Function (6.18) can be simply derived by performing also the minimizations (6.16b) at the stage $t = 0$ for any $\mathbf{x}_0 \in X_0$. \triangleleft

In Problem C1, the generation of the optimal control vectors can be performed *a priori*, i.e., *offline*. In other words, the absence of random variables acting on the dynamic system allows the DM not to modify the control vectors during the *online* control phase. Therefore, it is not necessary to observe the state vector by a feedback channel. The state vectors of the optimal trajectory $\mathbf{x}_1^\circ, \dots, \mathbf{x}_T^\circ$ can also be determined *a priori*, and a closed-loop control of the system does not have any advantage over an open-loop one. These considerations result from assuming complete absence of random disturbances. If, in contrast with the assumption of a deterministic framework, some unpredicted events (e.g., disturbances) should remove the state from

the optimal trajectory, the DM should solve a new Problem C1 by reoptimizing the remainder of the trajectory. However, this might not be possible if the time for computing the optimal control for the specific value of the state the DM has observed is not “short” enough as compared with the length of the time interval Δt between two subsequent stages. In such a case, a DM that has stored in memory not only the controls \mathbf{u}_t° but also the functions $\mu_t^\circ(\mathbf{x}_t)$ would be able to react to the unpredicted events in an optimal way.

We formalize the discussion above and in Remark 6.1 by stating the following extension of Problem C1:

Problem C1'. *For any $\mathbf{x}_0 \in X_0$, find the sequence of the optimal control functions*

$$\mathbf{u}_0^\circ = \mu_0^\circ(\mathbf{x}_0), \mathbf{u}_1^\circ = \mu_1^\circ(\mathbf{x}_1), \dots, \mathbf{u}_{T-1}^\circ = \mu_{T-1}^\circ(\mathbf{x}_{T-1})$$

that minimizes the cost J , subject to Constraints (6.1), (6.2), and (6.3). \triangleleft

The functions $\mu_t^\circ(\mathbf{x}_t)$ in Problem C1' – determined in the backward phase of the calculation procedure – have the structure of closed-loop control functions and are only used in the feedforward phase of the computation. As said above, they allow one to cope with the situation in which something unpredicted should remove the state from the optimal trajectory. However, we must say in advance that, if we are in a position to predict that the action of the disturbances will not be negligible and if it is possible to know the statistical properties of the disturbances themselves, then the correct solution lies in stating the optimal control problem in a stochastic framework and in deriving optimal closed-loop control functions. This will be the subject of Chaps. 7 and 8.

Finally, note that the minimizations in (6.16b) call for the solution of the following NLP problems.

Problem 6.2 *For all stages $T - 1, T - 2, \dots, 0$ and any $\mathbf{x}_t \in \bar{X}_t$, find optimal control vectors \mathbf{u}_t° that minimize the cost*

$$\mathcal{J}_t(\mathbf{x}_t, \mathbf{u}_t) \triangleq h_t(\mathbf{x}_t, \mathbf{u}_t) + J_{t+1}^\circ[\mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t)] \quad (6.19)$$

subject to the constraints

$$\mathbf{u}_t \in U_t(\mathbf{x}_t),$$

$$\mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t) \in X_{t+1}.$$

\triangleleft

Remark 6.2 We have to point out that Problems 6.2 have meanings that are more conceptual than practical. Indeed, if the sets \bar{X}_t are made of an infinite number of states, then the number of NLP problems to be solved at each stage is also infinite. Therefore, we are forced to resort to some form of discretization of the sets \bar{X}_t in order to solve a finite number of such problems. This implies approximations that will be addressed in the next sections. The formulation of Problem C1' will be particularly useful to deal with such approximation issues. \triangleleft

6.4 Sampling of the State Space and Application of FSP Functions: Approximate Dynamic Programming

Owing to the very general assumptions under which Problems C1 and C1' have been stated, the DP equation can only be solved analytically in a few cases, typically in the LQ framework. Then, in general, we have to look for an approximate approach. This usually begins with the sampling of the state space. In this way, Recursive Equation (6.16) has to be solved only for a finite number of state vectors. However, as we shall see later in this chapter, the optimal costs-to-go must be known not only in the discretized states but also in the entire sets \bar{X}_t . This can be obtained by computing approximate costs-to-go functions (on the basis of the values of the costs-to-go computed in the discretized states) that will take on the form, for example, of FSP functions. These two approximations will give rise to approximate dynamic programming.

6.4.1 Sampling of the State Space

At each stage t , we select L_t discretized points $\mathbf{x}_t^{(l)}$, $l = 1, \dots, L_t$ belonging to the admissible sets \bar{X}_t , thus defining the sets

$$X'_t \triangleq \{\mathbf{x}_t^{(l)} \in \bar{X}_t : l = 1, \dots, L_t\}, \quad t = 0, 1, \dots, T - 1. \quad (6.20)$$

As in Chap. 4, in the next paragraphs, we shall call the integers L_t *sample cardinalities*. Since FSP functions will be used to approximate the optimal cost-to-go and control functions, both the model complexity and the sample cardinality should grow “moderately” with the dimension d of the state vector. Indeed, we shall try to mitigate Bellman’s *curse of dimensionality* by exploiting: (i) the properties of suitable FSP functions and (ii) efficient sampling techniques.

We consider three main possibilities of discretizing the sets \bar{X}_t to obtain the sets X'_t :

1. The discretized sets X'_t take on the form of *full uniform grids*.

The simplest (until some time ago, also the most widely used) approach of this kind consists of discretizing each component of the state vector \mathbf{x}_t into q_t equally spaced values. Of course, each state component could be discretized into a different number of levels but, without loss of generality, we assume that the number of discretization values is the same for all the components in order to avoid notational complications. Therefore, for each decision stage t , we obtain a full uniform grid in which the total number of discretized nodes grows with d as q_t^d . This exponential growth leads the sample cardinalities L_t to incur the curse of dimensionality – i.e., an unacceptable growth with d of memory and computational requirements – and limits the applicability of full uniform grids to a small dimension of the state.

2. The sets X'_t are obtained via the Monte Carlo approach by drawing i.i.d. sample vectors $\mathbf{x}_t^{(l)}$ according to an uniform probability density.

This way, the cardinalities of the sets X'_t do not depend “structurally” on the state dimension d (as in the full uniform grids). As one can intuit, the cardinalities L_t depend essentially on the accuracy one wants to achieve in approximating the optimal cost-to-go and control functions. Of course, as in the next point, it can be expected that, for a desired accuracy, L_t increases with d but not necessarily in an exponential way.

3. The sets X'_t are given by deterministic sequences of nodes that are spread in the “most uniform way.”

Examples of this discretization are quasi-Monte Carlo sequences (described in Chap. 4), orthogonal arrays, Latin hypercubes, etc. In this case, the use of deterministic sequences enables one to obtain deterministic results. Furthermore, as we have discussed, the resulting sets X'_t are generally “more uniformly spread” than the ones generated by the uniform probability densities mentioned in the previous point (see Fig. 4.13). For these reasons, from now on we shall assume that the sets X'_t are generated by deterministic techniques and, more specifically, by the low-discrepancy sequences described in Sect. 4.4.7. As we shall see in Sect. 6.6.2, this will enable us to apply ADP with the hope of getting small estimation errors in computing the optimal cost-to-go and control functions.

Note that the dependence of the sample cardinality L_t on the stage t is due to the fact that the regularity of the functions to be approximated and the sets \bar{X}_t may be time-varying. Therefore, the sample cardinality L_t may also be time-varying in order to obtain an uniform approximation accuracy at various stages. As in Sect. 4.1, we stress that the factors on which the generation of a generic set X'_t depends are given by

- (i) The set \bar{X}_t , from which the discretized points $\mathbf{x}_t^{(l)}$ are selected.
- (ii) The number L_t of discretized state vectors (input samples).
- (iii) The algorithm by which the L_t discretized vectors are selected. The algorithm may generate the vectors $\mathbf{x}_t^{(l)}$ according to either probabilistic or deterministic criteria (see the MC and quasi-MC methods, respectively).

Remark on Notation 6.3 On the basis of the three elements reported above as describing the construction of the sample set X'_t , we should write X'_{tL_t} instead of X'_t . However, for simplicity, we continue omitting the sample cardinality L_t and writing simply X'_t . The same can be repeated for the I/O sample set Σ_t . \triangleleft

Remark 6.4 Notice that all the three aforementioned procedures are “blind”, in the sense that once we have defined the number L_t , we obtain the whole set X'_t in one shot, without iteratively selecting the next optimal sampling point on the basis of the previous ones and on the previously observed values of the cost-to-go function. Since the latter “active” approach typically involves computationally intensive procedures to select the new points, here we shall not consider it, and illustrate the ADP approach without this additional burden on the computational requirements. However, the

application of *active learning* algorithms to dynamic programming is an interesting and promising research direction. The interested reader can consult, for instance, [14] and the references therein for some recent developments. \triangleleft

6.4.2 The Recursive Equation of ADP

For all the vectors $\mathbf{x}_t^{(l)} \in X'_t$, we want to compute the optimal cost-to-go functions $J_t^\circ(\mathbf{x}_t^{(l)})$. In order to derive these values, we need to perform Minimizations (6.16b), i.e.,

$$J_T^\circ(\mathbf{x}_T) = h_T(\mathbf{x}_T), \quad \mathbf{x}_T \in \bar{X}_T, \quad (6.21a)$$

$$J_t^\circ(\mathbf{x}_t^{(l)}) = \min_{\mathbf{u}_t} \left\{ h_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t) + J_{t+1}^\circ \left[\mathbf{f}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t) \right] \right\},$$

$$\mathbf{x}_t^{(l)} \in X'_t, \quad t = T - 1, T - 2, \dots, 0. \quad (6.21b)$$

A first obstacle to Minimizations (6.21b) is given by the fact that each next-stage vector $\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t^{(l)}, \mathbf{u}_t)$ falls into the admissible domain \bar{X}_{t+1} but generally does not coincide with any previously discretized state $\mathbf{x}_{t+1}^{(l)} \in X'_{t+1}$. Then, some type of approximation procedure is needed to compute the optimal cost $J_{t+1}^\circ \left[\mathbf{f}(\mathbf{x}_t^{(l)}, \mathbf{u}_t) \right]$. Let us describe in detail how the ADP recursive equation (6.21) has to be modified in order to take into account the state discretization.

- *Stage T – 1*

For each $\mathbf{x}_{T-1}^{(l)} \in X'_{T-1}$, compute

$$J_{T-1}^\circ(\mathbf{x}_{T-1}^{(l)})$$

$$= \min_{\mathbf{u}_{T-1}} \left\{ h_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1}) + h_T \left[\mathbf{f}_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1}) \right] \right\}. \quad (6.22)$$

Obviously, we have no problem about the discretization, since the final cost function $h_T(\mathbf{x}_T)$ is known for any $\mathbf{x}_T \in \bar{X}_T$. The minimizations in (6.22) call for the solution of L_{T-1} NLP problems, one for each vector $\mathbf{x}_{T-1}^{(l)}$.

- *Stage T – 2*

At this stage, we need to compute the costs $J_{T-2}^\circ(\mathbf{x}_{T-2})$. As we said before, the function $J_{T-1}^\circ(\mathbf{x}_{T-1})$ is known at the discretized points $\mathbf{x}_{T-1}^{(l)} \in X'_{T-1}$ but not, in general, at the points

$$\mathbf{x}_{T-1} = \mathbf{f}_{T-2}(\mathbf{x}_{T-2}^{(l)}, \mathbf{u}_{T-2}), \quad \mathbf{x}_{T-2}^{(l)} \in X'_{T-2},$$

so we have to estimate the values assumed by the function $J_{T-1}^\circ(\mathbf{x}_{T-1})$ at such points. A large number of techniques for performing this estimation have been proposed in the literature. The simplest one consists of approximating $J_{T-1}^\circ[\mathbf{f}_{T-2}(\mathbf{x}_{T-2}^{(l)}, \mathbf{u}_{T-2})]$ by the value of $J_{T-1}^\circ(\mathbf{x}_{T-1}^{(l)})$, where $\mathbf{x}_{T-1}^{(l)} \in X'_{T-1}$ is the discretized state that is nearest to $\mathbf{f}_{T-2}(\mathbf{x}_{T-2}^{(l)}, \mathbf{u}_{T-2})$ (nearest neighbor). Other techniques involve resorting to linear combinations of fixed basis functions (LCFBFs; see Sect. 3.1) like algebraic polynomials.

Let us extend the case in which LCFBFs are used to the more general case in which we want to approximate the costs $J_{T-1}^\circ(\mathbf{x}_{T-1})$ by FSP functions belonging to the nested families

$$\mathcal{A}_{T-1,n_{T-1}} \triangleq \left\{ \tilde{J}(\mathbf{x}_{T-1}, \mathbf{w}_{T-1,n_{T-1}}) : \mathbf{x}_{T-1} \in \mathbb{R}^d, \mathbf{w}_{T-1,n_{T-1}} \in \mathbb{R}^{\mathcal{N}(n_{T-1})} \right\}, \quad n_{T-1} = 1, 2, \dots . \quad (6.23)$$

In the special case in which FSP functions are given by one-hidden-layer (OHL) networks, they take on the form (see 3.3) and hence

$$\tilde{J}(\mathbf{x}_{T-1}, \mathbf{w}_{T-1,n_{T-1}}) = \sum_{i=1}^{n_{T-1}} c_{i,T-1} \varphi(\mathbf{x}_{T-1}, \kappa_{i,T-1}) + c_{T-1}, \quad n_{T-1} = 1, 2, \dots , \quad (6.24)$$

where the vectors $\mathbf{w}_{T-1,n_{T-1}}$ are defined in an obvious way.

The optimal approximation of the cost $J_{T-1}^\circ(\mathbf{x}_{T-1})$ by FSP functions belonging to the families (6.23) can be obtained by using the least-squares criterion (this is the most common choice, but other loss functions may also be adopted). To this end, we define the I/O sample set (see (4.3))

$$\Sigma_{T-1} \triangleq \left\{ \left(\mathbf{x}_{T-1}^{(l)}, J_{T-1}^\circ(\mathbf{x}_{T-1}^{(l)}) \right) : \mathbf{x}_{T-1}^{(l)} \in X'_{T-1} \right\}. \quad (6.25)$$

Then, we solve the following NLP problem:

$$\min_{\mathbf{w}_{T-1,n_{T-1}}} \frac{1}{L_{T-1}} \sum_{l=1}^{L_{T-1}} \left[J_{T-1}^\circ(\mathbf{x}_{T-1}^{(l)}) - \tilde{J}(\mathbf{x}_{T-1}^{(l)}, \mathbf{w}_{T-1,n_{T-1}}) \right]^2. \quad (6.26)$$

The parameter vector that solves Minimization Problem (6.26) is given by $\mathbf{w}_{T-1,n_{T-1}, \Sigma_{L_{T-1}}}^\circ$. However, to simplify the notation, we simply write $\mathbf{w}_{T-1,n_{T-1}, \Sigma_{T-1}}^\circ$, that is, we omit the explicit dependence of Σ_{T-1} on the sample cardinality L_{T-1} (see Remark on Notation 6.3).

Note that the adoption of the least-squares criterion can bring computational benefits if one chooses LCFBFs. These benefits are lost if one uses OHL networks or

more complex FSP functions. However, as we shall see later, the use of suitable OHL networks may enable us to mitigate the danger of incurring the curse of dimensionality in computing the approximate optimal costs $\tilde{J}_t(\mathbf{x}_t, \mathbf{w}_{t,n,\Sigma_t}^\circ)$. Now, at stage $T - 2$, the DP equation becomes

$$\begin{aligned}\bar{J}_{T-2}^\circ(\mathbf{x}_{T-2}^{(l)}) &= \min_{\mathbf{u}_{T-2}} \left\{ h_{T-2}(\mathbf{x}_{T-2}^{(l)}, \mathbf{u}_{T-2}) \right. \\ &\quad \left. + \tilde{J}[f_{T-2}(\mathbf{x}_{T-2}^{(l)}, \mathbf{u}_{T-2}), \mathbf{w}_{T-1,n_{T-1},\Sigma_{T-1}}^\circ] \right\}, \quad \mathbf{x}_{T-2}^{(l)} \in X'_{T-2}.\end{aligned}\quad (6.27)$$

Note that in (6.22) $J_{T-1}^\circ(\mathbf{x}_{T-1}^{(l)})$ is the “true” value of the optimal cost-to-go, provided that no computational error affects the minimization procedure. Let us assume this error to be negligible. Instead, $\bar{J}_{T-2}^\circ(\mathbf{x}_{T-2}^{(l)})$ can only be an approximation of $J_{T-2}^\circ(\mathbf{x}_{T-2}^{(l)})$ because of the presence of the FSP function $\tilde{J}(\cdot, \mathbf{w}_{T-1,n_{T-1},\Sigma_{T-1}}^\circ)$ in (6.27). Analogously, at stages $t = T - 3, T - 4, \dots, 0$, the costs $\bar{J}_t^\circ(\mathbf{x}_t^{(l)})$, which are defined in a similar way as in (6.27), approximate the associated optimal costs $J_t^\circ(\mathbf{x}_t^{(l)})$. In Fig. 6.4, we show pictorially the true and approximate optimal cost-to-go functions at stages $T - 1$ and $T - 2$ for a scalar state x_t .

Remark on Notation 6.5 In (6.27), we put the symbol “ \circ ” over the cost $\bar{J}_{T-2}^\circ(\mathbf{x}_{T-2}^{(l)})$. This is the second symbol we have introduced besides the symbol “ \sim ”. Not to confuse

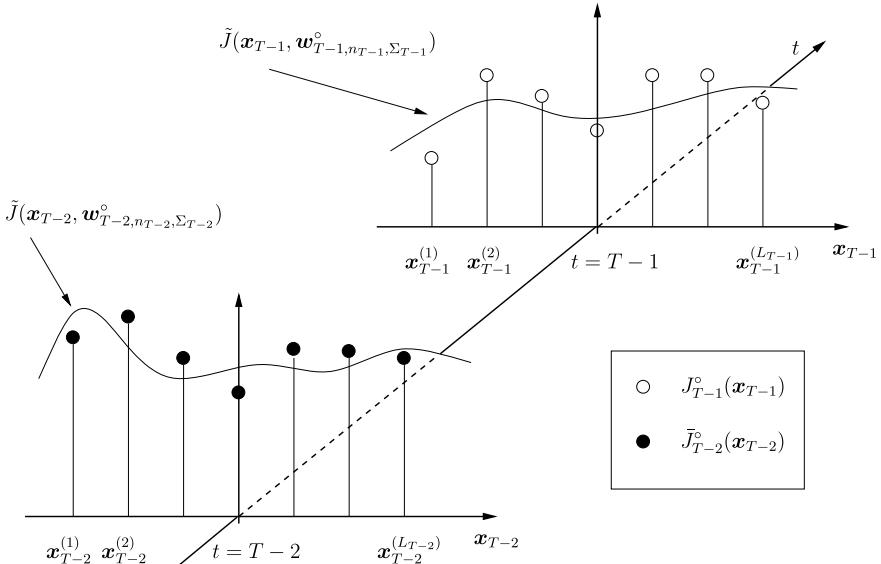


Fig. 6.4 True and FSP optimal cost-to-go functions at the stage $T - 1$, and approximate and FSP optimal cost-to-go functions at the stage $T - 2$

the reader with too many symbols over the functions and variables, it is now appropriate to clarify the meaning of such symbols, which are used in (6.27) and which will be used later on in the ADP recursive equation. The symbol “ \circ ” denotes the cost-to-go functions \bar{J}_t° and the control functions $\bar{\mu}_t^\circ$ derived by the minimization performed in the backward phase of DP. The symbol “ $\tilde{\circ}$ ” characterizes the cost-to-go functions \tilde{J}_t° and the control functions $\tilde{\mu}_t^\circ$ implemented by the FSP optimal functions that approximate the functions \bar{J}_t° and $\bar{\mu}_t^\circ$, respectively. The costs \bar{J}_t° and \tilde{J}_t° will be called *approximate optimal* and *FSP optimal cost-to-go functions*, respectively. Sometimes, when there will be no risk of ambiguity, they may be simply called “optimal” cost-to-go functions. A similar terminology will be used for $\bar{\mu}_t^\circ$ and $\tilde{\mu}_t^\circ$.

Obviously, a distinction should also be made between the optimal control and state vectors and their approximations. However, we shall not do it, not only to avoid too complicated a notation but, most of all, because there is no risk of confusion. Indeed, the essential difference must be pointed out between the costs \bar{J}_t° and \tilde{J}_t° . Instead, once one is aware that the trajectories of the control and the state are approximations of the optimal ones, to emphasize these approximations is less important. Then, we simply define the controls generated by both the functions $\tilde{\mu}_t^\circ$ and $\bar{\mu}_t^\circ$ as \mathbf{u}_t° . \triangleleft

- Stages $T - 3, T - 4, \dots, 0$

The computational scheme, used at stages $T - 1$ and $T - 2$ to derive Equations (6.22) and (6.27), can be repeated backwards at stages $T - 3, T - 4, \dots, 0$. More specifically, we constrain the optimal cost-to-go functions $J_t^\circ(\mathbf{x}_t)$ to take on the form of FSP functions belonging to nested families similar to Sets (6.23), that is,

$$\begin{aligned} \mathcal{A}_{tn_t} &\triangleq \left\{ \tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t}) : \mathbf{x}_t \in \mathbb{R}^d, \mathbf{w}_{tn_t} \in \mathbb{R}^{\mathcal{N}(n_t)} \right\}, \\ t &= 0, 1, \dots, T - 2, n_t = 1, 2, \dots. \end{aligned} \quad (6.28)$$

The FSP functions $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{t,n_t})$ are optimized on the basis of the I/O sample sets that are like (6.25), i.e.,

$$\Sigma_t \triangleq \left\{ \left(\mathbf{x}_t^{(l)}, \bar{J}_t^\circ(\mathbf{x}_t^{(l)}) \right) : \mathbf{x}_t^{(l)} \in X'_t \right\}, \quad t = 0, 1, \dots, T - 2. \quad (6.29)$$

The parameter vectors of the FSP functions belonging to \mathcal{A}_{n_t} are computed by solving NLP problems similar to (6.26), that is,

$$\min_{\mathbf{w}_{tn_t}} \frac{1}{L_t} \sum_{l=1}^{L_t} \left[\bar{J}_t^\circ(\mathbf{x}_t^{(l)}) - \tilde{J}(\mathbf{x}_t^{(l)}, \mathbf{w}_{tn_t}) \right]^2, \quad t = T - 2, T - 3, \dots, 0. \quad (6.30)$$

The optimal parameter vectors are then given by $\mathbf{w}_{tn_t, \Sigma_{L_t}}^\circ$. As in stage $T - 1$, this vector is simply written as $\mathbf{w}_{tn_t, \Sigma_t}^\circ$ instead of $\mathbf{w}_{tn_t, \Sigma_{L_t}}^\circ$.

We have assumed that the FSP functions remain of the same type stage after stage. If so, the integer t does not appear as an argument of the functions since they depend on the stage t only through the model complexity n_t and the time-varying parameter vector \mathbf{w}_{tn_t} . As we have remarked about the sample cardinality L_t , the model complexity n_t may also depend on the stage t . This is because the regularity of the functions to be approximated and the sets X_t may be time-varying.

The above-described ADP procedure, which is the approximate version of Recursive DP Equation (6.16b), is then given by

$$\begin{aligned} \bar{J}_{T-1}^{\circ}(\mathbf{x}_{T-1}^{(l)}) &= \min_{\substack{\mathbf{u}_{T-1} \in U_{T-1}(\mathbf{x}_{T-1}^{(l)}) \\ f_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1}) \in X_T}} \left\{ h_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1}) + h_T[f_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1})] \right\}, \\ \mathbf{x}_{T-1}^{(l)} &\in X'_{T-1}, \end{aligned} \quad (6.31a)$$

$$\begin{aligned} \bar{J}_t^{\circ}(\mathbf{x}_t^{(l)}) &= \min_{\substack{\mathbf{u}_t \in U_t(\mathbf{x}_t^{(l)}) \\ f_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t) \in X_{t+1}}} \left\{ h_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t) + \tilde{J}[f_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t), \mathbf{w}_{t+1, n_{t+1}, \Sigma_{t+1}}^{\circ}] \right\}, \\ \mathbf{x}_t^{(l)} &\in X'_t, \quad t = T-2, T-3, \dots, 0. \end{aligned} \quad (6.31b)$$

Of course, in the case of Problem C1, in (6.31b) the set X'_0 simply reduces to the starting point $\mathbf{x}_0 = \hat{\mathbf{x}}$ and $\bar{J}_t^{\circ}(\mathbf{x}_0)$ is the (approximate) optimal cost of the decision process.

As pointed out by (6.26) and (6.30), at any stage of the recursive ADP equations (6.31), one has to determine optimal parameter vectors $\mathbf{w}_{tn_t, \Sigma_t}^{\circ}$, by which the FSP functions $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t})$ approximate the costs-to-go $\bar{J}_t^{\circ}(\mathbf{x}_t)$. If one continues to use the least-squares criterion, then the optimal parameter vectors are obtained by solving the following NLP problems.

Problem 6.3 For all stages $T-1, \dots, 0$, find the optimal parameter vector $\mathbf{w}_{tn_t, \Sigma_t}^{\circ}$ that minimizes the quadratic error

$$\frac{1}{L_t} \sum_{l=1}^{L_t} \left[\bar{J}_t^{\circ}(\mathbf{x}_t^{(l)}) - \tilde{J}(\mathbf{x}_t^{(l)}, \mathbf{w}_{tn_t}) \right]^2. \quad (6.32)$$

△

As observed previously, the choice of the least-squares criterion for obtaining the FSP optimal cost-to-go functions is rather questionable. As is well known, this criterion gives computational advantages if LCFBFs are used; otherwise, NLP algorithms are needed to determine $\mathbf{w}_{tn_t, \Sigma_t}^{\circ}$. A serious drawback of this criterion (and of other loss functions) comes from the impossibility of guaranteeing that the errors are bounded by a given threshold. A better approach would be the use of a min-max criterion. This implies replacing the quadratic error (6.32) with the maximum error

$$\max_{l \in \{1, \dots, L_t\}} |\tilde{J}_t^\circ(\mathbf{x}_t^{(l)}) - \tilde{J}(\mathbf{x}_t^{(l)}, \mathbf{w}_{tn_t})| \quad (6.33)$$

and minimizing it with respect to \mathbf{w}_{tn_t} .

Remark 6.6 Note that the approximations related to the use of the FSP functions and the limited cardinality L_t of X'_t give rise to the *approximation* and *estimation errors* examined in Chap. 4, respectively. We shall address these errors in Sect. 6.6. \triangleleft

In order to compute the approximate optimal costs-to-go $\tilde{J}_t^\circ(\mathbf{x}_t)$, one has to minimize the right-hand sides of Equation (6.31b) with respect to the control vectors \mathbf{u}_t . Therefore, besides Problems 6.3, one has to solve a second NLP problem for any decision stage and for any sample vector $\mathbf{x}_t^{(l)}$. This will be addressed in the following section.

6.4.3 Solving the Minimization Problems in the ADP Equation

In solving ADP Equation (6.31), we have seen the alternation of two NLP problems. One is given by the approximation Problem 6.3 and the other aims at minimizing the right-hand side of (6.31b). We concentrate on the latter.

When we stated Problem 6.2, we pointed out that this NLP problem has more a conceptual than a practical meaning since it should be solved for an infinite number of states $\mathbf{x}_t \in \bar{X}_t$. Instead, the introduction of the sample sets X'_t modifies Problems 6.2 and gets feasibility since we have to solve a finite number of NLP problems, each for any state $\mathbf{x}_t^{(l)} \in X'_t$. These problems, which constitute the “practical” (though approximate) counterpart of Problem 6.2, can be stated in the following form.

Problem 6.4 For all stages $T - 2, T - 3, \dots, 0$, and any $\mathbf{x}_t^{(l)} \in X'_t$, find the optimal control vector $\mathbf{u}_t^{(l)\circ}$ that minimizes the cost

$$\tilde{\mathcal{J}}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t) \triangleq h_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t) + \tilde{J}[\mathbf{f}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t), \mathbf{w}_{t+1, n_{t+1}, \Sigma_{t+1}}] \quad (6.34)$$

subject to the constraints

$$\mathbf{u}_t \in U_t(\mathbf{x}_t^{(l)}), \quad (6.35)$$

$$\mathbf{f}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t) \in X_{t+1}. \quad (6.36)$$

For the stage $T - 1$ and any $\mathbf{x}_{T-1}^{(l)} \in X'_{T-1}$, find the optimal control vector $\mathbf{u}_{T-1}^{\circ(l)}$ that minimizes the cost

$$\tilde{\mathcal{J}}_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1}) \triangleq h_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1}) + h_T[\mathbf{f}_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1})] \quad (6.37)$$

subject to Constraints (6.35) and (6.36) with $t = T - 1$. \triangleleft

Note that in (6.34) and (6.37) we wrote $\tilde{\mathcal{J}}_t$ to distinguish this case from \mathcal{J}_t defined in (6.19). For uniformity of notation, we also write $\tilde{\mathcal{J}}_{T-1}$ in (6.37). The ADP computational procedure is summarized in the flowchart of Fig. 6.5.

The solution of Equation (6.31) or, equivalently, of Problem 6.4 gives the *approximate optimal cost-to-go functions* $\bar{J}_t^\circ(\mathbf{x}_t^{(l)})$ for any discretized state (they are used in Problem 6.3) and the *approximate optimal controls*

$$\mathbf{u}_t^{\circ(l)} \triangleq \bar{\mu}_t^\circ(\mathbf{x}_t^{(l)}), \quad \mathbf{x}_t^{(l)} \in X'_t, \quad t = 0, 1, \dots, T - 1. \quad (6.38)$$

Of course, the functions $\bar{\mu}_t^\circ: \bar{X}_t \rightarrow U_t$ (where $U_t \triangleq \cup_{\mathbf{x}_t \in \bar{X}_t} U(\mathbf{x}_t)$) are known only in the points $\mathbf{x}_t^{(l)}$. We use the term “approximate optimal” (together with the symbol “ \circ ”; see the Remark on Notation 6.5) because the values of \bar{J}_t° and $\bar{\mu}_t^\circ$ are optimal within the limits imposed by the approximations we have introduced. We recall them: (i) the state discretization; (ii) the use of the FSP functions; and (iii) the errors associated with the approximate minimizations in Problem 6.4 owing to the use of NLP algorithms.

Note that, in general, the determination of the errors in applying the ADP is not an easy task since one has to consider not only the effects of the approximations at a given stage t but also the backward propagation of the errors in the recursive computation of Equation (6.31). Bellman already studied this propagation in his seminal work [1]. For more recent results, we refer to [10, 11], where upper bounds on the backpropagated error in ADP are derived in a deterministic and a stochastic framework, respectively. The properties of polynomially complex approximating FSP functions taking on the form of OHL networks are exploited therein. The model of T -stage optimal control problem adopted in [10, 11] is expressed in terms of “correspondences” instead of dynamic systems, as is typically done, e.g., in economics (see, for instance, [15]). For a discussion on the equivalence between the two models, see [11, pp. 33–34] and [15].

As regards the minimization techniques to be used for the solution of Problems 6.3 and 6.4, one may resort to the direct- or gradient-based algorithms mentioned in Sects. 5.3.1 and 5.3.2 for the solution of Problem P_n , respectively. Cost Functions (6.32), (6.34), and (6.37) take the place of the cost $\tilde{F}(\mathbf{w}_n)$. No expectation is now involved, so that only the left-hand equality in (5.26) has to be considered. If one chooses a direct method, then the MC random search (see Sect. 5.4.1) and the quasi-MC one (see Sect. 5.4.2) can be important alternatives to the classic methods mentioned in Sect. 5.3.1.

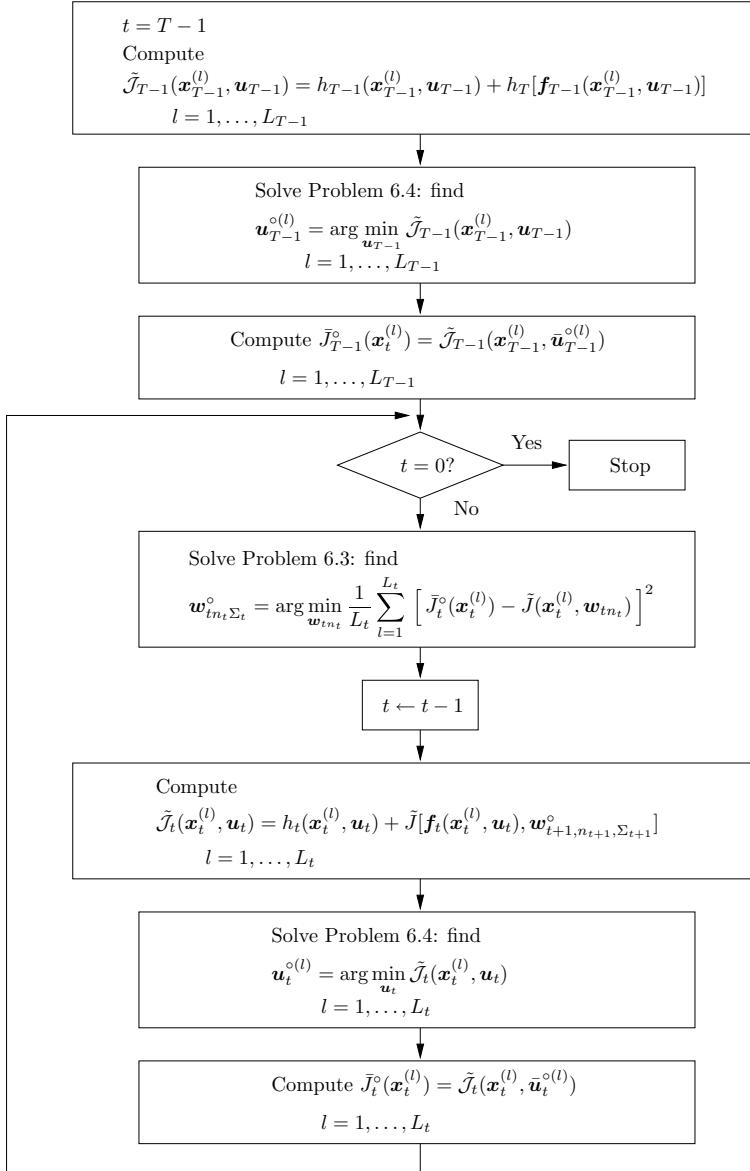


Fig. 6.5 Flowchart of the ADP computational procedure. Note that the procedure implies that Problems 6.3 and 6.4 alternate

6.4.4 The Kernel Smoothing Models in ADP

An interesting particular case is given by the kernel smoothing models (KSMs) (see Sect. 3.5) used as FSP functions. Refer to a stage $0 \leq t \leq T - 2$ and suppose that an FSP optimal cost-to-go function has been computed at the stage $t + 1$. This cost takes on the form

$$\begin{aligned} & \tilde{J}_{t+1}(\mathbf{x}_{t+1}, \alpha_{t+1}, \Sigma_{t+1}) \\ &= \frac{\sum_{l=1}^{L_{t+1}} \bar{J}_{t+1}^\circ(\mathbf{x}_{t+1}^{(l)}) \mathcal{K}_{\alpha_{t+1}}(\mathbf{x}_{t+1}, \mathbf{x}_{t+1}^{(l)})}{\sum_{l=1}^{L_{t+1}} \mathcal{K}_{\alpha_{t+1}}(\mathbf{x}_{t+1}, \mathbf{x}_{t+1}^{(l)})}, \end{aligned} \quad (6.39)$$

where the “free” parameter α_{t+1} has to be optimized in a suitable way, using the available data (for details on this issue, see the end of this section). Let α_{t+1}° be the optimal value for such a parameter. Then, the function

$$\tilde{\mathcal{J}}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t) \triangleq h_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t) + \tilde{J}_{t+1}(f_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t), \alpha_{t+1}^\circ, \Sigma_{t+1}) \quad (6.40)$$

can be computed for each $\mathbf{x}_t^{(l)} \in X'_t$ and $\mathbf{u}_t \in U_t(\mathbf{x}_t^{(l)})$ (compare with (6.34)). The minimization of (6.40) with respect to \mathbf{u}_t (i.e., the solution of the corresponding instance of Problem 6.4) gives the approximate optimal controls $\mathbf{u}_t^{(l)\circ} = \bar{\mu}_t^\circ(\mathbf{x}_t^{(l)})$ (see (6.38)), and hence the approximate optimal costs-to-go $\bar{J}_t^\circ(\mathbf{x}_t^{(l)}) = \tilde{\mathcal{J}}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t^{(l)\circ})$, $l = 1, \dots, L_t$.

At this point, one might think of computing the approximate optimal cost-to-go at stage t by solving Problem 6.3, that is, by minimizing the quadratic error

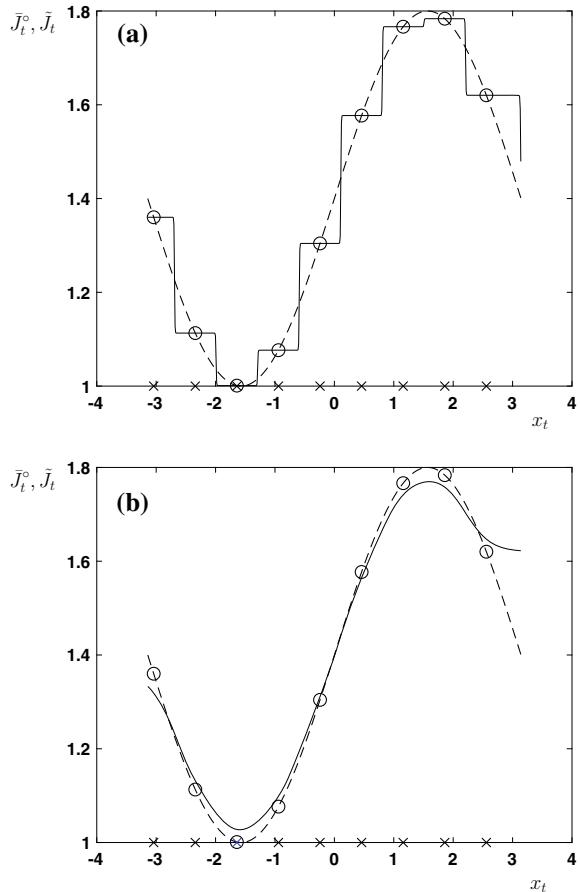
$$\frac{1}{L_t} \sum_{l=1}^{L_t} \left[\bar{J}_t^\circ(\mathbf{x}_t^{(l)}) - \tilde{J}_t(\mathbf{x}_t^{(l)}, \alpha_t, \Sigma_t) \right]^2, \quad (6.41)$$

where

$$\tilde{J}_t(\mathbf{x}_t, \alpha_t, \Sigma_t) \triangleq \frac{\sum_{l=1}^{L_t} \bar{J}_t^\circ(\mathbf{x}_t^{(l)}) \mathcal{K}_{\alpha_t}(\mathbf{x}_t, \mathbf{x}_t^{(l)})}{\sum_{l=1}^{L_t} \mathcal{K}_{\alpha_t}(\mathbf{x}_t, \mathbf{x}_t^{(l)})}, \quad (6.42)$$

with respect to the scalar α_t . In so doing, however, one would obtain a poor result. Indeed, the minimization of (6.41) would force a small value of the optimal value α_t° , since on the points $\mathbf{x}_t^{(l)}$, $l = 1, \dots, L_t$, – that is, on the points belonging to the sample set X'_t – one would obtain $\bar{J}_t^\circ(\mathbf{x}_t^{(l)}) \simeq \tilde{J}_t(\mathbf{x}_t^{(l)}, \alpha_t^\circ, \Sigma_t)$ and Quadratic Error (6.41)

Fig. 6.6 **a** The pairs of symbols (\times , \circ) correspond to the samples $(\mathbf{x}_t^{(l)}, \bar{J}_t^\circ(\mathbf{x}_t^{(l)}))$, $l = 1, \dots, 9$, and hence to the samples of the set Σ_t . We assume that the approximate optimal cost-to-go is given by $\bar{J}_t^\circ(\mathbf{x}_t) = 2 + \sin(\omega t)$ (dashed line). For very small values of α_t , KSM (6.42) (continuous line) almost interpolates the points Σ_t , but the approximation is very poor elsewhere. **b** We use the same symbols as in **a**. With a larger value of α_t the samples of Σ_t are no longer interpolated by the optimal KSM, but its approximating properties are good for any \mathbf{x}_t



would be almost zero. In other words, the cost $\tilde{J}_t(\mathbf{x}_t^{(l)}, \alpha_t^\circ, \Sigma_t)$ would “almost” interpolate the costs $\bar{J}_t^\circ(\mathbf{x}_t^{(l)})$ on the samples $\mathbf{x}_t^{(l)}$, $l = 1, \dots, L_t$, but its approximating behavior would be very poor out of these samples. To illustrate this, in the very simple example reported in Fig. 6.6a, a piecewise constant behavior is shown, which is typical for small values of α_t .

Therefore, it is necessary to get further information by constructing a second sample set $X_t'^* \triangleq \{\mathbf{x}_t^{*,(1)}, \dots, \mathbf{x}_t^{*,(L_t^*)}\}$ such that $X_t'^* \cap X_t' = \emptyset$. New approximate optimal control vectors $\mathbf{u}_t^{*,(i)\circ} = \bar{\mu}_t^\circ(\mathbf{x}_t^{*,(i)})$, $i = 1, \dots, L_t^*$, are computed, and hence new approximate optimal costs $\bar{J}_t^\circ(\mathbf{x}_t^{*,(i)})$, $i = 1, \dots, L_t^*$, are derived. In the following, the associated I/O sample set

$$\Sigma_t^* \triangleq \left\{ \left(\mathbf{x}_t^{*,(i)}, \bar{J}_t^\circ(\mathbf{x}_t^{*,(i)}) \right) : \mathbf{x}_t^{*,(i)} \in X_t'^* \right\} \quad (6.43)$$

is considered. Now, instead of the kernels in (6.42), let us choose the following ones:

$$\mathcal{K}_{\alpha_t}(\mathbf{x}_t, \mathbf{x}_t^{*,(i)}) , \quad \mathbf{x}_t^{*,(i)} \in X_t'^* .$$

Thus, define the FSP cost-to-go

$$\tilde{J}_t(\mathbf{x}_t, \alpha_t, \Sigma_t^*) \triangleq \frac{\sum_{i=1}^{L_t^*} \bar{J}_t^\circ(\mathbf{x}_t^{*,(i)}) \mathcal{K}_{\alpha_t}(\mathbf{x}_t, \mathbf{x}_t^{*,(i)})}{\sum_{i=1}^{L_t^*} \mathcal{K}_{\alpha_t}(\mathbf{x}_t, \mathbf{x}_t^{*,(i)})} \quad (6.44)$$

(compare with (6.42)) and find the optimal width α_t° by minimizing the quadratic error

$$\frac{1}{L_t} \sum_{l=1}^{L_t} \left[\bar{J}_t^\circ(\mathbf{x}_t^{(l)}) - \tilde{J}_t(\mathbf{x}_t^{(l)}, \alpha_t, \Sigma_t^*) \right]^2 . \quad (6.45)$$

The resulting FSP optimal cost-to-go $\tilde{J}_t(\mathbf{x}_t^{(l)}, \alpha_t^\circ, \Sigma_t^*)$ should now be more correct: the samples of Σ_t are no longer “almost” interpolated by $\tilde{J}_t(\mathbf{x}_t^{(l)}, \alpha_t^\circ, \Sigma_t^*)$. Again, this is illustrated by the simple example reported in Fig. 6.6b.

A robust way to implement this procedure in practice is by means of a method inspired by *cross-validation*, a technique well known in statistics and in machine learning. In brief, we divide the sample set X'_t into m subsets X'_{tm} , $m = 1, \dots, M$ of the same cardinality² $\bar{L}_t = L_t/M$ (typically, $M = 5$ or $M = 10$; see, for instance, [12]). Then, we associate the L_t samples of X'_t to the subsets X'_{tm} as follows:

$$X'_{tm} = \{\mathbf{x}_t^{(l)} : l = (m-1)\bar{L}_t + 1, \dots, m\bar{L}_t\}, \quad m = 1, \dots, M .$$

For each X'_{tm} , $m = 1, \dots, M$, the samples of all other subsets X'_{tj} , $j \neq m$ (i.e., the $L_t(M-1)/M$ samples of $X'^*_t \triangleq X'_t \setminus X'_{tm} = \cup_{j \neq m} X'_{tj}$) are used to build the quadratic error for the corresponding KSM that has to approximate \bar{J}_t° . The minimization of the average performed over M suitable errors (see (6.47)) provides the optimal width α_t° . The KSM built on the basis of the samples of the subset X'^*_t is given by (see (6.44))

²Obviously, in practice, we must ensure that \bar{L}_t is an integer. Then, we can either directly choose a value of L_t that is a multiple of M or, alternatively, allow only $M-1$ sets to be of equal size $\lfloor L_t/M \rfloor$ and take the M th set of length $L_t - (M-1)\lfloor L_t/M \rfloor$.

$$\tilde{J}_t(\mathbf{x}_t, \alpha_t, \Sigma_{tm}^*) = \frac{\sum_{i=1}^{L_t(M-1)/M} \bar{J}_t^\circ(\mathbf{x}_t^{*,(i)}) \mathcal{K}_{\alpha_t}(\mathbf{x}_t, \mathbf{x}_t^{*,(i)})}{\sum_{i=1}^{L_t(M-1)/M} \mathcal{K}_{\alpha_t}(\mathbf{x}_t, \mathbf{x}_t^{*,(i)})}, \quad (6.46)$$

where $\Sigma_{tm}^* \triangleq \left\{ \left(\mathbf{x}_t^{*,(i)}, \bar{J}_t^\circ(\mathbf{x}_t^{*,(i)}) \right) : \mathbf{x}_t^{*,(i)} \in X'_{tm} \right\}$ (see (6.29)). The quadratic error related to FSP Approximator (6.46) is expressed as

$$\frac{1}{\bar{L}_t} \sum_{l=1}^{\bar{L}_t} \left[\bar{J}_t^\circ(\mathbf{x}_t^{(l)}) - \tilde{J}_t(\mathbf{x}_t^{(l)}, \alpha_t, \Sigma_{tm}^*) \right]^2. \quad (6.47)$$

Finally, the optimal width α_t° is determined by minimizing the overall quadratic error, that is, the average of the M Errors (6.47):

$$\frac{1}{M} \sum_{m=1}^M \frac{1}{\bar{L}_t} \sum_{l=1}^{\bar{L}_t} \left[\bar{J}_t^\circ(\mathbf{x}_t^{(l)}) - \tilde{J}_t(\mathbf{x}_t^{(l)}, \alpha_t, \Sigma_{tm}^*) \right]^2.$$

Therefore, once α_t° has been computed via cross-validation, a reasonable way to determine the FSP optimal cost-to-go function consists of using the KSM computed on the basis of the whole sample set Σ_t for that choice of α_t . Let us denote this cost by $\tilde{J}_t(\mathbf{x}_t, \alpha_t^\circ, \Sigma_t)$. Indeed, in applying the cross-validation procedure, various I/O sample sets Σ_{tm}^* turn out to be generated by the same I/O sample set Σ_t . Then, it is sufficient to report in the notation only the set Σ_t .

The reason to use the whole set Σ_t for building the final approximation $\tilde{J}_t(\mathbf{x}_t, \alpha_t^\circ, \Sigma_t)$ is that we have obtained a value of α_t° that is optimized on the basis of all the available L_t points, by using them all in turn to define the various sets X'_{tm} and X'^*_{tm} . Therefore, even if the global quadratic error was formally defined on the basis of KSMs built using $L_t(M-1)/M$ points, it is reasonable to assume that the value of α_t° is also good for a KSM built using all the L_t points involved in its optimization. In this way, no information about the L_t points is lost. Obviously, this is truer as M grows, up to the extreme situation known as *leave-one-out cross-validation*, in which we put $M = L_t$ and all the points of Σ_t are used, in turn, to form singleton sets X'_{tm} . Since this leave-one-out procedure greatly increases the computational burden (unless linear models are employed, for which an efficient computation is possible), a value of $M = 10$ is generally considered a reasonable compromise. Another motivation to define $\tilde{J}_t(\mathbf{x}_t, \alpha_t^\circ, \Sigma_t)$ is that, in this way, a single approximation is used instead of M approximations $\tilde{J}_t(\mathbf{x}_t, \alpha_t^\circ, \Sigma_{tm}^*)$.

To sum it up, at each stage $t = 0, \dots, T-2$, an instance of Problem 6.4 must be solved by minimizing the following cost-to-go function with respect to \mathbf{u}_t :

$$\tilde{J}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t) \triangleq h_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t) + \tilde{J}_{t+1}[\mathbf{f}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t), \alpha_{t+1}^\circ, \Sigma_{t+1}], \quad \forall \mathbf{x}_t^{(l)} \in X'_t.$$

It is worth noting that, in the computation of the FSP optimal cost-to-go function $\tilde{J}_t(\mathbf{x}_t, \alpha_t^\circ, \Sigma_t)$, there is seemingly no danger of incurring the curse of dimensionality as regards the number of parameters to be optimized (compare with the costs $\tilde{J}(\mathbf{x}_t^{(l)}, \mathbf{w}_{tn_l})$ in Problem 6.3) since only the scalar α_t has to be optimized. However, one must take care of the computational burden for calculating $\tilde{J}_t(\mathbf{x}_t, \alpha_t^\circ, \Sigma_t)$ when the state dimension d increases. More details on the efficient use of KSMs in the ADP context can be found in [7, 8].

This concludes the backward phase of ADP. In the next section, we shall address the forward one.

6.5 Computation of the Optimal Controls in the Forward Phase of ADP

Once the backward phase of the ADP procedure has been completed (i.e., Equation (6.31) has been solved), the optimal control functions (6.38) are available. At the initial stage $t = 0$, as soon as $\mathbf{x}_0 \in X_0$ becomes known to the DM, the optimal control $\mathbf{u}_0^\circ = \bar{\mu}_0^\circ(\mathbf{x}_0)$ can be decided (we continue considering Problem C1'). However, as pointed out in the backward phase of ADP, the function $\bar{\mu}_0^\circ$ is known only in the discretized states $\mathbf{x}_0^{(l)} \in X'_0$, and, in general, \mathbf{x}_0 does not coincide with any state belonging to X'_0 . The same occurs for the stages $t = 1, \dots, T - 1$. Again, we may take the vector associated with the discretized state $\mathbf{x}_t^{(l^*)} \in X'_t$ that is nearest to \mathbf{x}_t as optimal control \mathbf{u}_t° . Nevertheless, if a more precise computation is needed (and, from now on, we shall assume that this is the case), some kinds of approximate procedures are feasible, namely, (i) the use of the FSP functions $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t, \Sigma_t}^\circ)$ along with the *reoptimization* procedure and (ii) the application of a second family of FSP functions to approximate the optimal control functions using the least-squares criterion (for simplicity, we continue to use this criterion). We address these procedures in what follows:

1. Reoptimization based on the FSP optimal costs-to-go $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t, \Sigma_t}^\circ)$

In the backward phase of ADP, we have computed the FSP optimal costs-to-go $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t, \Sigma_t}^\circ)$ by solving Problem 6.3. Then, at stage $t = 0$, as soon as the DM knows the state \mathbf{x}_0 , it determines the optimal control vector \mathbf{u}_0° by the following minimization:

$$\mathbf{u}_0^\circ = \underset{\substack{\mathbf{u}_0 \in U_0(\mathbf{x}_0^\circ) \\ f_0(\mathbf{x}_0, \mathbf{u}_0) \in X_1}}{\operatorname{argmin}} \left\{ h_0(\mathbf{x}_0, \mathbf{u}_0) + \tilde{J} \left[f_0(\mathbf{x}_0, \mathbf{u}_0), \mathbf{w}_{tn_1, \Sigma_1}^\circ \right] \right\}. \quad (6.48)$$

This minimization is repeated, stage after stage, at the states $\mathbf{x}_{t+1}^\circ = f_t(\mathbf{x}_t^\circ, \mathbf{u}_t^\circ)$ for $t = 1, \dots, T - 1$. Then, the general formula to obtain the optimal control vectors $\mathbf{u}_0^\circ, \mathbf{u}_1^\circ, \dots, \mathbf{u}_{T-1}^\circ$ is given by

$$\begin{aligned} \mathbf{u}_t^\circ = \underset{\substack{\mathbf{u}_t \in U_t(\mathbf{x}_t^\circ) \\ f_t(\mathbf{x}_t^\circ, \mathbf{u}_t) \in X_{t+1}'}}{\operatorname{argmin}} & \left\{ h_t(\mathbf{x}_t^\circ, \mathbf{u}_t) + \tilde{J}[\mathbf{f}_t(\mathbf{x}_t^\circ, \mathbf{u}_t), \mathbf{w}_{t+1, n_{t+1}, \Sigma_{t+1}}^\circ] \right\}, \\ t = 0, 1, \dots, T-1. \end{aligned} \quad (6.49)$$

The double minimization, i.e., in the backward (Problems 6.4) and the forward phases (see (6.49)) justifies the term “reoptimization”. This approach deserves some comments. Indeed, in the forward phase, we have to perform a sequence of T minimization operations. This may entail a certain computational burden but this is nothing to worry about as Problems (6.49) are much simpler than the determination of the FSP optimal costs-to-go in the backward phase of ADP. For the case of Problem C1', they can be solved once the initial state \mathbf{x}_0 becomes known. In the particular case of Problem C1, they can be solved offline.

Of course, if the computational time is sufficiently small with respect to the interval $(t, t+1)$, then it is possible to perform the minimizations (6.49) *online* at each stage t . The online procedure may involve an important advantage with respect to the offline one. If the DM realizes that the state is out of the optimal trajectory, it can generate a correcting optimal control, thus implementing a *feedback mechanism*. Clearly, all this means that the deterministic framework – on the basis of which Problem C1' had been stated – was too optimistic.

2. Reoptimization by a second family of FSP functions

During the backward phase of ADP or even after it has been completed, the optimal closed-loop control functions $\bar{\mu}_t^\circ(\mathbf{x}_t)$ can be approximated by another family of FSP functions. Let us denote these functions by

$$\mathbf{u}_t = \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t}^u), \quad t = 0, 1, \dots, T-1, \quad (6.50)$$

and define the I/O sample sets similar to (6.29), i.e.,

$$\Sigma_t^u \triangleq \left\{ (\mathbf{x}_t^{(l)}, \bar{\mu}_t^\circ(\mathbf{x}_t^{(l)})) : \mathbf{x}_t^{(l)} \in X_t' \right\}, \quad t = 0, 1, \dots, T-1. \quad (6.51)$$

(It might occur that samples $\mathbf{x}_t^{(l)}$ are needed other than the ones pertaining to X_t' in order to improve the new approximation.) Obviously, the integers n_t in (6.50) are generally different from the ones appearing in the costs $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t})$. As we did for these costs, we assume that the FSP functions (6.50) remain of the same type during all stages.

As has been done for the approximation of the optimal cost-to-go functions $\tilde{J}_t^\circ(\mathbf{x}_t)$ by the FSP optimal cost-to-go functions $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t, \Sigma_t}^\circ)$ (see Problem 6.3), once the vectors $\mathbf{u}_t^{\circ(l)}$ have been computed for each discretized state $\mathbf{x}_t^{(l)} \in X_t'$, the vectors $\mathbf{w}_{tn_t}^u$ can be optimized by using (for example) the least-squares criterion. The parameter vectors $\mathbf{w}_{tn_t, \Sigma_t^u}^u$ are then optimized by solving the following NLP problems.

Problem 6.5 For all stages $t = 0, 1, \dots, T - 1$, find the optimal parameter vector $\mathbf{w}_{tn_t, \Sigma_t^u}^{u^\circ}$ that minimizes the quadratic cost

$$\frac{1}{L_t} \sum_{l=1}^{L_t} \left| \bar{\mu}_t^\circ(\mathbf{x}_t^{(l)}) - \tilde{\mu}(\mathbf{x}_t^{(l)}, \mathbf{w}_{tn_t, \Sigma_t^u}^{u^\circ}) \right|^2. \quad (6.52)$$

△

The solutions of Problems 6.5 provide the FSP optimal control functions of the form (6.50). These functions give the approximate optimal controls for all the states $\mathbf{x}_t \in \bar{X}_t$. The control vectors $\mathbf{u}_0^\circ, \mathbf{u}_1^\circ, \dots, \mathbf{u}_{T-1}^\circ$ can then be determined offline by (6.50) and the state equation. Moreover, Functions (6.50), once optimized, allow the DM to generate the approximate optimal controls if it realizes to be out of the optimal trajectory.

6.6 Generalization Error Owing to the Use of Sampling Procedures and FSP Functions

The use of the FSP functions for approximating the (approximate) optimal cost-to-go and control functions and the use of the sampling techniques for generating the sets X'_t constitute two fundamental issues for achieving a satisfactory compromise between accuracy and the danger that the memory and computational burden gives rise to two kinds of curse of dimensionality.

The choice of the type of FSP functions is the first step in the approximation of the (approximate) optimal cost-to-go functions (see Problem 6.3) and of the (approximate) optimal control functions if Problem 6.5 have to be solved. As stated previously, an effective technique for applying ADP should require a limited number of samples. We shall now discuss the consequences of the results presented in Chap. 4 about the sampling of the sets \bar{X}_t and the use of the FSP functions when ADP is applied in high-dimensional settings. The purpose is to get insights into the possibility of bounding the errors incurred by approximating the cost-to-go and control functions.

More specifically, we consider the approximation of the cost-to-go functions $\bar{J}_t^\circ(\mathbf{x}_t)$ by the FSP functions $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t, \Sigma_t}^{u^\circ})$. Note that we consider here the errors of the costs $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t, \Sigma_t}^{u^\circ})$ with respect to the approximate optimal costs \bar{J}_t° and not with respect to the “true” optimal costs J_t° . Of course, similar problems should be faced if one has to derive the FSP functions $\mathbf{u}_t^\circ = \tilde{\mu}(\mathbf{x}_t, \tilde{\mathbf{w}}_{tn_t}^{u^\circ})$, which approximate the control functions $\mathbf{u}_t^\circ = \bar{\mu}_t^\circ(\mathbf{x}_t)$.

It is useful to point out the following correspondences between Problems 6.3 and 4.3. They hold for $t = 0, 1, \dots, T - 1$.

- Quadratic Cost (6.32) corresponds to Empirical Risk (4.18).
- The number of samples L_t corresponds to L .

- The state vectors $\mathbf{x}_t \in \bar{X}_t$ correspond to the independent or input variable $\mathbf{x} \in X$. The discretized states $\mathbf{x}_t^{(l)} \in X'_t$ correspond to the discretized vectors $\mathbf{x}^{(l)} \in X_L$, where X_L is given by (4.1).
- The I/O samples $(\mathbf{x}_t^{(l)}, \tilde{J}_t^\circ(\mathbf{x}_t^{(l)}))$ correspond to the I/O samples $(\mathbf{x}^{(l)}, y^{(l)})$. These pairs are the elements of the sample sets Σ_t (see (6.25), (6.29)), which correspond to the I/O sample set Σ_L defined by (4.3).
- The values of the cost-to-go functions $\tilde{J}_t^\circ(\mathbf{x}_t^{(l)})$, computed via ADP Recursive Equation (6.31), correspond to the response or output values $y^{(l)}$.
- The FSP function $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t, \Sigma_t}^\circ) \in \mathcal{A}_{tn_t}$, obtained by solving Problems 6.3, corresponds to the FSP optimal function $\mathbf{y}(\mathbf{x}, \mathbf{w}_{n \Sigma_L}^\circ) \in \mathcal{A}_n$ that solves Problem 4.3.

We now introduce, at any decision stage, the *generalization error*, in order to evaluate the errors due to both the limited model complexity n_t of the FSP functions $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t, \Sigma_t}^\circ)$ and the limited sample cardinality L_t of the set Σ_t . We assume the errors related to the solutions of the various NLP problems to be negligible. So, we neglect the errors deriving from solving approximately Problems 6.3 (and also 6.4). Then, we are in the framework of the estimation of functions without noise addressed in Sect. 4.4.1. Of course, the inputs $\mathbf{x}^{(l)}$ can be generated in a completely controllable way, and the I/O relationship is deterministic. Thus, we are in the deterministic learning theory (DLT) framework (see Fig. 4.2). On the basis of the correspondences previously emphasized, the generalization errors (4.23), which are simply given by (4.86), can be written as follows:

$$\begin{aligned} \mathcal{E}_{tn_t, \Sigma_t}^{\text{gen}} &= \int_{\bar{X}_t} \left[\tilde{J}_t^\circ(\mathbf{x}_t) - \tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t, \Sigma_t}^\circ) \right]^2 d\mathbf{x}_t = R(\tilde{J}_{tn_t, \Sigma_t}^\circ), \\ t &= 0, 1, \dots, T-1. \end{aligned} \quad (6.53)$$

Moreover, we define the *approximation errors* as in (4.87), that is,

$$\begin{aligned} \mathcal{E}_{tn_t}^{\text{appr}} &= \int_{\bar{X}_t} \left[\tilde{J}_t^\circ(\mathbf{x}_t) - \tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t}^\circ) \right]^2 d\mathbf{x}_t = R(\tilde{J}_{tn_t}^\circ), \\ t &= 0, 1, \dots, T-1. \end{aligned} \quad (6.54)$$

Errors (6.54) are those of the two terms that contribute to form the generalization errors (6.53). The other terms are the *estimation errors*. As we know, $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t}^\circ)$ are the FSP optimal functions that would approximate $\tilde{J}_t^\circ(\mathbf{x}_t)$ if, ideally, we had at our disposal an infinite number of samples. In practice, this number is finite. Then, on the basis of (4.88), the estimation errors are given by

$$\mathcal{E}_{tn_t, \Sigma_t}^{\text{est}} = \left| R_{\text{emp}}(\tilde{J}_{tn_t, \Sigma_t}^\circ) - R(\tilde{J}_{tn_t, \Sigma_t}^\circ) \right|, \quad t = 0, 1, \dots, T-1. \quad (6.55)$$

Considerations hold for the FSP optimal control functions (6.50) that are similar to the ones made about the FSP optimal cost-to-go functions. We also define sample sets for the control vectors by recalling that, for each stage t and for each discretized

state $\mathbf{x}_t^{(l)} \in X'_t$, a control vector \mathbf{u}_t° has been determined (see (6.38)). Then, we have the *generalization error*

$$\mathcal{E}_{tn_t, \Sigma_t}^{u, \text{gen}} = \int_{\bar{X}_t} \left| \bar{\mu}_t^\circ(\mathbf{x}_t) - \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t, \Sigma_t^u}^{u\circ}) \right|^2 d\mathbf{x}_t = \left\| \bar{\mu}_t^\circ - \tilde{\mu}_{tn_t, \Sigma_t^u}^{u\circ} \right\|_{\mathcal{L}_2(\bar{X}_t)}^2, \\ t = 0, 1, \dots, T-1, \quad (6.56)$$

where we have used the usual conventional notation $\tilde{\mu}_{tn_t, \Sigma_t}^{u\circ} = \tilde{\mu}(\cdot, \mathbf{w}_{tn_t, \Sigma_t}^{u\circ})$. The functions $\tilde{\mu}_{tn_t}^{u\circ}$ are the FSP optimal functions that would approximate $\bar{\mu}_t^\circ$ if an infinite number of samples were available. In such an ideal case, $\mathbf{w}_{tn_t}^{u\circ}$ would be given by

$$\mathbf{w}_{tn_t}^{u\circ} = \operatorname{argmin}_{\mathbf{w}_{tn_t}^u} \int_{\bar{X}_t} \left| \bar{\mu}_t^\circ(\mathbf{x}_t) - \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t}^u) \right|^2 d\mathbf{x}_t, \quad t = 0, 1, \dots, T-1$$

and for them an approximation error can be defined as in (6.54).

Again, we can define an estimation error owing to the limited number of vectors $\mathbf{x}_t^{(l)}$ where we compute the optimal control vectors.

6.6.1 Approximation Errors and Approximating FSP Functions

This section is strictly related to the definitions and properties of \mathcal{M} -approximating sequences of sets stated in Sect. 2.6 (see Definition 2.6) and to the definitions and properties of polynomially complex \mathcal{M} -approximating sequences of sets stated in Sect. 2.7 (see Definition 2.8). As we said just after giving these definitions, for simplicity we shall use the terms *\mathcal{M} -* and *pc \mathcal{M} -approximating FSP functions* to denote the FSP functions that belong to the sets of the abovementioned sequences. In order to establish a clear connection between Sects. 2.6, 2.7 and this section, we recall the following concepts.

In Sect. 2.6, we have considered two cases where functions have to be approximated. In Case (i), we have to ascertain the possibility of approximating admissible decision functions of Problem P^d to any degree of accuracy, that is, the functions belonging to the set (see (1.4) and (2.60))

$$\mathcal{S}^d = \{\gamma^d: D \rightarrow C\}.$$

In Case (ii), the functions γ^d belong to sets (see 2.61))

$$\mathcal{M}^d \subseteq \mathcal{G}^d. \quad (6.57)$$

We recall that the sets \mathcal{M}^d and the ambient spaces \mathcal{G}^d have, in general, nothing to do with the IDO Problem P^d . This means that we can focus on Case (ii) as we just

want to address the approximation problems faced in this section. More specifically, we investigate the capability of the FSP functions to approximate any “candidate” optimal cost-to-go and control functions (\tilde{J}_t and $\tilde{\mu}_t$, respectively).

In particular, two questions have to be answered. The first question is the following: given a properly defined degree of accuracy, do FSP functions \tilde{J}_{tn_t} and $\tilde{\mu}_{tn_t}^u$ ($t = 0, 1, \dots, T - 1$) exist that are able to approximate the functions \tilde{J}_t and $\tilde{\mu}_t$ (hence, obviously, the optimal functions \tilde{J}_t° and $\tilde{\mu}_t^\circ$) to this degree? Once it has been verified that these FSP functions exist, the second question is the following: can they achieve a desired degree of accuracy having model complexities that increase at most as powers of the dimension $d = \dim(\mathbf{x}_t)$? The concepts and the results presented in Sects. 2.6, 2.7, and Chap. 3 allow us to answer both questions.

As regards the first question, the issue is closely related to the existence of families of FSP functions that enjoy *density properties* in the sets of functions where one searches the optimal cost-to-go and control functions, thus becoming *approximating FSP functions* for such sets.

With this in mind, it is useful to assume that the candidate functions \tilde{J}_t and $\tilde{\mu}_t$ have suitable regularity properties. Let us focus on the functions \tilde{J}_t ; similar arguments hold for the functions $\tilde{\mu}_t$. Consistently with the examples presented at the end of Sect. 2.6, the following hypotheses play an important role.

Assumption 6.1 For $t = 0, 1, \dots, T - 1$, (i) the admissible state domains \bar{X}_t are compact sets and (ii) the sets \mathcal{M}_t^d of the “candidate” optimal cost-to-go functions are subsets of linear spaces equipped with the \mathcal{L}_2 -norm (or, more in general, with the \mathcal{L}_p -norm, $p \in [1, \infty)$) or of spaces of continuous functions equipped with the supremum norm. \triangleleft

In other words, we suppose that

$$\tilde{J}_t \in \mathcal{M}_{\mathcal{L}_2 t}^d \subseteq \mathcal{L}_2(\bar{X}_t, \mathbb{R}), \quad t = 0, 1, \dots, T - 1, \quad (6.58)$$

(see (2.64)) or

$$\tilde{J}_t \in \mathcal{M}_{\mathcal{C}_t}^d \subseteq \mathcal{C}(\bar{X}_t, \mathbb{R}), \quad t = 0, 1, \dots, T - 1,$$

(see (2.63)). From (6.23) and (6.28), we define the following families of FSP functions (see (2.2)):

$$\begin{aligned} \mathcal{A}_{tn_t} &\triangleq \left\{ \tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t}) : \mathbf{x} \in \mathbb{R}^d, \mathbf{w}_{tn_t} \in \mathbb{R}^{\mathcal{N}_t(n_t)} \right\}, \\ t &= 0, 1, \dots, T - 1, n_t = 1, 2, \dots, \end{aligned}$$

where $\mathcal{N}_t(n_t)$ is the number of parameters of the network \tilde{J}_{tn_t} , i.e., $\mathcal{N}_t(n_t) = \dim(\mathbf{w}_{tn_t})$ (see (2.1)).

If Assumption 6.1 is verified, then we know from Chap. 3 that, for $t = 0, 1, \dots, T - 1$, there exist sequences of sets $\{\mathcal{A}_{tn_t}\}_{n_t=1}^\infty$ that are dense in $\mathcal{M}_{\mathcal{L}_2 t}^d$ and $\mathcal{M}_{\mathcal{C}_t}^d$. This

means that, given any cost-to-go function \bar{J}_t belonging to $\mathcal{M}_{\mathcal{L}_2 t}^d$ or $\mathcal{M}_{\mathcal{C} t}^d$, there exists an $\mathcal{M}_{\mathcal{L}_2 t}^d$ - or an $\mathcal{M}_{\mathcal{C} t}^d$ -approximating FSP function, respectively, that is arbitrarily close to this function in the respective norm. More specifically, for any $\varepsilon > 0$, there exist \bar{n}_t , $\bar{n}'_t \in \mathbb{Z}^+$ and vectors $\bar{\mathbf{w}}_{tn_t}$, $\bar{\mathbf{w}}'_{tn'_t}$ such that

$$\left\| \bar{J}_t - \tilde{J}(\cdot, \bar{\mathbf{w}}_{tn_t}) \right\|_{\mathcal{L}_2} \leq \varepsilon \quad (6.59)$$

and

$$\left\| \bar{J}_t - \tilde{J}(\cdot, \bar{\mathbf{w}}'_{tn'_t}) \right\|_{\mathcal{C}} \leq \varepsilon, \quad (6.60)$$

for any $n_t \geq \bar{n}_t$ and any $n'_t \geq \bar{n}'_t$, respectively.

Consider the optimal $\mathcal{M}_{\mathcal{L}_2 t}^d$ -approximating FSP functions $\tilde{J}_{tn_t}^\circ$, that minimize the \mathcal{L}_2 -norm in (6.59) for $\bar{J}_t = \tilde{J}_t^\circ$ and $n_t \geq \bar{n}_t$. Then, from (6.59) one obtains

$$\mathcal{E}_{tn_t, \mathcal{L}_2}^{\text{appr}} \triangleq \left\| \bar{J}_t^\circ - \tilde{J}_{tn_t}^\circ \right\|_{\mathcal{L}_2} \leq \varepsilon, \quad \forall n_t \geq \bar{n}_t, \quad t = 0, 1, \dots, T-1. \quad (6.61)$$

Consider now the supremum norm and define the approximation error

$$\mathcal{E}_{tn_t, \mathcal{C}}^{\text{appr}} \triangleq \left\| \bar{J}_t^\circ - \tilde{J}(\cdot, \bar{\mathbf{w}}'_{tn_t}) \right\|_{\mathcal{C}}, \quad t = 0, 1, \dots, T-1, \quad (6.62)$$

where the meaning of $\bar{\mathbf{w}}'_{tn_t}$ is similar to that of $\mathbf{w}_{tn_t}^\circ$. Then, following the same reasoning we used for (6.61), for $\bar{J}_t = \tilde{J}_t^\circ$ and $n_t \geq \bar{n}'_t$ in (6.60), we obtain

$$\mathcal{E}_{tn_t, \mathcal{C}}^{\text{appr}} = \left\| \bar{J}_t^\circ - \tilde{J}_{tn'_t}^\circ \right\|_{\mathcal{C}} \leq \varepsilon, \quad \forall n'_t \geq \bar{n}'_t, \quad t = 0, 1, \dots, T-1. \quad (6.63)$$

As regards the choice of the criterion according to which we approximate the costs \bar{J}_t° , we have already argued that the minimax criterion is clearly preferable, since it gives bounds on $|\bar{J}_t^\circ(\mathbf{x}_t) - \tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t})|$ for any $\mathbf{x}_t \in \bar{X}_t$. It is also worth noting that similar results for function approximation have been obtained in the literature for both \mathcal{L}_2 - and sup-norm. However, we conclude this section considering the least-squares criterion we have addressed in Chap. 4.

As regards the second question we posed at the beginning of this section, let us consider the possibility of mitigating the curse of dimensionality related to an excessive growth of the model complexity n_t of the FSP functions \tilde{J}_{tn_t} , for a fixed upper bound $\varepsilon > 0$ on the approximation error $\mathcal{E}_{tn_t}^{\text{appr}}$, when the state dimension d increases. To this end, we address Sets (6.58).

We recall from Chap. 3 that, for several sets $\mathcal{M}_{\mathcal{L}_2 t}^d$ provided with suitable regularity properties, there exist constants $p_t, q_t, c_t \in \mathbb{R}$, $p_t \geq 0, q_t, c_t > 0$ such that the approximation errors can be bounded as follows:

$$\mathcal{E}_{tn_t \mathcal{L}_2}^{\text{appr}} = \left\| \bar{J}_t^\circ - \tilde{J}_{tn_t}^\circ \right\|_{\mathcal{L}_2}^2 \leq c_t \frac{d^{p_t}}{n_t^{q_t}}, \quad t = 0, 1, \dots, T-1. \quad (6.64)$$

Then, for any $\varepsilon > 0$, it is possible to determine large enough model complexities n_t such that (6.64) becomes

$$\mathcal{E}_{tn_t \mathcal{L}_2}^{\text{appr}} \leq c_t \frac{d^{p_t}}{n_t^{q_t}} \leq \varepsilon, \quad t = 0, 1, \dots, T-1. \quad (6.65)$$

Of course, it must be

$$n_t \geq \left\lceil \left(\frac{c_t}{\varepsilon} \right)^{1/q_t} d^{p_t/q_t} \right\rceil, \quad t = 0, 1, \dots, T-1. \quad (6.66)$$

(Compare (6.65) with (2.71) and (6.66) with (2.73).)

It has to be remarked that, to estimate the propagation of the error through the stages, one has to quantify the approximation accuracy of the cost-to-go functions in the supremum norm. In fact, typically the \mathcal{L}_2 -norm used in Equation (6.64) does not allow one to control the backward propagation of the error through the stages. For recent results in this direction, which include also the stochastic case, we refer the reader to [10, 11].

As regards the approximation errors related to the computation of the “true” optimal controls \mathbf{u}_t° by FSP Functions (6.50), we point out that the control functions $\bar{\mu}_t^\circ$ are generally vector valued, whereas the functions \bar{J}_t° take on scalar values. This is not a drawback. Indeed, one may address the parallel of single-output FSP functions

$$u_t^{(j)} \triangleq \bar{\mu}_t^{(j)}(\mathbf{x}_t, \mathbf{w}_{tn_j}^u), \quad t = 0, 1, \dots, T-1, \quad j = 1, \dots, m. \quad (6.67)$$

It follows that the reasoning on the approximation error previously reported for the cost-to-go functions \bar{J}_t can be repeated for each component of the control functions $\bar{\mu}_t$ without any conceptual difference.

6.6.2 Estimation Errors in the Context of Deterministic Learning Theory

In order to present bounds on the estimation errors $\mathcal{E}_{tn_t \Sigma_t}^{\text{est}}$ (see (6.55)), we have to address the discretized sets X'_t . As pointed out in Chap. 4, in DLT these sets are generated by deterministic algorithms. This choice is justified for both practical and theoretical reasons. From a practical point of view, low-discrepancy sequences cope with the problem of how to sample the state space in an uniform way by a scheme that may not depend structurally on the input dimension d . In other words, sequences of any desired length L may be generated for any state dimension d , in such a way that all the areas of the input space are sampled evenly.

From a theoretical point of view, relying on the properties described in Sect. 4.4, low-discrepancy sequences may be able to exploit possible regularities of the cost-to-go functions leading to efficient error rates. In particular, under suitable assumptions on the regularity of the cost-to-go functions and on the properties of the FSP functions used to solve Problem C1', the rates of convergence of the estimation errors of the FSP optimal costs-to-go $\tilde{J}(\cdot, \mathbf{w}_{tn_t, \Sigma_t}^\circ)$ are almost linear in the sample cardinalities. This may enable us to mitigate the curse of dimensionality in applying the ADP related to the sampling of the sets \bar{X}_t by regular grids.

At any stage $t = 0, 1, \dots, T - 1$, the deterministic algorithms generate a sequence of vectors $\mathbf{x}_t^{(1)}, \dots, \mathbf{x}_t^{(L_t)}$, giving rise to the set X'_t , and hence to the set Σ_t . Now, we want to examine the behavior of $\mathcal{E}_{tn_t, \Sigma_t}^{\text{est}}$ when the sample cardinality L_t increases in such sets.

For the sake of simplicity, in order to extend the results on variations of functions and discrepancies of sequences (see also Sect. 4.4) from the set $[0, 1]^d$ to the sets \bar{X}_t , we assume that each set \bar{X}_t , $t = 0, 1, \dots, T - 1$, is a unit hypercube $[0, 1]^d$ or a set that can be brought back to the unit hypercube by means of scaling (e.g., a bounded non-unit hypercube) or by simple transformations, e.g., a simplex (as explained in Sect. 6.3, we have a certain “freedom” in building the sets \bar{X}_t , $t = 0, 1, \dots, T - 1$). In order to derive an upper bound on the convergence rate of the estimation error $\mathcal{E}_{tn_t, \Sigma_t}^{\text{est}}$ as L_t increases (we should write Σ_{tL_t} , but see Remark on Notation 6.3), we have to make sure a priori that the functions \tilde{J}_t° and \tilde{J}_{tn_t} belong to suitable spaces (see (4.99)). So, we have bounded variations in the sense of Hardy and Krause. Toward this end, we formulate the following assumption.

Assumption 6.2 (i) For any \mathbf{w}_{tn_t} and $t = 0, 1, \dots, T - 1$, there exist finite scalars M_t^J such that $\tilde{J}_{tn_t} \in \mathcal{W}_{M_t^J}([0, 1]^d)$, hence, the FSP cost-to-go functions $\tilde{J}(\cdot, \mathbf{w}_{tn_t})$ have bounded variations in the sense of Hardy and Krause.

(ii) For $t = 0, 1, \dots, T - 1$, there exist finite scalars $M_t^{\bar{J}}$ such that $\tilde{J}_t^\circ \in \mathcal{W}_{M_t^{\bar{J}}}([0, 1]^d)$, hence, the approximate optimal cost-to-go functions \tilde{J}_t° have bounded variations in the sense of Hardy and Krause. \triangleleft

Note that Point (i) of the above assumption is easily verified as we can choose suitable FSP functions \tilde{J}_{tn_t} . In most cases, they are given by sigmoidal OHL networks (see (4.100)), radial OHL networks (see (4.101)), and other OHL networks. Point (ii) of the assumption formalizes regularities of the optimal costs-to-go \tilde{J}_t° that can be exploited when using low-discrepancy sampling.

Now, following the same procedure that led Inequality (4.102) to take on the form of Inequality (4.105), let us consider a quadratic loss function (see (4.38) and (4.41) for $p = 2$)

$$v[\tilde{J}_t^\circ(\mathbf{x}_t), \tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t})] = [\tilde{J}_t^\circ(\mathbf{x}_t) - \tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t})]^2, \quad t = 0, 1, \dots, T - 1. \quad (6.68)$$

Then, from KH Inequality (4.102) we obtain

$$\begin{aligned} & \left| \frac{1}{L_t} \sum_{l=1}^{L_t} [\bar{J}_t^\circ(\mathbf{x}_t^{(l)}) - \tilde{J}(\mathbf{x}_t^{(l)}, \mathbf{w}_{tn_t})]^2 - \int_{[0,1]^d} [\bar{J}_t^\circ(\mathbf{x}_t) - \tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t})]^2 d\mathbf{x}_t \right| \\ & \leq V_{HK} \left\{ [\bar{J}_t^\circ(\mathbf{x}_t) - \tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t})]^2 \right\} D_{L_t}^*(\{\mathbf{x}^{L_t}\}), \\ & t = 0, 1, \dots, T-1. \end{aligned} \quad (6.69)$$

We recall that $D_{L_t}^*(\{\mathbf{x}^{L_t}\})$ denotes the star discrepancy of each sample sequence $\{\mathbf{x}^{L_t}\}$, whose elements belong to $[0, 1]^d$.

Taking into account (4.90) and (4.91) and the form (6.68) of the loss function, (6.69) becomes

$$|R_{\text{emp}}(\mathbf{w}_{tn_t}, \Sigma_t) - R(\mathbf{w}_{tn_t})| \leq V'_{HK}(\mathbf{w}_{tn_t}) D_{L_t}^*(\{\mathbf{x}^{L_t}\}), \quad t = 0, 1, \dots, T-1, \quad (6.70)$$

where

$$V'_{HK}(\mathbf{w}_{tn_t}) \triangleq V_{HK} \left\{ [\bar{J}_t^\circ - \tilde{J}(\cdot, \mathbf{w}_{tn_t})]^2 \right\}$$

(see (4.104)). If Assumption 6.2 is verified, then the two functions \bar{J}_t° and $\tilde{J}(\cdot, \mathbf{w}_{tn_t})$ have bounded variations in the sense of Hardy and Krause. Then, thanks to Corollary 4.1, the square of their difference also has a bounded variation in the same sense and $V'_{HK}(\mathbf{w}_{tn_t})$ takes on finite values for any \mathbf{w}_{tn_t} . Assuming in addition that they are uniformly bounded with respect to the choice of \mathbf{w}_{tn_t} , we can define the following constants (see (4.111)):

$$\bar{V}_{tn_t} \triangleq \sup_{\mathbf{w}_{tn_t}} V'_{HK}(\mathbf{w}_{tn_t}) < \infty, \quad t = 0, 1, \dots, T-1. \quad (6.71)$$

From (6.70) and (6.71), we obtain the inequalities

$$|R_{\text{emp}}(\mathbf{w}_{tn_t}, \Sigma_t) - R(\mathbf{w}_{tn_t})| \leq \bar{V}_{tn_t} D_{L_t}^*(\{\mathbf{x}^{L_t}\}), \quad t = 0, 1, \dots, T-1. \quad (6.72)$$

Letting $\mathbf{w}_{tn_t}^\circ = \mathbf{w}_{tn_t}$, Inequalities (6.72) give upper bounds on Estimation Errors (4.28). We have

$$\mathcal{E}_{n_t \Sigma_t}^{\text{est}} \leq \bar{V}_{tn_t} D_{L_t}^*(\{\mathbf{x}^{L_t}\}), \quad t = 0, 1, \dots, T-1. \quad (6.73)$$

Let us now consider the convergence of the risk $R(\gamma_{n \Sigma_L}^\circ)$ to the risk $R(\gamma_n^\circ)$ (see (4.22)). By using Proposition 4.1, from (6.72) we obtain (see (4.3.1))

$$R(\mathbf{w}_{tn_t \Sigma_t}^\circ) - R(\mathbf{w}_{tn_t}^\circ) \leq 2 \bar{V}_{tn_t} D_{L_t}^*(\{\mathbf{x}^{L_t}\}). \quad (6.74)$$

Taking into account (6.72) and (6.74), we summarize the results given in [9] as follows.

Theorem 6.1 *If Assumption 6.2 is verified and (6.71) holds, then*

$$R(\tilde{J}_{tn_t, \Sigma_t}^\circ) - R(\tilde{J}_{tn_t}^\circ) = R(\mathbf{w}_{tn_t, \Sigma_t}^\circ) - R(\mathbf{w}_{tn_t}^\circ) \leq 2\bar{V}_{tn_t} D_{L_t}^*(\{\mathbf{x}^{L_t}\}), \\ t = 0, 1, \dots, T-1. \quad (6.75)$$

□

We can conclude that, for given values of \bar{V}_{tn_t} , the rates of convergence of $R(\tilde{J}_{tn_t, \Sigma_t}^\circ)$ to $R(\tilde{J}_{tn_t}^\circ)$ when $L_t \rightarrow \infty$ follow the rates of the discrepancies of the sequences $\{\mathbf{x}^{L_t}\}$. In case of low-discrepancy sequences, these rates are $O(\ln(L_t)^{d-1}/L_t)$ (compare with Theorem 4.10).

However, it is important to note that the boundedness of the variations V_{HK} (see (6.69)) does not prevent the constants \bar{V}_{tn_t} from growing fast with the dimension d of the state vector. On the other hand, once we have fixed the dimension d , Theorem 6.1 ensures that, if the cost-to-go functions are sufficiently well-behaved and the FSP functions are properly chosen, low-discrepancy sequences allow one to obtain a favorable rate of approximation in terms of the sample cardinalities L_t .

6.7 Reduction of Problem C1 to an NLP Problem

In the previous sections, we have seen that DP provides an important “by-product”, i.e., the FSP closed-loop optimal control functions $\mathbf{u}_t^\circ = \tilde{\mu}(\mathbf{x}_t, \tilde{\mathbf{w}}_{tn_t}^{u_0})$, $t = 0, 1, \dots, T-1$. If one is not interested in getting such a by-product, since the presence of disturbances can be completely excluded, then Problem C1 can be viewed as an NLP problem, where Cost (6.4) has to be minimized with respect to the vectors $\mathbf{u}_0, \mathbf{u}_1, \dots, \mathbf{u}_{T-1}$ and $\mathbf{x}_1, \dots, \mathbf{x}_T$ taking into account Constraints (6.1), (6.2), and (6.3).

6.7.1 Solution of Problem C1 by Gradient-Based Algorithms

This section is not so much aimed at highlighting the close connection between optimal control and NLP ([16] still remains a nice reference on this topic) but rather to point out a simple method for computing the gradient $\nabla_{\mathbf{u}_0^{T-1}} J_0(\mathbf{x}_0, \mathbf{u}_0^{T-1})$, where $\mathbf{u}_0^{T-1} \triangleq \text{col}(\mathbf{u}_0, \mathbf{u}_1, \dots, \mathbf{u}_{T-1})$ and $J_0(\mathbf{x}_0, \mathbf{u}_0^{T-1})$ is the cost (6.4) obtained by eliminating $\mathbf{x}_1, \dots, \mathbf{x}_T$ via the state equation. Of course, we must assume that the functions f_t, h_t , $t = 0, 1, \dots, T-1$, and h_T are continuously differentiable. For simplicity, assume also that Constraints (6.2) and (6.3) are absent or that they can be interpreted in a “soft way” and then transferred into Cost (6.4) by suitable penalty functions. Then, Problem C1 turns out to be an unconstrained NLP problem. This method is important as it is quite similar to the procedure for solving approximately the stochastic

extensions of Problem C1 and C1', i.e., Problems C2 and C2', respectively, which we shall describe in the next chapter.

Before presenting a gradient-based procedure to solve Problem C1, we recall the *first-order necessary condition* for the optimality of its solution (see, e.g., [6, Sect. 6.2] and [4, Sect. 1.9]). As we have assumed Problem C1 to be an NLP problem with only the equality constraint

$$\mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t) - \mathbf{x}_{t+1} = \mathbf{0}, \quad t = 0, 1, \dots, T-1, \quad (6.76)$$

with $\mathbf{x}_0 = \hat{\mathbf{x}}$, we can apply the *Lagrange multiplier theorem* (see, for instance, [4, Sect. 3.1]). We suppose that all the hypotheses of this theorem are verified. Then, we define the discrete-time Lagrangian function

$$\begin{aligned} L(\mathbf{x}_1, \dots, \mathbf{x}_T, \mathbf{u}_0, \dots, \mathbf{u}_{T-1}, \boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_T) \\ \triangleq \sum_{t=0}^{T-1} h_t(\mathbf{x}_t, \mathbf{u}_t) + h_T(\mathbf{x}_T) + \sum_{t=0}^{T-1} \boldsymbol{\lambda}_{t+1}^\top [\mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t) - \mathbf{x}_{t+1}], \end{aligned} \quad (6.77)$$

where $\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_T$ are vectors of Lagrange multipliers.

We now impose the necessary conditions for optimality. In order to highlight the terms in (6.77) that share the dependence on the same vector \mathbf{x}_t , it is convenient to rewrite L as follows:

$$\begin{aligned} L(\mathbf{x}_1, \dots, \mathbf{x}_T, \mathbf{u}_0, \dots, \mathbf{u}_{T-1}, \boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_T) &= h_0(\mathbf{x}_0, \mathbf{u}_0) + \boldsymbol{\lambda}_1^\top \mathbf{f}_0(\mathbf{x}_0, \mathbf{u}_0) \\ &+ \sum_{t=1}^{T-1} [h_t(\mathbf{x}_t, \mathbf{u}_t) + \boldsymbol{\lambda}_{t+1}^\top \mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t) - \boldsymbol{\lambda}_t^\top \mathbf{x}_t] - \boldsymbol{\lambda}_T^\top \mathbf{x}_T + h_T(\mathbf{x}_T). \end{aligned}$$

Then, we can write

$$\frac{\partial L}{\partial \mathbf{x}_t} = \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t)}{\partial \mathbf{x}_t} + \boldsymbol{\lambda}_{t+1}^\top \frac{\partial \mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t)}{\partial \mathbf{x}_t} - \boldsymbol{\lambda}_t^\top = \mathbf{0}^\top, \quad t = 1, \dots, T-1, \quad (6.78a)$$

$$\frac{\partial L}{\partial \mathbf{x}_T} = -\boldsymbol{\lambda}_T^\top + \frac{\partial h_T(\mathbf{x}_T)}{\partial \mathbf{x}_T} = \mathbf{0}^\top, \quad (6.78b)$$

where $\frac{\partial L}{\partial \mathbf{x}_t}$, $\frac{\partial h_t}{\partial \mathbf{x}_t}$, and $\frac{\partial \mathbf{f}_t}{\partial \mathbf{x}_t}$ are Jacobian matrices. From (6.78), we get

$$\boldsymbol{\lambda}_t^\top = \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t)}{\partial \mathbf{x}_t} + \boldsymbol{\lambda}_{t+1}^\top \frac{\partial \mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t)}{\partial \mathbf{x}_t}, \quad t = 1, \dots, T-1, \quad (6.79a)$$

$$\boldsymbol{\lambda}_T^\top = \frac{\partial h_T(\mathbf{x}_T)}{\partial \mathbf{x}_T}. \quad (6.79b)$$

Equation (6.79) is referred to as the *adjoint equation*. The other necessary conditions for optimality are given by

$$\frac{\partial L}{\partial \mathbf{u}_t} = \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t)}{\partial \mathbf{u}_t} + \lambda_{t+1}^\top \frac{\partial \mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t)}{\partial \mathbf{u}_t} = \mathbf{0}^\top, \quad t = 0, \dots, T-1. \quad (6.80)$$

Equations (6.79) and (6.80) are known as the *discrete-time Euler–Lagrange equations*. Note that (6.76), (6.79), and (6.80) constitute an algebraic system of $T(d + d + m)$ equations for determining the $T(d + d + m)$ variables that provide the optimal state and control trajectories and the optimal Lagrange multipliers. This system of equations gives rise to a *two-point boundary-value problem* (TPBV problem). The boundary conditions are split. For (6.76), we have the initial condition $\mathbf{x}_0 = \hat{\mathbf{x}}$; for (6.79), we have the final condition (6.79b). If the system is nonlinear, then it is generally difficult to solve it. Therefore, it is quite natural to resort to some gradient-based algorithm that we describe in the following paragraphs.

Define the costs-to-go at stage t :

$$J_t(\mathbf{x}_t, \mathbf{u}_t^{T-1}) \triangleq \sum_{k=t}^{T-1} h_k(\mathbf{x}_k, \mathbf{u}_k) + h_T(\mathbf{x}_T), \quad t = 0, 1, \dots, T-1. \quad (6.81)$$

We decompose the gradient $\nabla_{\mathbf{u}_0^{T-1}}$ as follows:

$$\nabla_{\mathbf{u}_0^{T-1}} J = \begin{bmatrix} \nabla_{\mathbf{u}_0} J \\ \vdots \\ \nabla_{\mathbf{u}_{T-1}} J \end{bmatrix} \quad (6.82)$$

and we compute the gradients $\nabla_{\mathbf{u}_t} J = \nabla_{\mathbf{u}_t} J_t$ and $\nabla_{\mathbf{x}_t} J = \nabla_{\mathbf{x}_t} J_t$, $t = 0, 1, \dots, T-1$. By using the chain rule, we can write³

$$\begin{aligned} (\nabla_{\mathbf{u}_t} J)^\top &= \frac{\partial J_t(\mathbf{x}_t, \mathbf{u}_t^{T-1})}{\partial \mathbf{u}_t} \\ &= \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t)}{\partial \mathbf{u}_t} + \frac{\partial J_{t+1}(\mathbf{x}_{t+1}, \mathbf{u}_{t+1}^{T-1})}{\partial \mathbf{x}_{t+1}} \frac{\partial \mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t)}{\partial \mathbf{u}_t}, \\ &\quad t = 0, 1, \dots, T-1. \end{aligned} \quad (6.83)$$

³We recall an operation involving partial derivatives used here and frequently in the next chapters. Consider the functions $f: \mathbb{R}^h \times \mathbb{R}^k \rightarrow \mathbb{R}$ ($f = f(\mathbf{y}, \mathbf{z})$), and the vectors $\mathbf{y}: \mathbb{R}^d \rightarrow \mathbb{R}^h$ ($\mathbf{y} = \mathbf{y}(\mathbf{x})$) and $\mathbf{z}: \mathbb{R}^d \rightarrow \mathbb{R}^k$ ($\mathbf{z} = \mathbf{z}(\mathbf{x})$). We define $\nabla_{\mathbf{x}} f \triangleq [\frac{\partial f}{\partial x_1} \cdots \frac{\partial f}{\partial x_d}]^\top = (\frac{\partial f}{\partial \mathbf{x}})^\top$. Then, we have

$$\frac{\partial f}{\partial \mathbf{x}} = \frac{\partial f}{\partial \mathbf{y}} \frac{\partial \mathbf{y}}{\partial \mathbf{x}} + \frac{\partial f}{\partial \mathbf{z}} \frac{\partial \mathbf{z}}{\partial \mathbf{x}}$$

where $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$ and $\frac{\partial \mathbf{z}}{\partial \mathbf{x}}$ are Jacobian matrices of dimensions $h \times d$ and $k \times d$, respectively.

By letting

$$\boldsymbol{\lambda}_t^\top \triangleq \frac{\partial J_t}{\partial \mathbf{x}_t}, \quad t = 1, \dots, T, \quad (6.84)$$

(6.83) can be rewritten as

$$(\nabla_{\mathbf{u}_t} J)^\top = \frac{\partial h_t}{\partial \mathbf{u}_t} + \boldsymbol{\lambda}_{t+1}^\top \frac{\partial \mathbf{f}_t}{\partial \mathbf{u}_t}, \quad t = 0, 1, \dots, T-1. \quad (6.85)$$

To complete the computations of the gradients $\nabla_{\mathbf{u}_t} J$, we must determine the vectors $\boldsymbol{\lambda}_t$. By using Definition (6.84), we have

$$\boldsymbol{\lambda}_t^\top = \frac{\partial J_t}{\partial \mathbf{x}_t} = \frac{\partial h_t}{\partial \mathbf{x}_t} + \frac{\partial J_{t+1}}{\partial \mathbf{x}_{t+1}} \frac{\partial \mathbf{f}_t}{\partial \mathbf{x}_t}, \quad t = T-1, T-2, \dots, 1, \quad (6.86)$$

hence obtaining

$$\boldsymbol{\lambda}_t^\top = \frac{\partial h_t}{\partial \mathbf{x}_t} + \boldsymbol{\lambda}_{t+1}^\top \frac{\partial \mathbf{f}_t}{\partial \mathbf{x}_t}, \quad t = T-1, T-2, \dots, 1, \quad (6.87a)$$

$$\boldsymbol{\lambda}_T^\top = \frac{\partial h_T(\mathbf{x}_T)}{\partial \mathbf{x}_T}. \quad (6.87b)$$

On this basis, one can develop a gradient-based algorithm of the following type.

Procedure 6.1

- Guess the initial value of the vector $\mathbf{u}_0^{T-1}(0)$, i.e., the control trajectory $\mathbf{u}_0(0)$, $\mathbf{u}_1(0), \dots, \mathbf{u}_{T-1}(0)$, and set $k = 0$.
- (*) Starting from $\mathbf{x}_0 = \hat{\mathbf{x}}$, compute the state trajectory $\mathbf{x}_1, \dots, \mathbf{x}_T$ by State Equation (6.1).
- Evaluate $\boldsymbol{\lambda}_T$ by (6.87b).
- Compute $\boldsymbol{\lambda}_{T-1}, \dots, \boldsymbol{\lambda}_1$ by (6.87a) (*backward iteration*).
- Compute the gradients $\nabla_{\mathbf{u}_0} J, \nabla_{\mathbf{u}_1} J, \dots, \nabla_{\mathbf{u}_{T-1}} J$ and then $\nabla_{\mathbf{u}_0^{T-1}} J$ (see (6.82) and (6.85)).
- Compute the “new” vector $\mathbf{u}_0^{T-1}(k+1)$ by using some gradient-based algorithm.
- If some suitable termination criterion is satisfied, then stop (the “suitable” criterion is satisfied if the distance measured according to some norm between the “new” vector and the “old” one and/or the difference between the “old” value of the cost and the “new” one and/or some other quantity, like the norm of $\nabla_{\mathbf{u}_0^{T-1}} J$, are smaller than a given threshold).
- Otherwise, set $k \leftarrow k + 1$ and go to (*). \triangleleft

Of course, variants of Problem C1 can be stated with other types of constraints on the initial and/or the final state vectors. Other TPBV problems provide necessary conditions for optimality; see, for instance, [6] for several examples. Furthermore,

Constraints (6.2) and (6.3) may be present. Typically, they can take on the form of equalities or inequalities, that is,

$$\mathbf{g}_t(\mathbf{x}_t, \mathbf{u}_t) = \mathbf{0}, \quad t = 0, 1, \dots, T - 1,$$

$$\tilde{\mathbf{g}}_t(\mathbf{x}_t, \mathbf{u}_t) \leq \mathbf{0}, \quad t = 0, 1, \dots, T - 1,$$

where we have assumed that \mathbf{x}_0 is an initial state vector to be optimized. There may be a target set, described by the inequalities

$$\tilde{\mathbf{g}}_T(\mathbf{x}_T) \leq \mathbf{0},$$

in which the state must be steered. \mathbf{g}_t , $\tilde{\mathbf{g}}_t$, and $\tilde{\mathbf{g}}_T$ are vector-valued functions of appropriate dimensions. Then, the Lagrangian function can be determined and the Karush–Kuhn–Tucker necessary conditions for optimality can be established. Once one has derived the necessary conditions for optimality, it should be able to compute the optimal state and control trajectories. This can be done by determining the partial derivatives of the Lagrangian function so as to apply a suitable gradient-based algorithm for constrained NLP. Hopefully, this algorithm leads the trajectories to satisfy the necessary optimality conditions.

6.7.2 The Final Time of Problem C1 is not Fixed

Finally, it is important to highlight that Cost (6.4) may depend on the duration of the decision process. As we said in Sect. 6.1, this possible dependence is connected to the origins of Problem C1 and, in general, to the nature of the T -stage decision problem we have to solve. For the reasons we have given before formulating Problem 6.1, this problem is simple but meaningful. With respect to Problem 1.2, the final time t_f has to be determined and the constraints have been omitted. In this section, we address the discrete-time approximation of the problem.

Assume that t_f is free but that the number T of stages is fixed and known. Then the fixed-time approximation is obtained by partitioning the interval $[0, t_f]$ into T subintervals of duration $\Delta_0 = t_1$, $\Delta_1 = t_2 - t_1, \dots, \Delta_{T-1} = t_T - t_{T-1} = t_f - t_{T-1}$. Integral (6.5) is approximated by a summation and State Equation (1.12) by the following discrete-time one (a first-order Euler's method is used):

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{f}_c(\mathbf{x}_k, \mathbf{u}_k, t_k) \Delta_k = \mathbf{f}_k(\mathbf{x}_k, \mathbf{u}_k, \Delta_k), \quad k = 0, 1, \dots, T - 1, \quad (6.88)$$

where $\mathbf{x}_0 = \hat{\mathbf{x}}$, $\mathbf{x}_k = \mathbf{x}(t_k)$ and $\mathbf{u}_k = \mathbf{u}(t_k)$, $k = 0, 1, \dots, T - 1$. Then, Problem 6.1 is “approximated” by the following “extended” form of Problem C1.

Problem C1 t_f . Find the sequence of optimal control vectors $\mathbf{u}_0^\circ, \mathbf{u}_1^\circ, \dots, \mathbf{u}_{T-1}^\circ$ and the optimal sequence of sampling interval durations $\Delta_0^\circ, \Delta_1^\circ, \dots, \Delta_{T-1}^\circ$ that minimize the cost

$$J = \sum_{k=0}^{T-1} h_c(\mathbf{x}_k, \mathbf{u}_k, t_k) \Delta_k + h_f(\mathbf{x}_T, t_T) \quad (6.89)$$

subject to Constraints (6.88) and

$$t_{k+1} = t_k + \Delta_k, \quad k = 0, 1, \dots, T - 1, \quad (6.90)$$

$$\Delta_k \geq 0, \quad k = 0, 1, \dots, T - 1, \quad (6.91)$$

where $t_0 = 0$. \triangleleft

An additional positive upper bound on each Δ_k may be inserted in the problem formulation to avoid a too large error at optimality in the approximation of the original continuous-time dynamic system by a discrete-time one. Note that the sampling instants t_k and the sampling intervals durations Δ_k (connected to each other by (6.90)) can be interpreted as scalar state and control variables in an “auxiliary” state equation, respectively. Equation (6.90) has to be “added” to the process state equation (6.88). Of course, the optimal final time is given by

$$t_f^\circ = \sum_{k=0}^{T-1} \Delta_k^\circ.$$

Therefore, Problem C1 t_f is a TPBV problem similar to the one considered in Sect. 6.7.1. As mentioned earlier, minor changes are required if one interprets the constraints (6.91) in a “soft way” by using some suitable penalty functions. Otherwise, one may resort to constrained NLP algorithms.

Clearly, all this has the drawback that only a fixed value of the integer T has been considered. If it is believed that such a drawback is too limiting, various values of T can be suitably chosen in order to minimize J also with respect to T . To avoid too many computationally cumbersome attempts, some techniques can be derived from direct line-search methods.

References

1. Bellman R (1957) Dynamic programming. Princeton University Press
2. Bellman R (1961) Adaptive control processes: a guided tour. Princeton University Press
3. Bellman R, Dreyfus SE (1962) Applied dynamic programming. Princeton University Press
4. Bertsekas DP (1999) Nonlinear programming. Athena Scientific, 2nd edn
5. Blanchini F, Miani S (2008) Set-theoretic methods in control. Birkhäuser
6. Bryson AE, Ho Y-C (1975) Applied optimal control. Hemisphere Publishing Corporation

7. Cervellera C, Gaggero M, Macciò D (2014) Low-discrepancy sampling for approximate dynamic programming with local approximators. *Comput Oper Res* 43:108–115
8. Cervellera C, Gaggero M, Macciò D, Marcialis R (2015) Efficient use of Nadaraya-Watson models and low-discrepancy sequences for approximate dynamic programming. In: Proceedings of the international joint conference on neural networks
9. Cervellera C, Muselli M (2007) Efficient sampling in approximate dynamic programming. *Comput Optim Appl* 38:417–443
10. Gaggero M, Gnecco G, Sanguineti M (2013) Dynamic programming and value-function approximation in sequential decision problems: Error analysis and numerical results. *J Optim Theory Appl* 156:380–416
11. Gaggero M, Gnecco G, Sanguineti M (2014) Approximate dynamic programming for stochastic N -stage optimization with application to optimal consumption under uncertainty. *Comput Optim Appl* 58:31–85
12. Hastie T, Tibshirani R, Friedman J (2008) The elements of statistical learning. Springer
13. Larson RE (1968) State increment dynamic programming. Elsevier
14. Powell WB (2011) Approximate dynamic programming: solving the curse of dimensionality. Wiley, 2nd edn
15. Stokey NL, Lucas RE, Prescott EC (1989) Recursive methods in economic dynamics. Harvard University Press
16. Tabak D, Kuo BK (1971) Optimal control by mathematical programming. Prentice Hall
17. Zoppoli R (1972) Minimum-time routing as an N -stage decision process. *J Appl Meteorol* 11:429–435

Chapter 7

Stochastic Optimal Control with Perfect State Information over a Finite Horizon



In the previous chapter, we dealt with an optimal control problem stated in a deterministic context. We shall now consider the case where random disturbances act on the dynamic system and the cost functional. We assume the state vector \mathbf{x}_t to be perfectly measurable at any decision stage. The possibility of measuring \mathbf{x}_t is obviously of fundamental importance because it enables one to design a control system with a closed-loop structure. Unlike in Problem C1, the control and state trajectories are no longer predictable. While in Problem C1 one has to find a sequence of optimal controls \mathbf{u}_t^o , $t = 0, 1, \dots, T - 1$, in the stochastic optimal control problem that we shall call Problem C2 one must determine a sequence of control functions $\mathbf{u}_t^o = \mu_t^o(\mathbf{x}_t)$, $t = 0, 1, \dots, T - 1$. As we emphasized in Sect. 1.1, there is a basic difference between Problem C1 and Problem C2. Problem C1 is a finite-dimensional optimization (FDO) problem or a nonlinear programming (NLP) problem, whereas Problem C2 is an infinite-dimensional optimization (IDO) problem or a functional optimization problem, since continuous random disturbances are considered. Note that Problem C2 is substantially Problem 1.4 stated in Sect. 1.5.3 as an example of IDO Problem.

In this chapter, we shall address another problem besides Problem C2, defined as Problem C2'. In Problem C2, the initial state $\mathbf{x}_0 = \hat{\mathbf{x}}$ is fixed while in Problem C2' \mathbf{x}_0 can take on any value from a given set $X_0 \subseteq \mathbb{R}^d$. If there is no time to determine online the control function $\mu_0^o(\mathbf{x}_0)$, then the decision-maker (DM) must be provided with this function computed a priori. Both problems are stated in Sect. 7.1. Throughout the chapter, there will be no difficulty in passing from one problem to the other.

The chapter is divided into two parts. Because of the very general assumptions under which Problem C2 is formulated (for example, the classical LQ hypotheses are not assumed), the problem is faced by an approximate technique in both parts: in the first part by approximate dynamic programming (ADP) and in the second part by the extended Ritz method (ERIM). In Sect. 7.2, we begin with presenting a

“formal” approach based on an “exact” application of dynamic programming (DP). Then, in Sect. 7.3, three approximations are addressed. Two of these are similar to those described in Chap. 6. In fact, state discretization by Monte Carlo (MC) and especially by quasi-Monte Carlo (quasi-MC) techniques is considered to mitigate the curse of dimensionality, instead of using regular grids. Further, the optimal cost-to-go functions are approximated by FSP functions. The backward equation is obviously more complicated than the corresponding equation in Problem C1, because of the necessity to perform the averaging operations. MC and quasi-MC techniques are presented following the line of reasoning discussed in Chap. 5. Section 7.4 provides a brief survey on discretization and approximation issues in ADP, some comments on the preservation of structural properties of optimal cost-to-go and control functions through each iteration of the DP procedure, and a short description of the connections between DP and reinforcement learning.

In Sect. 7.5, which begins the second part of the chapter, the ERIM is proposed as a new approximate tool to solve Problem C2, as an alternative to ADP. More precisely, Problem C2 is replaced with the FDO Problem C2_n, where, at each decision stage t , the FSP control functions $\tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn})$ take the place of the control functions μ_t (the components of the vectors \mathbf{w}_{tn} are “free” parameters to be optimized). All this is described in Sect. 7.6, where batch algorithms and stochastic gradient algorithms are presented for the solution of the NLP Problem C2_n. In particular, if suitable differentiability assumptions are verified, then the gradient of the cost can be computed analytically. Such a computation can be performed by resorting, for a given sequence of the disturbances ξ_t , to a recursive equation that is obtained from the classical adjoint equation for the solution of Problem C1 with the addition of one term, dependent on the chosen family of FSP functions.

Examples comparing ADP with the ERIM are presented in Sect. 7.7. The examples consider networks of water reservoirs and are characterized by high numbers of state components. The ERIM behaves more or less as ADP. This is important since ADP proves its great efficiency in problems like T -stage optimal control problems but, as mentioned several times, it cannot be applied in a large variety of optimization problems where the ERIM does not encounter great difficulties. Finally, in Sect. 7.8, some problems equivalent to Problem C2 are considered.

7.1 Statement of Problems C2 and C2'

We consider the discrete-time stochastic dynamic system

$$\mathbf{x}_{t+1} = f_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t), \quad t = 0, 1, \dots, T-1, \quad (7.1)$$

where $\mathbf{x}_0 = \hat{\mathbf{x}}$ is a given initial state. As in Problem C1, the control vector \mathbf{u}_t may be constrained to take values from a set $U_t(\mathbf{x}_t)$ dependent on t and the current state \mathbf{x}_t , that is (see (6.3)),

$$\mathbf{u}_t \in U_t(\mathbf{x}_t) \subseteq \mathbb{R}^m, \quad t = 0, 1, \dots, T-1. \quad (7.2)$$

$\xi_0, \xi_1, \dots, \xi_{T-1}$ are random disturbances acting on the dynamic system (7.1). We also assume that ξ_t is characterized by a known probability density function that may depend on \mathbf{x}_t and \mathbf{u}_t , then taking on the form $p_t(\xi_t | \mathbf{x}_t, \mathbf{u}_t)$. This probability density does not depend on the previous disturbances $\xi_0, \xi_1, \dots, \xi_{t-1}$ if not indirectly through \mathbf{x}_t .

As is frequent in the stochastic control literature, we do not necessarily introduce constraint sets like (6.2) on the state vector, i.e., $\mathbf{x}_t \in X_t \subseteq \mathbb{R}^d$, $t = 1, \dots, T$ (unless it is possible to properly define such sets). Indeed, we have typically no guarantee that the state equation generates state vectors belonging to the constraint set X_{t+1} , for any $\mathbf{x}_t \in X_t$, any $\mathbf{u}_t \in U_t(\mathbf{x}_t)$, and any $\xi_t \in \Xi_t(\mathbf{x}_t, \mathbf{u}_t) \subseteq \mathbb{R}^q$, $t = 0, 1, \dots, T-1$. However, in Sect. 7.3 we shall address the numerical solution of Problem C2 by DP. In that case, it will be necessary to limit the sets where the states \mathbf{x}_t take on their values. Some simple arguments on the definition of such sets will be necessary (see Sect. 7.2).

The cost function has the same additive form as in the deterministic case, that is,

$$J = \sum_{t=0}^{T-1} h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t) + h_T(\mathbf{x}_T). \quad (7.3)$$

As the state vector is driven by the action of the control vectors and the random disturbances, we cannot foresee its trajectory over time. However, the possibility of perfectly measuring the state provides us with the maximum possible information. Therefore, the most effective way of acting is to generate, at each stage t , the control vector \mathbf{u}_t as a function of the current state \mathbf{x}_t . Then, we define the *closed-loop control functions*

$$\mathbf{u}_t = \mu_t(\mathbf{x}_t), \quad t = 0, 1, \dots, T-1. \quad (7.4)$$

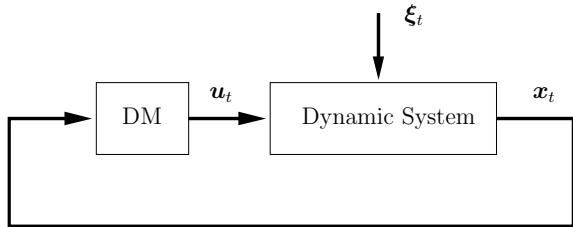
Since the initial state $\mathbf{x}_0 = \hat{\mathbf{x}}$ is fixed, a vector \mathbf{u}_0 has to be optimized instead of a function $\mu_0(\mathbf{x}_0)$. As J is a random variable, a reasonable approach to designing the optimal control law consists of minimizing the expected value of J , i.e., the expected cost

$$\hat{J}(\hat{\mathbf{x}}) = \underset{\xi_0, \xi_1, \dots, \xi_{T-1}}{\mathbb{E}} \left\{ h_0(\mathbf{x}_0, \mathbf{u}_0, \xi_0) + \sum_{t=1}^{T-1} h_t[\mathbf{x}_t, \mu_t(\mathbf{x}_t), \xi_t] + h_T(\mathbf{x}_T) \right\}, \quad (7.5)$$

where $\mathbf{x}_0 = \hat{\mathbf{x}}$.

A more general situation may be conceived, in which it is required that an optimal control vector \mathbf{u}_0° be associated to any initial state vector $\mathbf{x}_0 \in X_0$, where X_0 is the set of the possible states from which the decision process may have to start. In this case, a function $\mu_0(\mathbf{x}_0)$ must also be optimized for any $\mathbf{x}_0 \in X_0$, and Cost (7.5) is replaced by the following one:

Fig. 7.1 The closed-loop structure of the control scheme



$$\hat{J}(\mathbf{x}_0) \triangleq \underset{\xi_0, \xi_1, \dots, \xi_{T-1}}{\mathbb{E}} \left\{ \sum_{t=0}^{T-1} h_t[\mathbf{x}_t, \mu_t(\mathbf{x}_t), \xi_t] + h_T(\mathbf{x}_T) \right\}. \quad (7.6)$$

As we shall see in Sects. 7.2 and 7.5, when applying DP there is no difference in minimizing Costs (7.5) and (7.6). When applying the ERIM, however, one has to be careful. In that case, for the sake of clarity, it is advisable to state two distinct T -stage stochastic optimal control problems.

Problem C2. Find the optimal control vector \mathbf{u}_0° and the sequence of optimal control functions $\mu_1^\circ(\mathbf{x}_1), \dots, \mu_{T-1}^\circ(\mathbf{x}_{T-1})$ (i.e., the optimal control law) that minimize the cost $\hat{J}(\hat{\mathbf{x}})$ subject to Constraints (7.1) and (7.2). \triangleleft

Problem C2'. Find the sequence of optimal control functions $\mu_0^\circ(\mathbf{x}_0), \mu_1^\circ(\mathbf{x}_1), \dots, \mu_{T-1}^\circ(\mathbf{x}_{T-1})$ (i.e., the optimal control law) that minimizes the cost $\hat{J}(\mathbf{x}_0)$ for any $\mathbf{x}_0 \in X_0$ subject to Constraints (7.1) and (7.2). \triangleleft

For both problems, the closed-loop structure of the control system solving Problems C2 and C2' is shown in Fig. 7.1.

In rough mathematical terms, Problem C2 is related to Problem PM with $M = T - 1$ (see the discussion in Sect. 2.9).

7.2 Solution of Problems C2 and C2' by Dynamic Programming: The Exact Approach

In the following, we discuss the application of DP to solve Problems C2 and C2'. In order to simplify the discussion without significant conceptual losses, we assume that the random vectors $\xi_0, \xi_1, \dots, \xi_{T-1}$ are mutually independent with known probability densities $p_t(\xi_t)$, $t = 0, 1, \dots, T - 1$, which do not depend on \mathbf{x}_t and \mathbf{u}_t ; hence, also the domains $\mathcal{E}_t(\mathbf{x}_t, \mathbf{u}_t)$ do not actually depend on \mathbf{x}_t and \mathbf{u}_t , and are denoted in the following by \mathcal{E}_t (for a general case, see [11]). The application of DP is then quite similar to what we did to solve Problem C1 in Sect. 6.3. Before defining the optimal cost-to-go functions, we have to specify the sets \bar{X}_t , $t = 1, \dots, T - 1$, where

such functions have their domain and must be computed. In the next section, when we shall address the numerical application of DP, we have to construct discretized sets X'_t similar to (6.20). We limit ourselves to considering Problem C2' since, as stated previously, DP can be applied to the solution of Problem C2 in a completely analogous way by simply replacing \bar{X}_0 with $x_0 = \hat{x}$ and $\mu_0(x_0)$ with u_0 .

In order to derive the sets \bar{X}_t (hence, the sets X'_t), we need to assume that the state vector, evolving under the actions of control u_t and of the random noise ξ_t , remains inside bounded regions \bar{X}_t , $t = 1, \dots, T$, of the state space. This seems quite reasonable since Constraints (7.2) on the control vectors generally exist. Similarly, it is very likely that the sets \mathcal{E}_t are bounded. Note that, in one among several possible extensions of Problems C2 and C2', it may happen that the density $p_t(x_t)$ is unknown but that the sets \mathcal{E}_t are known. In this case, such sets are important to characterize the vectors ξ_t and are called *set-membership constraints for the disturbances* (see, for instance, [12] or, more extensively, [11, Sect. 4.6.2]). We shall come back to this issue in Sect. 7.2 when we shall mention minimax control. Moreover, in another extension, it may happen that the sets \mathcal{E}_t are not explicitly specified. In this case, practical considerations may lead us to construct them, even if only in a qualitative way. For example, as an extreme case, even if we were not sure that the disturbances ξ_t are really bounded, we could define “reasonably large” sets \mathcal{E}_t .

Coming back to Problems C2 and C2', we now assume the disturbances ξ_t to be bounded. On this basis, there is no reason to exclude the constraint sets X_t associated with the states as we assumed at the beginning of Sect. 7.1. We have now to point out that, for a given state $x_t \in \bar{X}_t$, not all the controls $u_t \in U_t(x_t)$ are necessarily always admissible. Indeed, the DM can choose the control u_t but it (obviously) has only a partial influence on the transition of the state from x_t to x_{t+1} , owing to the action exerted by ξ_t . Consequently, the DM can choose u_t only if this control leads the state x_{t+1} to belong to the set X_{t+1} . We summarize what has been said by defining the following sequence of sets:

$$\begin{aligned} \bar{X}_{t+1} &= \{x_{t+1} \in \mathbb{R}^d : \exists x_t \in \bar{X}_t \text{ and } \exists u_t \in U_t(x_t) \text{ such that} \\ &\quad (1) \forall \xi_t \in \mathcal{E}_t \text{ one has } f_t(x_t, u_t, \xi_t) \in X_{t+1}, \text{ and} \\ &\quad (2) \exists \xi_t \in \mathcal{E}_t \text{ for which } x_{t+1} = f_t(x_t, u_t, \xi_t)\}, \\ t &= 0, 1, \dots, T-1, \\ \bar{X}_0 &= X_0. \end{aligned} \tag{7.7}$$

Of course, $\bar{X}_0 = \{\hat{x}\}$ for Problem C2.

To compute the sets \bar{X}_t through Eqs. (7.7) analytically is in general a very difficult task unless the state equation (7.1) and the various sets enjoy sufficiently simple structures. Similarities between the sets (7.7) and the “effective target sets” are considered in [11, Sect. 4.6.2] and in previous works by the same author about the “control with unknown but bounded disturbances.” We simply limit ourselves to a “forward”

definition of Sets (7.7). However, in [11], for instance, a “backward” constructive recursion is presented, based on a DP approach in a minimax context.

In particular, two cases are addressed. In both, the state equation (7.1) is assumed to be linear, i.e.,

$$\mathbf{x}_{t+1} = A_t \mathbf{x}_t + B_t \mathbf{u}_t + \boldsymbol{\xi}_t, \quad t = 0, 1, \dots, T-1,$$

and the sets $U_t(\mathbf{x}_t)$ to be independent of \mathbf{x}_t ; hence, in the following they are denoted by U_t . In the first case, the sets X_t , U_t , and Ξ_t are polyhedral. It can be shown that the target sets are polyhedral as well and that they can be determined by linear programming techniques. In the second case, X_t , U_t , and Ξ_t are ellipsoids. The effective target sets are not ellipsoids but they can be approximated by internal ellipsoids.

Since we have to construct the sets (7.7) in a very qualitative way to obtain the sample sets X'_t , the introduction of the constraint sets X_t (when available) and Ξ_t is important as they allow us to keep Sets (7.7) bounded. Any type of method can be applied for our purposes, be it exact or widely approximated, as an approximation is unavoidable in most cases. Once Sets (7.7) have been built, both sets X_t and Ξ_t can be forgotten, unless they are intrinsically part of the optimal control problem we have to address.

After these preliminary considerations, we derive the recursive equation of DP in a “somewhat informal” way. Such a derivation extends the “proof” of Eq. (6.16) from the deterministic to the stochastic case.

Let us define the optimal cost-to-go functions

$$\hat{J}_t^\circ(\mathbf{x}_t) \triangleq \min_{\mu_1, \dots, \mu_{T-1}} \mathbb{E}_{\xi_1, \dots, \xi_{T-1}} \left\{ \sum_{k=1}^{T-1} h_k[\mathbf{x}_k, \mu_k(\mathbf{x}_k), \boldsymbol{\xi}_k] + h_T(\mathbf{x}_T) \right\}, \\ \mathbf{x}_t \in \bar{X}_t, \quad t = 0, 1, \dots, T-1. \quad (7.8)$$

Obviously, the minimum total expected cost is given by $\hat{J}_0^\circ(\mathbf{x}_0)$ for any $\mathbf{x}_0 \in X_0$.

As the control vectors $\mathbf{u}_1 = \mu_1(\mathbf{x}_1), \dots, \mathbf{u}_{T-1} = \mu_{T-1}(\mathbf{x}_{T-1})$ do not affect the cost $h_0[\mathbf{x}_0, \mu_0(\mathbf{x}_0), \boldsymbol{\xi}_0]$, we can write

$$\begin{aligned} \hat{J}_0^\circ(\mathbf{x}_0) &= \min_{\mu_0} \left\{ \mathbb{E}_{\boldsymbol{\xi}_0} h_0[\mathbf{x}_0, \mu_0(\mathbf{x}_0), \boldsymbol{\xi}_0] \right. \\ &\quad \left. + \min_{\mu_1, \dots, \mu_{T-1}} \mathbb{E}_{\boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_{T-1}} \left(\sum_{k=1}^{T-1} h_k[\mathbf{x}_k, \mu_k(\mathbf{x}_k), \boldsymbol{\xi}_k] + h_T(\mathbf{x}_T) \right) \right\}. \end{aligned} \quad (7.9)$$

Consider now the second term in the right-hand side of Eq. (7.9) and note that, whatever the values of \mathbf{u}_0 and $\boldsymbol{\xi}_0$, what we have to do at \mathbf{x}_1 is to determine the control functions $\mu_1^\circ, \dots, \mu_{T-1}^\circ$ minimizing the remaining portion of the expected process

cost. Then, we can move the control functions μ_1, \dots, μ_{T-1} inside the averaging operation with respect to ξ_0 and write¹

$$\begin{aligned}\hat{J}_0^\circ(\mathbf{x}_0) &= \min_{\mu_0} \mathbb{E}_{\xi_0} \left\{ h_0[\mathbf{x}_0, \mu_0(\mathbf{x}_0), \xi_0] \right. \\ &\quad \left. + \min_{\mu_1, \dots, \mu_{T-1}} \mathbb{E}_{\xi_1, \dots, \xi_{T-1}} \left(\sum_{k=1}^{T-1} h_k[\mathbf{x}_k, \mu_k(\mathbf{x}_k), \xi_k] + h_T(\mathbf{x}_T) \right) \right\}. \quad (7.10)\end{aligned}$$

By applying (7.8) we obtain

$$\hat{J}_0^\circ(\mathbf{x}_0) = \min_{\mu_0} \mathbb{E}_{\xi_0} \left\{ h_0[\mathbf{x}_0, \mu_0(\mathbf{x}_0), \xi_0] + \hat{J}_1^\circ(\mathbf{x}_1) \right\}. \quad (7.11)$$

Given any function G of \mathbf{x} and \mathbf{u} , we have

$$\min_{\mu} G[\mathbf{x}, \mu(\mathbf{x})] = \min_{\mathbf{u}} G(\mathbf{x}, \mathbf{u}), \quad (7.12)$$

that is, we can replace the minimization with respect to the admissible function μ with the minimization with respect to the admissible vector \mathbf{u} . Then, using (7.12), (7.11) becomes

$$\hat{J}_0^\circ(\mathbf{x}_0) = \min_{\mathbf{u}_0} \mathbb{E}_{\xi_0} \left[h_0(\mathbf{x}_0, \mathbf{u}_0, \xi_0) + \hat{J}_1^\circ(\mathbf{x}_1) \right].$$

By repeating the same procedure for stages $t = 1, \dots, T - 1$, we obtain the “exact” recursive equation for DP (the term “exact” is used with the meaning specified in Sect. 6.3):

$$\hat{J}_T^\circ(\mathbf{x}_T) = h_T(\mathbf{x}_T), \quad (7.13a)$$

$$\begin{aligned}\hat{J}_t^\circ(\mathbf{x}_t) &= \min_{\mathbf{u}_t \in U_t(\mathbf{x}_t)} \mathbb{E}_{\xi_t} \left\{ h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t) + \hat{J}_{t+1}^\circ[f(\mathbf{x}_t, \mathbf{u}_t, \xi_t)] \right\}, \\ \mathbf{x}_t &\in \bar{X}_t, \quad t = T - 1, T - 2, \dots, 0. \quad (7.13b)\end{aligned}$$

¹As pointed out in Proposition 1.3.1 and in Sect. 1.5 of [11], this proof is “somewhat informal.” In particular, moving the minimum with respect to μ_1, \dots, μ_{T-1} inside the expectation with respect to ξ_0 is generally not a mathematically correct step of the proof, unless suitable conditions hold. It works, for example, if, among other assumptions, for any \mathbf{x}_0 and any admissible $\mathbf{u}_0 \in U_0(\mathbf{x}_0)$, the disturbance ξ_0 takes on a finite or a countable number of values in the set Ξ_0 , and the expected values of all the cost terms in (7.10) are finite. These expected values can be computed as summations of a finite or countable number of terms weighted by the probabilities of the elements in Ξ_0 . Moreover, as the proof continues, measurability assumptions may be required on the functions f_t, h_t, h_T , and μ_t . Quoting [11, p. 42], we report: “It turns out that these” (i.e., the abovementioned) “difficulties are mainly technical and do not substantially affect the basic results to be obtained. For this reason, we find it convenient to proceed with informal derivations and arguments; this is consistent with most of the literature on the subject.” Finally, it is worth noting that, for a rigorous treatment of DP, [11] refers to [14].

If we compare the determination of Eq. (7.13) with Eq. (6.16), we may point out what follows. The determination of the optimal control functions $\mu_0^\circ(\mathbf{x}_0)$, $\mu_1^\circ(\mathbf{x}_1), \dots, \mu_{T-1}^\circ(\mathbf{x}_{T-1})$ to solve Problems C1 and C2' (or C2) in the *backward phase* of DP is similar, apart from the fact that in the stochastic case we have to compute averages. The *forward phase*, however, is necessary for solving Problem C1 since we have to determine the sequence of the optimal control vectors $\mathbf{u}_0^\circ, \mathbf{u}_1^\circ, \dots, \mathbf{u}_{T-1}^\circ$. On the contrary, once Problem C2' has been solved, i.e., once the optimal closed-loop control functions have been derived, the forward phase is needed only to generate the optimal controls online.

Of course, the DP recursive equation (7.13) also constitutes the solution tool for Problem C2. In practice, one has to replace the set \bar{X}_0 with the initial state $\mathbf{x}_0 = \hat{\mathbf{x}}$. Then, only the optimal control vector \mathbf{u}_0° and the total cost $\hat{J}_0^\circ(\mathbf{x}_0)$ must be computed.

Note that the minimizations in (7.13b) call for the solution of an infinite number of NLP problems similar to Problems 6.2, at any decision stage t and for any state \mathbf{x}_t . Therefore, Recursive Equation (7.13) generally has a more conceptual than a practical meaning. It follows that, at each stage t , we have to discretize the state space in a finite number of points. This implies approximations that we shall describe in Sects. 7.3 and 7.4.

Minimax Control

The minimization of the expected value of Cost (7.3) is the criterion most frequently used in the optimal control of stochastic systems. This criterion generally meets the demands of the analyst quite well. However, it is not the only choice that can be made. For example, the minimization of a compromise between the expected value and the variance of J may constitute an attractive alternative. Unfortunately, this involves great computational difficulties even under simplifying assumptions.

In many practical situations – whenever a stochastic description of the random variables is not available, i.e., the probability densities $p_t(\xi_t | \mathbf{x}_t, \mathbf{u}_t)$ are not known – another choice may be required, given by minimax control. However, it may happen that the sets on which the random variables take their values are known. Denote such sets as $\Xi_t(\mathbf{x}_t, \mathbf{u}_t)$ using the same notation introduced before. We have $\xi_t \in \Xi_t(\mathbf{x}_t, \mathbf{u}_t) \subseteq \mathbb{R}^q$, $t = 0, 1, \dots, T - 1$. Then, a pessimistic but reasonable approach consists in assuming that the variables ξ_t are generated by an adversary which seeks to maximize the cost J , whereas the DM wants to minimize it. This is the origin of the following problem (the DM starts from a state $\mathbf{x}_0 \in X_0$, i.e., we consider the minimax version of Problem C2'; a similar statement can be derived for Problem C2).

Problem C2'-minimax. Find the sequence of optimal control functions $\mu_0^*(\mathbf{x}_0)$, $\mu_1^*(\mathbf{x}_1), \dots, \mu_{T-1}^*(\mathbf{x}_{T-1})$) that minimizes the cost

$$\max_{\substack{\xi_t \in \Xi_t(\mathbf{x}_t, \mu(\mathbf{x}_t)) \\ t=0,1,\dots,T-1}} \left\{ \sum_{t=0}^{T-1} h_t[\mathbf{x}_t, \mu_t(\mathbf{x}_t), \xi_t] + h_T(\mathbf{x}_T) \right\}, \quad (7.14)$$

subject to Constraints (7.1) and (7.2). \triangleleft

It is rather easy to realize the possibility of applying the principle of optimality by replacing the expectation with the maximization of the costs with respect to the random variables. The principle of optimality leads to the following DP procedure:

$$J_T^*(\mathbf{x}_T) = h_T(\mathbf{x}_T), \quad (7.15a)$$

$$\begin{aligned} J_t^*(\mathbf{x}_t) &= \min_{\mathbf{u}_t \in U_t(\mathbf{x}_t)} \max_{\xi_t \in \Xi_t(\mathbf{x}_t, \mathbf{u}_t)} \left\{ h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t) + J_{t+1}^*[f(\mathbf{x}_t, \mathbf{u}_t, \xi_t)] \right\}, \\ \mathbf{x}_t &\in \bar{X}_t, \quad t = T - 1, T - 2, \dots, 0. \end{aligned} \quad (7.15b)$$

For the proof of DP Eq. (7.15) as well as for further considerations about minimax control, we refer to [11]. Finally, if the computational efforts required by the expectation operations in (7.13) and the maximizations in (7.15) are not too different, what we say in the next section about the approximate numerical solution of Eq. (7.13) and the related computational burden also applies, generically speaking, to Eq. (7.15). Unfortunately, even under simplifying assumptions, Problem C2'-minimax can hardly be solved in closed form.

7.3 Sampling of the State Space, Application of FSP Functions, and Computation of Expected Values: Approximate Dynamic Programming

Repeating what we said in Sect. 6.4 for Problem C1, we point out that Problem C2' is also stated under assumptions too general to solve DP Eqs. (7.13) analytically. State discretization and approximation of the optimal costs-to-go by FSP functions are necessary again. These will be described in Sect. 7.3.1. Furthermore, the numerical solution of Eqs. (7.13) is obviously more complicated than the numerical solution of Eqs. (6.16) owing to the necessity of performing the averaging operations required by the presence of the random vectors $\xi_0, \xi_1, \dots, \xi_{T-1}$. Indeed, doing averaging operations is not an easy task, as we shall see in Sect. 7.3.2 (we have discussed this fact in Chap. 5). In Sect. 7.3.3, we shall address the offline and online computations of the optimal controls in the forward phase of DP. The error propagation in DP will be briefly discussed in Sect. 7.3.4.

7.3.1 Sampling of the State Space and Approximations by FSP Functions

As in Sect. 6.4.1 we have to select, at each stage t , L_t discretized points $\mathbf{x}_t^{(l)}$, $l = 1, \dots, L_t$. Then, we design the sets X'_t as subsets of the regions \bar{X}_t , that is (see (6.20)),

$$X'_t \triangleq \{\mathbf{x}_t^{(l)} \in \bar{X}_t : l = 1, \dots, L_t\}, \quad t = 0, 1, \dots, T - 1. \quad (7.16)$$

We discard the possibility of discretizing each component of the state vector \mathbf{x}_t in q values, that is, the possibility of resorting to a full uniform grid made of q^d points. We assume that such points are selected so as to mitigate the curse of dimensionality. This means that we adopt the MC approach or, better, discretization techniques aimed at spreading the samples of X'_t in the “most uniform way” by deterministic algorithms like quasi-MC methods (see Chap. 4), orthogonal arrays, Latin hypercubes, etc.

To apply DP, we have to solve the minimizations in (7.13b) for any discretized vector $\mathbf{x}_t^{(l)}$. However, as pointed out in Sect. 6.4, for such discretized vectors the minimizations cannot be performed “exactly” since, in general, each next-stage vector $\mathbf{x}_{t+1} = \mathbf{f}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \boldsymbol{\xi}_t)$ does not coincide with any previously quantized state belonging to X'_{t+1} . Of course, this does not occur at the stage $T - 1$, since the final cost $h_T(\mathbf{x}_T)$ is known by definition for any $\mathbf{x}_T \in X_T$. At the stages $T - 2, T - 3, \dots, 0$, the optimal cost-to-go functions $\hat{J}_t^\circ(\mathbf{x}_t^{(l)})$ can only be evaluated approximately.

If simple approximations like replacing the cost $\hat{J}_{t+1}^\circ(\mathbf{x}_{t+1})$ ($\mathbf{x}_{t+1} \notin X'_{t+1}$) with the value of the cost of the nearest discretized neighbor $\mathbf{x}_{t+1}^{(l*)}$ are considered too coarse, we may focus again on the possibility of approximating the optimal cost-to-go functions $\tilde{J}_t^\circ(\mathbf{x}_t)$ or, put it better, their approximations $\bar{J}_t^\circ(\mathbf{x}_t)$ by FSP functions of the form

$$\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t}), \quad t = 0, 1, \dots, T - 1, \quad (7.17)$$

belonging to the nested families (6.28).

Remark on Notation 7.1 In order not to confuse the reader with tedious and complicated notation, we refer again to Remark on Notation 6.5. The only change from such notation consists in the fact that, from now on, the “true” optimal cost-to-go functions J_t° are replaced by the “true” expected optimal costs \hat{J}_t° (so we have the additional symbol “ $\hat{\cdot}$ ” to denote expectation). For the rest, we continue to use the *approximate optimal* \tilde{J}_t° and the *FSP optimal cost-to-go functions* \bar{J}_t° to approximate the respective costs. A similar notation will be used for the control functions. \triangleleft

In order to optimize the FSP functions (7.17), we define I/O sample sets similar to (6.25), (6.29), that is,

$$\Sigma_t \triangleq \left\{ \left(\mathbf{x}_t^{(l)}, \bar{J}_t^\circ(\mathbf{x}_t^{(l)}) \right) : \mathbf{x}_t^{(l)} \in X'_t \right\}, \quad t = 0, 1, \dots, T - 1.$$

As pointed out in Remark on Notation 6.3, in order to define the sample sets completely, we should denote them by X'_{tL_t} and Σ_{tL_t} , thus making explicit their cardinalities. Here again, however, we simply write X'_t and Σ_t .

In the backward phase of ADP, at the stage $T - 1$, we write

$$\begin{aligned} \bar{J}_{T-1}^\circ(\mathbf{x}_{T-1}^{(l)}) &= \min_{\mathbf{u}_{T-1} \in U_{T-1}(\mathbf{x}_{T-1}^{(l)})} \mathbb{E}_{\boldsymbol{\xi}_{T-1}} \left\{ h_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1}, \boldsymbol{\xi}_{T-1}) \right. \\ &\quad \left. + h_T[\mathbf{f}_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1}, \boldsymbol{\xi}_{T-1})] \right\}, \\ \mathbf{x}_{T-1}^{(l)} &\in X'_{T-1}. \end{aligned} \quad (7.18)$$

As in the solution of Problem C1' by ADP, the need of approximating the optimal cost-to-go function by FSP functions occurs from the stage $T - 2$, where the optimal FSP cost function $\tilde{J}^\circ(\mathbf{x}_{T-1}, \mathbf{w}_{T-1,n_{T-1},\Sigma_{T-1}}^\circ)$ is required. The optimal parameter vector $\mathbf{w}_{T-1,n_{T-1},\Sigma_{T-1}}^\circ$ is computed by solving the NLP problem

$$\min_{\mathbf{w}_{T-1,n_{T-1}}} \frac{1}{L_{T-1}} \sum_{l=1}^{L_{T-1}} \left[\tilde{J}_{T-1}^\circ(\mathbf{x}_{T-1}^{(l)}) - \tilde{J}(\mathbf{x}_{T-1}^{(l)}, \mathbf{w}_{T-1,n_{T-1}}) \right]^2. \quad (7.19)$$

Then, at the stage $T - 2$, the ADP equation becomes

$$\begin{aligned} \tilde{J}_{T-2}^\circ(\mathbf{x}_{T-2}^{(l)}) = & \min_{\mathbf{u}_{T-2} \in U_{T-2}(\mathbf{x}_{T-2}^{(l)})} \mathbb{E}_{\xi_{T-2}} \left\{ h_{T-2}(\mathbf{x}_{T-2}^{(l)}, \mathbf{u}_{T-2}, \xi_{T-2}) \right. \\ & \left. + \tilde{J}[\mathbf{f}_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1}, \xi_{T-1}), \mathbf{w}_{T-1,n_{T-1},\Sigma_{T-1}}^\circ] \right\}, \\ \mathbf{x}_{T-2}^{(l)} \in X'_{T-2}. \end{aligned} \quad (7.20)$$

Note that, in (7.20), we wrote $\tilde{J}_{T-2}^\circ(\mathbf{x}_{T-2}^{(l)})$ instead of $\hat{J}_{T-2}^\circ(\mathbf{x}_{T-2}^{(l)})$. This means that, for $t = T - 2, T - 3, \dots, 0$, the costs $\tilde{J}_t^\circ(\mathbf{x}_t^{(l)})$ are approximations of the “true” optimal expected costs $\hat{J}_t^\circ(\mathbf{x}_t^{(l)})$. The approximations are due to the same reasons described in deriving the deterministic ADP Eq. (6.31). We recall them (i) the sampling of the state space; (ii) the use of the FSP functions (7.17); and (iii) the impossibility of computing the minima exactly by numerical methods. Moreover, the need of computing numerically the expected values adds a new reason for inaccuracy. Owing to this fourth reason, we wrote $\tilde{J}_{T-1}^\circ(\mathbf{x}_{T-1}^{(l)})$ in (7.18) too (compare (7.19) with (6.26), where no expectation has to be performed).

The same computations that were performed at the stage $T - 2$ to derive Eq. (7.20) can be repeated at the stages $T - 3, T - 4, \dots, 0$. Summing up, the entire ADP procedure, which is the approximate version of Exact Eq. (7.13), is given by (compare with (6.31))

$$\begin{aligned} \tilde{J}_{T-1}^\circ(\mathbf{x}_{T-1}^{(l)}) = & \min_{\mathbf{u}_{T-1} \in U_{T-1}(\mathbf{x}_{T-1}^{(l)})} \mathbb{E}_{\xi_{T-1}} \left\{ h_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1}, \xi_{T-1}) \right. \\ & \left. + h_T[\mathbf{f}_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1})] \right\}, \mathbf{x}_{T-1}^{(l)} \in X'_{T-1}, \end{aligned} \quad (7.21a)$$

$$\begin{aligned} \tilde{J}_t^\circ(\mathbf{x}_t^{(l)}) = & \min_{\mathbf{u}_t \in U_t(\mathbf{x}_t^{(l)})} \mathbb{E}_{\xi_t} \left\{ h_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \xi_t) \right. \\ & \left. + \tilde{J}[\mathbf{f}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \xi_t), \mathbf{w}_{t+1,n_{t+1},\Sigma_{t+1}}^\circ] \right\}, \\ t = T - 2, T - 3, \dots, 0, \quad \mathbf{x}_t^{(l)} \in X'_t. \end{aligned} \quad (7.21b)$$

Of course, for the solution of Problem C2, one has to reduce the set X'_0 to the starting point $\mathbf{x}_0 = \hat{\mathbf{x}}$.

If the least-squares criterion is adopted, then the optimal parameter vectors $\mathbf{w}_{tn_t, \Sigma_t}^\circ$ are determined by solving the following NLP problems, which are formally similar to Problems 6.3.

Problem 7.1 For all the stages $t = T - 1, T - 2, \dots, 0$, find the optimal parameter vector $\mathbf{w}_{tn_t, \Sigma_t}^\circ$ that minimizes the quadratic error

$$\frac{1}{L_t} \sum_{l=1}^{L_t} \left[\bar{J}_t^\circ(\mathbf{x}_t^{(l)}) - \tilde{J}(\mathbf{x}_t^{(l)}, \mathbf{w}_{tn_t}) \right]^2. \quad (7.22)$$

△

Like for Problem 6.3, the possibility of replacing the least-squares criterion with that of “min-max” has to be considered.

As in Chap. 6, we point out that solving the ADP recursive equation (7.21) entails the solution of NLP problems of the following form (compare with Problems 6.4).

Problem 7.2 For all the stages $T - 2, T - 3, \dots, 0$, and any $\mathbf{x}_t^{(l)} \in X'_t$, find the optimal control vector $\mathbf{u}_t^{\circ(l)}$ that minimizes the cost

$$\tilde{J}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t) \triangleq \underset{\xi_t}{\mathbb{E}} \left\{ h_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \xi_t) + \tilde{J}[f_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \xi_t), \mathbf{w}_{t+1, n_{t+1}, \Sigma_{t+1}}^\circ] \right\} \quad (7.23)$$

subject to the constraint $\mathbf{u}_t \in U_t(\mathbf{x}_t^{(l)})$.

For the stage $T - 1$ and any $\mathbf{x}_{T-1}^{(l)} \in X_{T-1}$, find the optimal control vector $\mathbf{u}_{T-1}^{\circ(l)}$ that minimizes the cost

$$\begin{aligned} \tilde{J}_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1}) &\triangleq \underset{\xi_{T-1}}{\mathbb{E}} \left\{ h_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1}, \xi_{T-1}) \right. \\ &\quad \left. + h_T[f_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1}, \xi_{T-1})] \right\} \end{aligned} \quad (7.24)$$

subject to the constraint $\mathbf{u}_{T-1} \in U_{T-1}(\mathbf{x}_{T-1}^{(l)})$. □

We recall that the only (nontrivial) computational difference between the ADP recursive equations (6.31) and (7.13) or, equivalently, between Problems 6.4 and Problems 7.2 consists in the averaging operations to be performed (minor changes have to be inserted in the flowchart of Fig. 6.5 in order to take into account the averaging operations) with respect to the random vectors ξ_t . Note also that, in solving Problems 7.2, we have to discretize three vector domains: (i) the state admissible regions \bar{X}_t to obtain the sets X'_t (see (7.16)); (ii) for each $\mathbf{x}_t \in \bar{X}_t$, the control admissible region $U_t(\mathbf{x}_t)$ to obtain the set $U'_t(\mathbf{x}_t)$ (if NLP algorithms have not been chosen to solve Problems 7.2); and (iii) for each $\mathbf{x}_t \in \bar{X}_t$ and $\mathbf{u}_t \in U_t(\mathbf{x}_t)$ (or $\mathbf{u}_t \in U'_t(\mathbf{x}_t)$), the domain $\mathcal{E}_t(\mathbf{x}_t, \mathbf{u}_t)$. This third discretization comes from the necessity of computing expected values, i.e., integrals. Actually, analytical computation of integrals is an

impossible task in our very general context. In this case, numerical approximations are necessary and we shall address them in Sect. 7.3.2.

The discretization (ii) involves an approximate computation of the optimal control functions. As regards the discretization (i), we note that Problems 6.4 and 7.2 show the same issues for what concerns the sampling techniques that enable one to generate the sets X'_t . Moreover, the optimizations of the FSP functions $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t})$ are formally similar in both Problems 6.3 and 7.1. Summing up, what has been said in Sect. 6.6 on sampling techniques, use of FSP functions, and generalization error can be repeated here for the approximate solution of Problems C2 and C2' by DP.

7.3.2 Computation of Expected Values in ADP Recursive Equations

In the following, for simplicity, we assume again that the domains $\mathcal{E}_t(\mathbf{x}_t, \mathbf{u}_t)$ do not actually depend on \mathbf{x}_t and \mathbf{u}_t ; hence, we denote them simply as \mathcal{E}_t . Similarly, we assume that the probability densities of the random disturbances ξ_t do not depend on \mathbf{x}_t and \mathbf{u}_t ; hence, we denote them as $p_t(\xi_t)$. In order to solve Problem 7.2 for each decision stage at any discretized state $\mathbf{x}_t^{(l)}$, $l = 1, \dots, L_t$, we have to minimize the expected cost

$$\tilde{\mathcal{J}}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t) = \int_{\mathcal{E}_t} \mathcal{C}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \xi_t) p_t(\xi_t) d\xi_t \quad (7.25)$$

with respect to \mathbf{u}_t , where

$$\begin{aligned} \mathcal{C}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \xi_t) &\triangleq \left\{ h_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \xi_t) + \tilde{J}[f_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \xi_t), \mathbf{w}_{t+1, n_{t+1}, \Sigma_{t+1}}^\circ] \right\}, \\ t &= 0, 1, \dots, T-2. \end{aligned} \quad (7.26)$$

A similar definition for $\mathcal{C}_{T-1}(\mathbf{x}_{T-1}^{(l)}, \mathbf{u}_{T-1}, \xi_{T-1})$ can be derived from (7.24).

We said previously that computing analytically the expected costs (7.25) is an impossible task since no particular assumptions were made about the integrands (7.26) and the probability densities. Then, we resort to the numerical computation of integrals following the methods presented in Sects. 5.1 and 5.2. To this end, consistently with the introduction of the grids X'_t , we generate a sequence of grids that discretize the domains \mathcal{E}_t . Since we assume that these domains do not depend on \mathbf{x}_t and \mathbf{u}_t , we can write

$$\mathcal{E}'_t \triangleq \left\{ \xi_t^{(l)} \in \mathcal{E}_t : l = 1, \dots, L''_t \right\}, \quad t = 0, 1, \dots, T-1. \quad (7.27)$$

We consider three kinds of grids of \mathcal{E}_t , namely, (i) regular grids, (ii) grids generated by MC techniques, and (iii) grids generated by quasi-MC methods.

1. Expected Values Computed by Regular Grids

As pointed out in Sect. 5.1.1, integration of the product $\mathcal{C}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \boldsymbol{\xi}_t) p_t(\boldsymbol{\xi}_t)$ by using regular grids is to be avoided unless the dimension q of $\boldsymbol{\xi}_t$ is suitably small. Indeed, let us assume, for simplicity, that $\Xi_t = [0, 1]^q$ and that each edge of such a cube has been partitioned into m intervals of equal length. Then, the integrand must be evaluated in $L''_t = (m+1)^q$ points and, for a given level of the numerical integration error, L''_t increases exponentially with q . This gives rise to a *third curse of dimensionality*.

Remark 7.2 For the reader's convenience, we simply call the curses of dimensionality due to the regular grids in the state, control (if the vectors \mathbf{u}_t are discretized), and noise spaces the "first", "second", and "third" curse of dimensionality, respectively. We should also take into account that, in approximating the (approximate) optimal costs-to-go \tilde{J}_t° by the FSP functions $\tilde{J}(\cdot, \mathbf{w}_{tn_t})$ (see Problems 6.3 and 7.1), the danger of a fourth curse of dimensionality is incurred. Indeed, as we have repeatedly pointed out, if the FSP approximating functions $\tilde{J}(\cdot, \mathbf{w}_{tn_t})$ are not properly chosen, for a given accuracy, the rate of growth of $\dim(\mathbf{w}_{tn_t})$ with $d = \dim(\mathbf{x}_t)$ may be too fast. \triangleleft

2. Expected Values Computed by MC Techniques

In order to compute the expected value of the function \mathcal{C}_t for a given stage t and fixed values of $\mathbf{x}_t^{(l)}$ and \mathbf{u}_t , we generate the nodes of the grid Ξ'_t by using a random procedure. As explained in Sect. 5.1.2 about the MC approximate computations of integrals, this means taking L''_t random samples $\boldsymbol{\xi}_t^{(1)}, \dots, \boldsymbol{\xi}_t^{(L''_t)}$ according to the probability density $p_t(\boldsymbol{\xi}_t)$. Then, we approximate the expected value $E[\mathcal{C}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \boldsymbol{\xi}_t)]$ by the MC estimate (see (5.10))

$$\frac{1}{L''_t} \sum_{i=1}^{L''_t} \mathcal{C}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \boldsymbol{\xi}_t^{(i)}), \quad t = 0, 1, \dots, T-1. \quad (7.28)$$

Now, for each given $\mathbf{x}_t^{(l)}$ and \mathbf{u}_t , we assume that $\mathcal{C}_t \in \mathcal{L}_2$ with respect to the probability density of $\boldsymbol{\xi}_t$ and we evaluate the probabilistic error of the MC estimate (7.28) by using the variance of \mathcal{C}_t , that is (see (5.12)),

$$\begin{aligned} \sigma^2[\mathcal{C}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \boldsymbol{\xi}_t)] &= \int_{\Xi_t} \{\mathcal{C}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \boldsymbol{\xi}_t) - E[\mathcal{C}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \boldsymbol{\xi}_t)]\}^2 p_t(\boldsymbol{\xi}_t) d\boldsymbol{\xi}_t, \\ &t = 0, 1, \dots, T-1. \end{aligned} \quad (7.29)$$

From Theorem 5.1 we obtain

$$\begin{aligned} & \int_{\Xi_t} \cdots \int_{\Xi_t} \left[\frac{1}{L''_t} \sum_{i=1}^{L''_t} \mathcal{C}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \boldsymbol{\xi}_t^{(i)}) - \mathbb{E}(\mathcal{C}_t) \right]^2 \\ & \quad \times p_t(\boldsymbol{\xi}_t^{(1)}) \cdots p_t(\boldsymbol{\xi}_t^{(L''_t)}) d\boldsymbol{\xi}_t^{(1)} \cdots d\boldsymbol{\xi}_t^{(L''_t)} = \frac{\sigma^2[\mathcal{C}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \boldsymbol{\xi}_t)]}{L''_t}, \\ & \quad \forall \mathbf{u}_t \in U_t(\mathbf{x}_t^{(l)}), l = 1, \dots, L''_t, t = 0, 1, \dots, T-1. \end{aligned} \quad (7.30)$$

The result (7.30) states that the error of the MC estimate of the expected value in the ADP Eq. (7.21) (or in (7.25)) decreases as $O(1/\sqrt{L''_t})$ when the number of nodes of the grid Ξ'_t increases. Note that this rate of decrease is independent of the dimension of the vector $\boldsymbol{\xi}_t$. Then, the MC estimate of the expected value in ADP should not incur the curse of dimensionality as is the case with the regular grids.

The foregoing enhances a very important property achieved by the MC estimates of the expected costs. However, we must not forget the drawbacks that affect the MC method, namely, (i) the errors have a probabilistic nature, (ii) the regularity properties of the functions to be integrated are not taken into account, and (iii) generating random samples (i.e., the discretized values $\boldsymbol{\xi}_t^{(i)}$) may be difficult (see Remark 5.1).

3. Expected Values Computed by Quasi-MC Techniques

To estimate the expected values $\mathbb{E}[\mathcal{C}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \boldsymbol{\xi}_t)], t = 0, 1, \dots, T-1$, by using quasi-MC techniques, we refer to Sect. 5.2. First of all, we assume that each domain Ξ_t is such that it can be reduced to the unit cube $[0, 1]^q$. According to (5.22), the quasi-MC estimates of Costs (7.23) and (7.25) are given by

$$\begin{aligned} \tilde{\mathcal{J}}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t) &= \mathbb{E}[\mathcal{C}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t, \boldsymbol{\xi}_t)] \\ &\simeq \frac{1}{L''_t} \sum_{i=1}^{L''_t} \mathcal{C}_t[\mathbf{x}_t^{(l)}, \mathbf{u}_t, \boldsymbol{\xi}_t^{(i)}] p_t(\boldsymbol{\xi}_t^{(i)}), \quad t = 0, 1, \dots, T-1, \end{aligned} \quad (7.31)$$

where $\boldsymbol{\xi}_t^{(1)}, \dots, \boldsymbol{\xi}_t^{(L''_t)} \in [0, 1]^q$ form sequences (define them as $\{\boldsymbol{\xi}_t^{L''_t}\}$) generated by a deterministic algorithm. Now, in a similar way to Assumption 5.1, we suppose the probability densities $p_t(\boldsymbol{\xi}_t)$ and the functions \mathcal{C}_t to belong to $\mathcal{W}_{M_t}([0, 1]^q)$, where M_t are positive finite scalars. Since \mathcal{C}_t are given by the composition of the functions h_t , f_t , and $\tilde{J}(\cdot, \mathbf{w}_{t+1, n_{t+1}, \Sigma_{t+1}}^\circ)$ (consider them as dependent merely on $\boldsymbol{\xi}_t$ for given values of \mathbf{x}_t and \mathbf{u}_t), suitable smoothness assumptions on them ensure that the functions \mathcal{C}_t belong to $\mathcal{W}_{M_t}([0, 1]^q)$ (this by virtue of Lemma 4.1). The same applies for the products $\mathcal{C}_t \cdot p_t$. As a consequence, they have bounded variations $V_{HK}(\mathcal{C}_t \cdot p_t)$ in the sense of Hardy and Krause. Inequalities similar to (5.23) follow, i.e.,

$$\left| \sum_{i=1}^{L''_t} \mathcal{C}_t[\mathbf{x}_t^{(l)}, \mathbf{u}_t, \boldsymbol{\xi}_t^{(i)}] p_t(\boldsymbol{\xi}_t^{(i)}) - \mathbb{E}(\mathcal{C}_t) \right| \leq V_{HK}(\mathcal{C}_t \cdot p_t) D_L^*(\{\boldsymbol{\xi}_t^{L''_t}\}),$$

$$t = 0, 1, \dots, T-1.$$

Therefore, if $\{\xi_t^{L''}\}$ are low-discrepancy sequences, the quasi-MC estimates (7.31) have errors with respect to $E(\mathcal{C}_t)$ that decrease with rate of convergence of order $O(1/L_t'')$ instead of $O(1/\sqrt{L_t''})$ as is in the case with MC estimates (7.28).

We should repeat here what we said at the end of Sect. 5.2.1 about the possible inefficiency of the above-described deterministic technique based on Estimate (5.22). We refer to the comments reported in Sect. 5.2.1 and to the possible improvements proposed in Sect. 5.2.2.

Remark 7.3 Even in the solution of Problems C2 and C2' by ADP there is the possibility of using kernel smoothing models (KSMs) as FSP functions that approximate the optimal costs-to-go in Eqs. (7.21). All the considerations reported in Sect. 6.4.4 can be repeated here replacing the costs-to-go dependent explicitly on ξ_t with their expected values. So, Costs (6.40) are replaced by (see (7.23))

$$\tilde{\mathcal{J}}_t(\mathbf{x}^{(l)}, \mathbf{u}_t) \triangleq E_{\xi_t} \left\{ h_t(\mathbf{x}^{(l)}, \mathbf{u}_t, \xi_t) + \tilde{J}_{t+1}[\mathbf{f}_t(\mathbf{x}^{(l)}, \mathbf{u}_t^{(l)}, \xi_t), \alpha_{t+1}^\circ, \Sigma_{t+1}] \right\},$$

which are minimized with respect to \mathbf{u}_t , thus giving $\mathbf{u}_t^{\circ(l)} = \bar{\mu}_t^\circ(\mathbf{x}_t^{(l)})$, $l = 1, \dots, L_t$ (see the solutions of Problem 7.2).

If one applies cross-validation, similar considerations can be made. More details on the use of KSMs in ADP can be found in [21, 23, 24]. \diamond

7.3.3 Computation of the Optimal Controls in the Forward Phase of Dynamic Programming

The approximate optimal cost-to-go functions $\tilde{J}_t^\circ(\mathbf{x}_t^{(l)})$ and the FSP optimal cost-to-go functions $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{m_t}^\circ)$ can be obtained *offline* in the backward phase of ADP. As regards the computation of the optimal controls, we can repeat similar considerations to the ones provided in Sect. 6.5, with the additional need of performing averaging operations. Of course, because of the stochastic nature of Problem C2', it is not possible to predict the optimal controls $\mathbf{u}_0^\circ, \dots, \mathbf{u}_{T-1}^\circ$ at the beginning of the decision process. Actually, they have to be computed *online* on the basis of the measurements taken on \mathbf{x}_t stage after stage (hence, on the basis of the realizations of the noises). As in the backward phase of the deterministic ADP Eq. (6.31), we obtain approximate optimal controls similar to (6.38). We rewrite them as:

$$\mathbf{u}_t^{\circ(l)} = \bar{\mu}_t^\circ(\mathbf{x}_t^{(l)}), \quad \mathbf{x}_t^{(l)} \in X'_t, \quad t = 0, 1, \dots, T-1. \quad (7.32)$$

Clearly, $X'_0 = \{\hat{\mathbf{x}}\}$ for Problem C2, whereas X'_0 is derived by the discretization of X_0 for Problem C2'.

We recall that, during the *online* decision process, the state trajectory is not made of a sequence of discretized vectors $\mathbf{x}_t^{(l)} \in X'_t$, $t = 0, 1, \dots, T-1$. More specifically, at stage $t = 0$, the dynamic system is generally not in a discretized state $\mathbf{x}_0^{(l)} \in X'_0$.

Likewise, at the stages $t > 0$, even if the system is in a discretized state $\mathbf{x}_t^{(l)} \in X'_t$, the control $\mathbf{u}_t^\circ = \bar{\mu}_t^\circ(\mathbf{x}_t^{(l)})$ and the random noise ξ_t do not generally drive the state to another discretized state $\mathbf{x}_{t+1}^{(l')} \in X'_{t+1}$. Therefore, approximations to obtain the optimal control vector $\mathbf{u}_t^\circ = \bar{\mu}_t^\circ(\mathbf{x}_t)$ are needed.

The simplest approximation consists in giving the functions $\bar{\mu}_t^\circ(\mathbf{x}_t)$ the value associated with the discretized state $\mathbf{x}_t^{(l'')} \in X'_t$ nearest to \mathbf{x}_t . In the following, we shall describe possibly better approximations based on FSP functions. The two possibilities we address correspond (in a shorter form) to those reported in the next two sections.

1. Online Reoptimization

The term “reoptimization” means that, once the initial state $\mathbf{x}_0 \in X_0$ becomes known to the DM or once the state \mathbf{x}_t , $t = 1, \dots, T - 1$, has been reached, the approximate optimal controls are derived by solving the optimization problems given by

$$\mathbf{u}_t^\circ = \underset{\mathbf{u}_t \in U_t(\mathbf{x}_t^\circ)}{\operatorname{argmin}} \underset{\xi_t}{\mathbb{E}} \left\{ h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t) + \tilde{J} [f_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t), \mathbf{w}_{t+1, n_{t+1}, \Sigma_{t+1}}^\circ] \right\}, \\ t = 0, 1, \dots, T - 1. \quad (7.33)$$

As regards the minimizations in (7.33), we can repeat what we said in Sect. 6.5 about Minimizations (6.49). Of course, the computations are now more time-demanding since both minimizations and averaging operations in (7.33) must be performed online.

Since in Problem C2 the initial state $\mathbf{x}_0 = \hat{\mathbf{x}}$ is one of the a priori data (i.e., it is known before the decision process starts), \mathbf{u}_0° can be computed offline (that is, before the instant $t = 0$). In Problem C2' the DM knows a priori the set X_0 but only observes the state $\mathbf{x}_0 \in X_0$ at time $t = 0$. Thus, \mathbf{u}_0° can only be computed at this instant, that is, online via (7.33) as, in general, \mathbf{x}_0 does not coincide with any discretized vector belonging to X'_0 .

2. Offline Approximation of the Closed-Loop Optimal Control Functions by FSP Functions and a Least-Squares Criterion

Following the general lines illustrated in Sect. 6.5, we introduce a second family of FSP functions to approximate the optimal closed-loop control functions $\bar{\mu}_t^\circ$. Such functions are given by (see (6.50))

$$\mathbf{u}_t = \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t}^u), \quad t = 0, 1, \dots, T - 1, \quad (7.34)$$

where $\mathbf{w}_{tn_t}^u$ are vectors of parameters to be optimized. Of course, if we consider Problem C2, no FSP function is necessary at the stage $t = 0$. As for the functions (6.50), we assume that the type of networks (7.34) remains time-invariant during the decision process.

The vectors $\mathbf{w}_{tn_t}^u$ can be optimized in the same way as in Sect. 6.5. Once the vectors \mathbf{u}_t° have been computed for each discretized state $\mathbf{x}_t^{(l)} \in X'_t$ in the backward

phase of the ADP procedure (see (7.32)), we define I/O sample sets Σ_t^u identical to Sets (6.51). Then, we have to solve the following NLP problems (compare with Problems 6.5).

Problem 7.3 For $t = 0, 1, \dots, T - 1$, find the optimal parameter vector $\mathbf{w}_{tn, \Sigma_t^u}^{u^\circ}$ that minimizes the quadratic cost

$$\sum_{l=1}^{L_t} \left| \tilde{\mu}^\circ(\mathbf{x}_t^{(l)}) - \tilde{\mu}(\mathbf{x}_t^{(l)}, \mathbf{w}_{tn, \Sigma_t^u}^u) \right|^2. \quad (7.35)$$

△

Then, we obtain the FSP optimal functions

$$\mathbf{u}_t^\circ = \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn, \Sigma_t^u}^{u^\circ}), \quad t = 0, 1, \dots, T - 1. \quad (7.36)$$

Control Functions (6.50) and (7.36) are formally identical. Of course, (6.50) constitutes an intermediate step to obtain the open-loop optimal solution given by the control vectors $\mathbf{u}_0^\circ, \mathbf{u}_1^\circ, \dots, \mathbf{u}_{T-1}^\circ$ in the forward phase of ADP, while Functions (7.36) – in their closed-loop form – are the FSP optimal solution of the stochastic Problem C2'.

As in the solutions of Problems C1 and C1', it is hard to compare the reoptimization with the approximation of the optimal control functions by FSP functions as regards the accuracy achieved. Reoptimization involves the use of only one sequence of FSP functions and perhaps it might perform better.

An important difference between the two procedures is given by the computational load. In a sense, reoptimization splits this burden into offline and online computations. This may lead to heavy online computational problems. In fact, at each decision stage, reoptimization requires both an averaging operation and the approximate computation of the optimal control \mathbf{u}_t° . The DM may not have the computational power to accomplish such tasks online if the process dynamics were too fast. By contrast, the computation of the function (7.36) is performed in the offline phase. Once such control functions have been determined, the DMs store them and generate the optimal control vectors almost instantaneously stage after stage. This may turn out to be an important advantage.

7.3.4 Error Propagation of the Optimal Cost-to-Go Functions Starting from Approximations of the Optimal Control Functions

Starting from the seminal works by Bellman (see, for example, [4]), there is a vast literature on the error propagation in DP and ADP (see also [15] and the recent works [33, 35–37]). In this context, it is important to establish conditions under

which suitable regularity properties of the optimal cost-to-go and control functions hold (e.g., continuity, belonging to the spaces considered in Chap. 3 [36]), since such properties can help in the choice of a suitable family of approximating functions. Such issues, however, require a treatment that goes beyond the scope of this book. In this section, we shall limit ourselves to developing a brief analysis of the error propagation in the approximation of the optimal costs-to-go in discrete-time stochastic optimal control problems with finite horizon T , stated in the form of Problem C2'. Here, the optimal cost-to-go functions are approximated using approximations of the optimal control functions (which may be implemented, e.g., by FSP functions in the case considered by the ERIM). Even if we only consider the case of stochastic optimal control problems, a similar analysis can be developed for the case of discrete-time deterministic optimal control problems with finite horizon T , stated in the form of Problem C1' (not reported here, in order to avoid redundancy; see, however, some comments in Sect. 6.4.3).

The results provided in this section are reported in [42]. In the following, we denote the approximations of the optimal control functions $\mu_0^\circ, \dots, \mu_{T-1}^\circ$ for Problem C2' by $\tilde{\mu}_0, \dots, \tilde{\mu}_{T-1}$ (expressed, e.g., by FSP functions). Using such approximate control functions, we now consider approximations of the optimal cost-to-go functions of the form

$$\tilde{J}_t(\mathbf{x}_t) = \underset{\xi_t, \dots, \xi_{T-1}}{\mathbb{E}} \left\{ \sum_{k=t}^{T-1} h_k[\mathbf{x}_k, \tilde{\mu}_k(\mathbf{x}_k), \xi_k] + h_T(\mathbf{x}_T) \right\},$$

for $t = 0, \dots, T - 1$.

Now, for a bounded and continuous function $s_t : X_t \rightarrow \mathbb{R}^l$, $s_t = \text{col}(s_{t1}, \dots, s_{tl})$, $l \in \mathbb{Z}^+$, we let

$$\|s_t\|_{\sup(X_t)} \triangleq \sup_{\mathbf{x}_t \in X_t} |s_t(\mathbf{x}_t)| \triangleq \sup_{\mathbf{x}_t \in X_t} \sqrt{\sum_{j=1}^l s_{tj}^2(\mathbf{x}_t)}.$$

Suppose that $U_t(\mathbf{x}_t)$ is independent of \mathbf{x}_t and that $\Xi_t(\mathbf{x}_t, \mathbf{u}_t)$ is independent of \mathbf{u}_t and \mathbf{x}_t . Hence, they are denoted, respectively, by U_t and Ξ_t . The following result holds (see [42, Theorem 3.1]).

Theorem 7.1 *For $t = 0, \dots, T - 1$, suppose that there exist optimal control functions μ_t° . Let f_t , h_t , h_T , and μ_t° be bounded and Lipschitz-continuous with Lipschitz constants bounded from above by L_{f_t} , L_{h_t} , L_{h_T} , and $L_{\mu_t^\circ}$, respectively. Let $\tilde{\mu}_t : X_t \rightarrow U_t$ be bounded and continuous approximating control functions such that $\|\mu_t^\circ - \tilde{\mu}_t\|_{\sup(X_t)} \leq \epsilon_t$ for some $\epsilon_t \geq 0$. Then, for each $\mathbf{x}_t \in X_t$, \tilde{J}_t is continuous with respect to $\mu_0^\circ, \dots, \mu_{T-1}^\circ$ in the supremum norm and*

$$\|\hat{J}_t^\circ - \tilde{J}_t\|_{\sup(X_t)} \leq \sum_{k=t}^{T-1} \Gamma_k \epsilon_k, \quad (7.37)$$

where, for $k = t, \dots, T - 1$, $\Theta_{kk} \triangleq 0$, $\Theta_{k+1,k} \triangleq L_{f_k}$, $\Theta_{k+i,k} \triangleq [\prod_{j=k+1}^{k+i-1} (1 + L_{\mu_j^o}) L_{f_j}] L_{f_k}$, $i = 2, \dots, T - k$, and $\Gamma_k \triangleq [\sum_{j=k}^{T-1} L_{h_j} ((1 + L_{\mu_k^o}) \Theta_{jk} + \delta_{jk})] + L_{h_T}$. Θ_{Tk} (δ_{jk} is the Kronecker's delta). \square

In the sequel, we also denote the approximations $\tilde{J}_t(\mathbf{x}_t)$ by

$$\tilde{J}_t(\mathbf{x}_t, \tilde{\mu}_t, \dots, \tilde{\mu}_{T-1}),$$

to emphasize their dependence on $\tilde{\mu}_t, \dots, \tilde{\mu}_{T-1}$. Of course, when considering the optimal control functions $\mu_t^o, \dots, \mu_{T-1}^o$, one has

$$\tilde{J}_t(\mathbf{x}_t, \mu_t^o, \dots, \mu_{N-1}^o) = \hat{J}_t^o(\mathbf{x}_t).$$

Then, Theorem 7.1 shows that, for each $\mathbf{x}_t \in X_t$, the cost-to-go functional $\tilde{J}_t(\mathbf{x}_t, \tilde{\mu}_t, \dots, \tilde{\mu}_{T-1})$ is continuous with respect to $\tilde{\mu}_t, \dots, \tilde{\mu}_{T-1}$ when the supremum norm is used to measure the approximation errors of the optimal control functions. One can also easily show that a similar result cannot be obtained, in general, if the \mathcal{L}_2 -norm is used for the same purpose (unless the sets X_t are compact, and suitable smoothness constraints are made on the family of approximators). It is important to recall that, under certain assumptions, the optimal control functions $\mu_t^o, \dots, \mu_{T-1}^o$ have a sufficiently large degree of smoothness so as to be approximated by FSP functions belonging to polynomially complex approximating sequences of sets in various norms (for instance, the \mathcal{L}_2 -norm and the supremum norm). See, for instance, [36, 41] for some of these smoothness results, obtained under both deterministic and stochastic models that have some similarities with the one considered in this section. In this case, supposing (for simplicity of the notation) that the model complexities of the FSP functions, used at each stage $k = t, \dots, T - 1$, are equal to the same positive integer n , and that the constants c, p, q of the FSP functions² (see Sect. 2.7.2) are the same for all the stages, one can take $\epsilon_k = c \frac{d^p}{n^q}$ in the statement of Theorem 7.1.

Then, setting $\varepsilon = \sum_{k=t}^{T-1} \Gamma_k \epsilon_k$, one gets

$$\|\hat{J}_t^o - \tilde{J}_t\|_{\sup(X_t)} \leq \varepsilon \quad (7.38)$$

when the model complexity n is larger than or equal to

$$n^o(d, \varepsilon, t) = \left\lceil \left(\frac{c \sum_{k=t}^{T-1} \Gamma_k}{\varepsilon} \right)^{1/q} d^{p/q} \right\rceil.$$

Finally, one can also observe that the approximation (7.38) of the optimal cost-to-go \hat{J}_t^o is obtained indirectly through the vector-valued FSP functions $\tilde{\mu}_t, \dots, \tilde{\mu}_{T-1}$, while in ADP a single scalar-valued FSP function is used to approximate \hat{J}_t^o directly.

²Here, q does not refer to the dimensions of the random disturbances.

Theorem 7.2 (formulated for simplicity only for the stage $t = T - 1$) follows directly from [42, Theorem 3.3]. It is shown therein that (under suitable assumptions specified in [42]) if one approximately minimizes $\tilde{J}_{T-1}(\mathbf{x}_{T-1}, \tilde{\mu}_{T-1})$ for each $\mathbf{x}_{T-1} \in X_{T-1}$, then the resulting approximate minimizer $\tilde{\mu}_{T-1}^\circ$ is a good approximation of the optimal control function μ_{T-1}° . Before stating the theorem, we introduce the following notation. For a function $g(z)$ of a vector argument z , we denote by $\lambda_{\min}(Hg(z))$ the minimum eigenvalue of the Hessian matrix $Hg(z)$ of $g(z)$. For a function $g(z_1, \dots, z_M)$ of M vector arguments z_1, \dots, z_M , we denote by $H_{jj}(g(z_1, \dots, z_j, \dots, z_M))$ the square submatrix of $Hg(z_1, \dots, z_M)$ that is associated with the vector argument z_j . Finally, $\text{int}(U_{T-1})$ is the interior of the set $U_{T-1} \subseteq \mathbb{R}^m$.

Theorem 7.2 *Let X_{T-1} , X_T , and Ξ_{T-1} be compact with non-empty interiors, U_{T-1} be compact and convex and with non-empty interior, and X_{T-1} and X_T be convex. Suppose that a continuous optimal control function μ_{T-1}° exists, and that for each $\mathbf{x}_{T-1} \in X_{T-1}$ one has $\mu_{T-1}^\circ(\mathbf{x}_{T-1}) \in \text{int}(U_{T-1})$. Let J_T° and h_{T-1} be of classes $\mathcal{C}^2(X_T)$ and $\mathcal{C}^2(X_{T-1} \times U_{T-1} \times \Xi_{T-1})$, respectively. Let the probability density $p_{T-1}(\xi_{T-1})$ be of class $\mathcal{C}^2(\Xi_{T-1})$ and $h_{T-1}(\cdot, \cdot, \xi_{T-1})$ be convex for each ξ_{T-1} . Suppose that there exists $\tau_T > 0$ for which one has*

$$\inf_{\mathbf{x}_T \in X_T} \lambda_{\min}(H J^\circ(\mathbf{x}_T)) \geq \tau_T, \quad (7.39)$$

and $\tau_{T-1} > 0$ for which, for each $\mathbf{x}_{T-1} \in X_{T-1}$ and $\xi_{T-1} \in \Xi_{T-1}$ one has

$$\inf_{\mathbf{u}_{T-1} \in U_{T-1}} \lambda_{\min}(H_{22} h_{T-1}(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}, \xi_{T-1})) \geq \tau_{T-1}, \quad (7.40)$$

where the subscript 2 in the Hessian refers to the vector \mathbf{u}_{T-1} . Finally, suppose that f_{T-1} is of class $\mathcal{C}^2(X_{T-1} \times U_{T-1} \times \Xi_{T-1})$ and is affine in the control, i.e., it has the form

$$\begin{aligned} f_{T-1}(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}, \xi_{T-1}) \\ = A_{T-1}(\mathbf{x}_{T-1}, \xi_{T-1}) \mathbf{u}_{T-1} + b_{T-1}(\mathbf{x}_{T-1}, \xi_{T-1}) \end{aligned} \quad (7.41)$$

(where $A_{T-1}(\mathbf{x}_{T-1}, \xi_{T-1})$ and $b_{T-1}(\mathbf{x}_{T-1}, \xi_{T-1})$ are $d \times m$ and $d \times 1$ matrices, respectively, which are affine in the state \mathbf{x}_t), and that there exists $\eta_{T-1} > 0$ such that

$$\inf_{\mathbf{x}_{T-1} \in X_{T-1}, \xi_{T-1} \in \Xi_{T-1}} \lambda_{\min}(A_{T-1}^\top(\mathbf{x}_{T-1}, \xi_{T-1}) A_{T-1}(\mathbf{x}_{T-1}, \xi_{T-1})) \geq \eta_{T-1}. \quad (7.42)$$

Then, for every $\varepsilon > 0$, one has the following implication:

$$\begin{aligned} & \| \tilde{J}_{T-1}(\cdot, \mu_{T-1}^\circ) - \tilde{J}_{T-1}(\cdot, \tilde{\mu}_{T-1}^\circ) \|_{\sup(X_{T-1})} \leq \varepsilon \\ & \Rightarrow \| \mu_{T-1}^\circ - \tilde{\mu}_{T-1}^\circ \|_{\sup(X_{T-1})} \leq \sqrt{\frac{2\varepsilon}{\tau_{T-1} + \tau_T \eta_{T-1}}} \end{aligned} \quad (7.43)$$

□

Note that, in Theorem 7.2, (7.39) and (7.40) are assumptions of strong convexity. The previous result can be extended with some modifications to the approximations of the optimal control functions at the stages $t = T - 2, T - 3, \dots, 0$, while taking fixed at each stage t the approximations $\hat{\mu}_{t+1}^\circ, \dots, \hat{\mu}_{T-1}^\circ$ of the optimal control functions $\mu_{t+1}^\circ, \dots, \mu_{T-1}^\circ$, which were previously derived for the subsequent stages $t + 1, \dots, T - 1$. Concepts somehow similar to the ones considered in this section were investigated in [41] for a deterministic model expressed in terms of correspondences instead of dynamic systems. These two formulations are equivalent under certain circumstances (see, for instance [41, Sect. 6], for an example of this equivalence).

7.4 Some Notes on Approximate Dynamic Programming

Chapter 6 and the previous sections of this chapter are devoted mainly to ADP. As in the remainder of this chapter we shall address the solution of Problems C2 and C2' leaving DP and passing to the ERIM, at this point we deem it suitable to cite some works that we consider directly related to our treatment of ADP. Needless to say, the list of works cited will be by no means complete. Moreover, we shall limit ourselves to dynamic systems with continuous-state, control, and disturbance variables.

This is the scope of Sects. 7.4.1 and 7.4.2, where we report two very short surveys on discretization and approximation issues in ADP, respectively, for optimal control problems stated over a finite horizon. The preservation of some structural properties of the optimal cost-to-go and control functions through the stages of the DP procedure is considered in Sect. 7.4.3. In Sect. 7.4.4, we illustrate some elementary concepts on *reinforcement learning* (RL) and, in Sect. 7.4.5, on *Q-learning* (a particular RL technique) referring to the infinite-horizon case. The right place in which to illustrate these two latter arguments would be Chap. 10. However, they are laid out in this chapter because they are closely related with DP. It is worth noting that RL and Q-learning put together concepts of adaptive control and optimal control. Adaptive control is far beyond the scope of this book where any control law can be computed a priori or offline. Yet, to completely ignore the connections between adaptive control and optimal control through DP would have been misleading. This is the reason why we have a dedicated part of this section to RL and Q-learning. For insights, the interested reader is referred, for example, to the thorough treatment in [10, Chap. 6].

7.4.1 Discretization Issues

In Remark 7.2, we pointed out that there are various origins of curses of dimensionality. However, the discretization of the state vector is, in general, the source of the most serious shortcoming of ADP, i.e., the curse of dimensionality due to the growth of the sample cardinality necessary to guarantee a given accuracy when the dimension

d of the state space increases. Here, we mention a few historical approaches to discretization addressed in Sect. 6.4.1: (i) the use of full uniform grids, (ii) the discretization of the sets \bar{X}_t (i.e., generation of the sample sets X'_t) by random and deterministic algorithms (MC, quasi-MC, etc.), and (iii) other methods, sometimes of heuristic nature, aimed at avoiding the disadvantages of regular grids.

The first discretization method introduced by Bellman [4] is the full uniform grid, which implies an exponential number q^d of discretized points (denoting by q , in this section, the number of equally spaced points in which each component of the state vector has been discretized). In spite of this drawback, for a long time the full uniform grid has been considered as the only discretization method for the numerical solution of the DP equations. From a historical point of view, it seems interesting to quote the following from [6, p. 248] (we have replaced the notation “ N ” used therein with our notation “ d ”): “In terms of the capacities of modern computers, we have an extremely efficient algorithm if $d = 1$, a scalar problem, and a feasible algorithm if $d = 2$. If $d = 3$ or more, we face fast memory difficulties if we attempt to proceed in a routine fashion.” In [7], approximation techniques were proposed.

An approach aimed at reducing the computational requirements was introduced by Larson [53]. Starting from continuous-time optimal control problems, the state components were still discretized therein by equally spaced intervals but with different lengths of the time intervals between two subsequent decision instants. This is due to the fact that a new control vector was not applied when time changes by a constant increment, as in conventional DP, but when one of the state components changes by a specific increment. Hence, the name *state increment dynamic programming*.

Among other efforts to mitigate the curse of dimensionality in DP, we cite the aggregation/disaggregation of grid points to diminish the number of discretized points and the partition of the original problem into smaller separable problems [1, 82, 90]. The major inconvenience of such a method lies in its lack of generality, since it is highly dependent on the particular problem it is applied to (in the aforementioned works, reservoir operation).

Regular grids are an obviously attractive choice for ADP, thanks to their straightforward implementation and structural guarantee of uniformity. Yet, the fact that its number of points has an exponential figure has stimulated the investigation of alternatives, hoping to achieve similar ease of implementation and uniformity without such an unfeasible growth of the number of discretization points, possibly allowing the freedom of choosing any desired number. Perhaps the simplest option is to use i.i.d. sequences with the desired number of points drawn according to a uniform distribution. Such techniques are implemented by many popular software packages. However, i.i.d. random generators often yield sets of points presenting clusters that undermine uniformity of sampling.

Consequently, a more attractive option is the use of sampling schemes that aim at ensuring uniformity in a deterministic way. Again, the possibilities are many. In [28] the use of deterministic sets known as *orthogonal arrays* (OAs) was first introduced for the solution of a high-dimensional optimal control problem. OAs are subsets of the full regular grid chosen so that each dimension of the state space is represented in a balanced way. In particular, if we consider a subset of k components

of the state, each of the possible q^k combinations of these k components occurs λ times in the OA (recall that q is the number of discretized points of each state component). The parameter k is called the *strength* of the OA. Then, an OA of strength k is characterized by a number of points equal to λq^k . This polynomial growth is obviously computationally much more manageable than the exponential growth of the full regular grid. In fact, by letting $\lambda = 1$, $k = 3$ and $q \simeq d$, an OA scheme is obtained with a number of points growing as $O(d^3)$. This approach, combined with the class of nonlinear approximators called *multivariate adaptive regression splines*, has allowed the solution of a nine-dimensional inventory-forecasting problem for the first time.

In [80], variants of the OAs, namely, *OA-based Latin hypercubes* (OA-LHs), where a Latin Hypercube design (i.e., an OA of strength 1) is constructed by randomization starting from an OA of higher strength, were used for the solution of a 20-dimensional wastewater treatment problem. In [26], OAs proved to yield satisfying results for the solution of a nine-dimensional inventory problem also when used with neural networks as approximators. A review of OAs, OA-LHs, and other related sampling schemes can be found in [29].

Another kind of deterministic sampling scheme, which has been used in its application to ADP, is based on the use of the *low-discrepancy sequences* described in Chap. 4. This kind of discretization satisfies both requirements mentioned earlier, since the resulting sequences are designed to be spread uniformly over the state space for any desired number of sample points. The use of low-discrepancy schemes such as *(t-d)-sequences* and *Sobol' sequences* was investigated in the context of both finite- and infinite-horizon DP in [25] where, for the simulation tests, a reservoir network optimal management problem with 10 basins and proper dynamics of the random inflows, resulting in a 30-dimensional state vector, was solved by means of neural networks as approximators. In [19], the 30-dimensional reservoir network problem was solved by using both OAs and low-discrepancy sequences with neural networks. Both sampling schemes turned out to perform very similarly on average over the various tests. Low-discrepancy sequences and neural networks were also employed for the DP solution of the seven-dimensional reservoir network problem used in [3], where the DP approach was compared with the ERIM. Lattice point sets, a special kind of low-discrepancy sampling, were also considered [22]. Despite the little differences in performance from case to case given by the various deterministic discretizations and approximating models used for the approximation of the optimal cost-to-go functions, the dimensions of the problems mentioned above show that, in general, the use of a smart deterministic sampling scheme, coupled with some flexible nonlinear approximator, allows one to cope with problems of type of Problem C2 that are impossible to deal with while using “classic” DP.

It has to be remarked that quasi-MC methods have also been proposed in the stochastic ADP literature for specific problems characterized by particular forms of the state equations and the cost (such as option pricing problems) so as to address at the same time the computation of the expected cost of the cost-to-go function and the discretization of the state space. This can be obtained by using quasi-MC uniform points not directly to sample the state and/or the random vectors spaces, but

indirectly as drivers for the simulation of system trajectories on which approximations of expected cost-to-go functions are computed through policy evaluation procedures typical of the Q-learning approach. The interested reader is referred to [27, 30] and the references cited therein among others.

Other approaches, often heuristic in nature, were proposed in the literature as alternatives to the full uniform grid. For instance, in [78] a random shift correction was added to low-discrepancy sequences aimed at minimizing dispersion; active sampling based on evolutionary methods was proposed as well. These and other sampling methods such as uniform random and low-discrepancy sets were compared on various standard test problems up to dimension 12.

An approach followed by some authors (see, for instance, [58] and the references therein) concerns the use of incremental tree structures to split the state space into different regions. Only the points corresponding to vertices of simplexes built over the partitions are used to interpolate the cost-to-go functions. Yet, even if the algorithm used to find the simplexes is efficient, the method turns out to be computationally unfeasible owing to the factorial growth of the number of simplexes in the regions. This makes this method unsuitable to high-dimensional contexts.

In [16], a method was developed that relies on a partition of the state space in fuzzy sets, each described by a membership function. Convergence and consistency analyses were provided. The admissible control set was discretized in a finite number of elements.

In [2], an approach was proposed that is based on adaptive random sampling of states, local models, and local trajectory optimization making use of different criteria. The technique proved to be successful for the control of a robotic system characterized by an eight-dimensional state.

7.4.2 Approximation Issues

As we have seen in Sect. 7.3.1, the discretization of the current and the next state vectors in ADP makes the knowledge of the approximate optimal cost-to-go functions $\bar{J}_t^\circ(\mathbf{x}_t)$ necessary not only in the sample points $\mathbf{x}_t^{(l)} \in X'_t$ but also in the whole set X_t . If the replacement of the cost $\bar{J}_t^\circ(\mathbf{x}_t)$, $\mathbf{x}_t \notin X'_t$, with the cost of the nearest neighbor $\mathbf{x}_t^{(l)} \in X'_t$ is considered too coarse, then $\bar{J}_t^\circ(\mathbf{x}_t)$ has to be approximated by a suitable FSP function. This can be obtained by solving Problem 7.1, based on the minimization of the quadratic error, or by solving a similar problem based on another criterion such as the minimax. Here, we complement all these considerations with a very short overview of some literature on this topic.

Effective approximation approaches require understanding of the structure of the problem at hand. In the case of continuous-variable problems, they typically share the feature of combining DP with tools from the theory of approximation so as to replace the cost-to-go functions with approximating FSP functions (e.g., orthogonal polynomials, splines, neural networks, etc. [45]) containing some parameters to be

optimized like the coefficients of orthogonal polynomials, the weights in the computational units of neural networks, and so on. The knowledge of smoothness properties of the approximate optimal cost-to-go functions $\bar{J}_t^o(\mathbf{x}_t)$ is useful to choose suitable FSP functions. We addressed the approximation issue in Sects. 2.6, 2.7, and Chap. 3. See, also, the discussion in [63, Chap. 11].

Combining DP with approximations of the cost-to-go functions is an approach dating back to 1959, at the very beginning of DP itself [5]. After the seminal contributions [4, 5], it is possible to trace an evolution of approximators aimed at representing the cost-to-go functions on the basis of the values computed at the nodes of regular grids. This evolution starts from polynomial approximation (see, for instance, [7]) and goes on with cubic Hermite polynomials [34, 49] and spline interpolation [46]. In [46], the use of cubic splines instead of tensor-product linear interpolants allowed the solution of a water supply problem up to dimension five. A development in this direction is [64], where the abovementioned approaches were extended to include information on some second derivatives of the cost-to-go functions. In [64], such improvements allowed as many as seven state variables, non-convex cost-to-go functions, numerous stages, nonlinear dynamics, and constraints, to be dealt with. In [28], spline interpolants were used for the solution of inventory-forecasting problems.

In [69], various approximation techniques based on linear combinations of fixed-basis functions were considered. In [81], linear approximators were exploited for the solution of certain sequential optimization problems via DP. A nice survey of approximation methods for the continuous-state case can be found in [47].

Approximation via neural networks and local approximators like kernel functions were exploited in [15, 20, 35–37, 40] and, for a class of discounted infinite-horizon stochastic optimal control problems, in [43]. In [35–37], upper bounds on the approximation error were derived in terms of the number of neural units. Among works on approximate dynamic programming, we cite also [44, 63, 71].

For discounted-reward Markov decision processes over an infinite horizon, a continuous state space, and a finite action space, [32, 59] investigate the error in the approximation of the optimal cost-to-go function owing to the replacement of the expectation operator in the (stationary) Bellman's equation with its empirical approximation. In [59], this is called *sampling-based fitted-value iteration*. For instance, [59, Sect. 5] provides probabilistic upper bounds on such an error, which depend on the number of samples. Other bounds were obtained in [32] (see, for instance, [32, Corollary 3]), where the effect of a regularization term in the optimization of the approximator was also investigated.

7.4.3 Some Structural Properties of the Optimal Cost-to-Go and Control Functions

Structural properties of the optimal cost-to-go and control functions are generally very useful in order to find a suitable family of approximating functions sharing the same properties. These properties can often be proved by showing that they are

preserved through one backward iteration of the DP procedure. Then, they can be extended to a finite and sometimes countable number of iterations [72]. Indeed, some of the following results refer to infinite-horizon optimal control problems.

- For certain families of optimal control problems, explicit solutions are available. The typical examples are the LQ problems. In this case, it is well known that the optimal cost-to-go functions are quadratic and the optimal control functions are linear in the state vector. This is proved by showing that the quadratic structure of the optimal costs-to-go is preserved through each iteration of the DP procedure. Under some assumptions, the result also holds in the limit case of an infinite horizon. Another case for which explicit solutions can be obtained by preserving the structural properties of optimal costs-to-go is Gale's cake-eating problem [39], frequently considered in macroeconomic theory.
- If the transition cost function h_t satisfies a condition of strict convexity and if at least one so-called potentially optimal path exists for each initial state, then the optimal costs-to-go are strictly convex.
- Under some conditions, the optimal cost-to-go functions are monotone.
- For a class of infinite-horizon optimal decision problems, some conditions guarantee monotonicity of the optimal stationary control function.
- For optimal decision problems with convex transition costs, it is possible to prove differentiability of the optimal costs-to-go under certain interiority assumptions by Benveniste–Scheinkman's envelope theorem (see [8]; see also [74, Chap. 4]).
- For a class of infinite-horizon optimal decision problems, differentiability of the optimal control function was proven in [56] under strict convexity of the transition cost. It can also be shown that twice local differentiability of the optimal stationary cost-to-go function is related to local differentiability of the optimal stationary control function. More precisely, under the hypotheses of Benveniste–Scheinkman's envelope theorem, if the optimal stationary control function is locally differentiable, then twice local differentiability of the optimal stationary cost-to-go function readily follows from the chain rule. Higher order differentiability of the optimal control functions was obtained locally by the analysis of the stability of the dynamic system associated with the discrete-time Euler–Lagrange equation, which provides a necessary condition for optimality. Finally, in [55], the Lipschitz continuity of the optimal stationary control function was proved under strong convexity hypotheses and an upper bound on the Lipschitz constant was given, too.
- For every admissible control function belonging to a certain family of functions, there exists an infinite-horizon optimal decision problem with convex transition costs for which such a function is the optimal control function [57].
- For finite-horizon stochastic optimal decision problems, smoothness properties of the optimal cost-to-go and control functions were studied in [36], and exploited for their approximation by FSP functions of OHL type.

7.4.4 Reinforcement Learning

As we pointed out in Sects. 7.4.1 and 7.4.2, the discretization of the state sets \tilde{X}_t to mitigate the danger of incurring the curse of dimensionality and the resulting need of computing the FSP optimal cost-to-go functions \tilde{J}_t° are two serious drawbacks. The *neuro-dynamic programming* (NDP) methodology was also proposed [15] to cope with these two inconveniences. Actually, NDP has a wider meaning and often aims at providing a theoretical basis for rather heuristic methods that solve highly complex optimization problems with surprising effectiveness. Some examples are given in [15, Chap. 8]. This is clearly expressed in the Preface of [15]: “A few years ago our curiosity was aroused by reports on new methods in reinforcement learning, a field that was developed primarily within the artificial intelligence community, starting a few decades ago. [...] Our first impression was that the new methods were ambitious, overly optimistic, and lacked firm foundation. Yet, there were claims of impressive successes and indications of a solid core to the modern developments in reinforcement learning, suggesting that the correct approach to their understanding was through dynamic programming.”

In the Preface of [15], Bellman’s curse of dimensionality is made worse by another curse. The latter is called the *curse of modeling* and arises whenever there is a lack of knowledge of the system’s dynamics. As regards the points of contact between our treatment of DP and NDP, we consider it important to point out the following two arguments:

1. In abandoning full regular grids to sample the state sets (as well as the sets of the random and sometimes of the control variables) differently in order to mitigate the danger of the curse of dimensionality, we prefer to use the quasi-MC techniques instead of the MC ones. This has been justified in Chap. 6 and in this chapter by the possibility of getting lower sample cardinalities. Such a choice does not emerge from [15] and even from other subsequent books such as [45, 63], etc. However, it must be said that a large part of [15] is devoted to various types of approximations of DP and to connections between DP and NDP. Moreover, attention is mainly focused on finite-state controlled Markov chains or finite Markov decision processes even if most of the results and algorithms can be applied to problems where infinite and possibly continuous state spaces are involved. We are unable to say with certainty in which work quasi-MC sampling was applied to DP for the first time.
2. The ADP techniques described in Chap. 6 and in this chapter are based on the fundamental hypothesis that the state equation is perfectly known. This is evident from Eqs. (6.31) and (7.21), where the knowledge of the functions f_t is required. Of course, the conditional probability densities $p_t(x_{t+1}|x_t, u_t)$ may be used instead of the functions f_t in applying ADP to the stochastic context; in this case, such densities must be known. Even when the knowledge of the system dynamics is not verified, the DP paradigm can still be applied and usually takes

the name of *reinforcement learning*.³ As is easily understandable, RL is strictly connected with both *adaptive control*⁴ and *optimal control* theories. However, it must be clear that this book has nothing to do with adaptive control methods even if neural networks provide powerful tools for the study of this very important control area.

Before describing RL, some information on the origins and developments of the method may be helpful. The bases of RL were set in 1959 by Samuel [68] who, independently from Bellman and Dreyfus [5], developed approximation techniques to train a computer to play checkers (see also [75]). The relationship between DP and RL was clearly put in [76], another key book which marked the entrance of computer scientists in the research on DP. The monograph [17] is devoted to RL and DP using function approximation techniques (see also [13]). In fact, even in the RL approach (as with the cost-to-go functions of DP), the need arises of approximating the so-called Q-functions (see Sect. 7.4.5) for the unknown state and control vectors to be used in the optimization phase. Many commonly employed approximating models were proposed for this purpose, from neural networks (see, for instance, [66]) to local methods based on kernels [61] and trees [31]. It has to be remarked that RL has so far been successful in solving problems of relatively small size (mostly standard test problems, such as the two-dimensional “car-on-the-hill” problem or the four-dimensional “swinging acrobot”) and characterized by a finite set of admissible controls. This is because the basic assumption of not knowing the system dynamics – combined with the fact that the Q-function has to be approximated over the space of the states *and* the space of the controls without the freedom of choosing the sampling points – increases the difficulty of the approximation task, making the dynamic optimization problem intrinsically hard. In [48], reinforcement learning methods were developed for the solution of optimal control problems in nonlinear deterministic dynamical systems. We also cite the recent surveys [50, 84].

In describing RL, it is worth saying in advance that, whereas DP determines the optimal control law and the optimal cost-to-go functions by an offline procedure that works backward, RL finds the control law and the costs by methods that impose progressively (i.e., stagewise) suboptimal control functions on the basis of the results observed along the system trajectory. As here we consider an infinite-horizon optimal control problem, Cost (7.3) is replaced by

³The terminology “reinforcement learning” originates from studies of animal learning in experimental psychology [79], in which it was observed that, when interacting with its own environment, an animal usually learns how to improve the outcomes of future choices from past experiences.

⁴Since we have assumed that the system dynamics is known, our use of the term “approximate” in the abbreviation ADP has to be interpreted essentially as “numerically approximate,” without reference to the classical situation of RL, in which a further approximation is associated with the lack of the model knowledge. In this different situation, where the adaptive control context plays a basic role, the abbreviation stands for *adaptive dynamic programming*. However, we shall not use the abbreviation ADP with this meaning.

$$J = \sum_{t=0}^{\infty} \alpha^t h(\mathbf{x}_t, \mathbf{u}_t, \xi_t), \quad (7.44)$$

where $0 < \alpha < 1$ is a discount factor. State Equation (7.1), Constraints (7.2), and the transition cost h_t are assumed to be time-independent.

We are interested in stationary control laws $\{\mathbf{u}_t = \mu(\mathbf{x}_t), t = 0, 1, \dots\}$. The corresponding cost-to-go functions are denoted by $J^\mu(\mathbf{x}_t)$. Consider the stationary optimal control law that minimizes the expected value of Cost (7.44). The optimal control functions and the related costs are denoted by $\mathbf{u}_t^\circ = \mu^\circ(\mathbf{x}_t)$ and $\hat{J}^\circ(\mathbf{x}_t)$, respectively. The considerations that follow are very qualitative. We briefly draw some concepts from [54], where techniques for adaptive online control laws through the RL paradigm are discussed (see also the references cited therein). Under suitable assumptions, the solution to the infinite-horizon optimal control problem is given by the following stationary version of DP Eq. (7.13b):

$$\hat{J}^\circ(\mathbf{x}_t) = \min_{\mathbf{u}_t \in U(\mathbf{x}_t)} \mathbb{E}_{\xi_t} \left[h(\mathbf{x}_t, \mathbf{u}_t, \xi_t) + \alpha \hat{J}^\circ(\mathbf{x}_{t+1}) \right]. \quad (7.45)$$

Note that the same optimal cost-to-go function $\hat{J}^\circ(\cdot)$ appears on both sides of (7.45). This function does not depend on the stage t .

Quite a natural technique for solving (7.45) is the use of the backward recursive procedure given by ADP Eq. (7.21) for $T \rightarrow \infty$, that is, in practice, for large values of T (put $h_T(\mathbf{x}_T) = 0$ in (7.21a)). In general, there are several methods to solve (7.45). We refer the reader to [10]. Under LQ assumptions and other well-known hypotheses, (7.45) can be solved via the solution of the discrete-time algebraic Riccati equation. However, in RL one is mainly interested in designing control functions that solve optimal control problems without knowing or only partially knowing the system dynamics.

For brevity, among the various techniques described in [54], we focus on Section “Optimal Adaptive Control Using a Policy Iteration Algorithm”. The algorithm described therein is based on the concept of *temporal difference learning*. Using a very qualitative description, we begin by replacing Eq. (7.45) with the following one:

$$\hat{J}^\mu(\mathbf{x}_t) = \mathbb{E}_{\xi_t} \left\{ h[\mathbf{x}_t, \mu(\mathbf{x}_t), \xi_t] + \alpha \hat{J}^\mu(\mathbf{x}_{t+1}) \right\}. \quad (7.46)$$

Comparing (7.46) with (7.45), we note that in (7.46) the “min” operator is not present. This is due because neither \hat{J}^μ nor μ are optimal. Now, the dynamic system trajectory is a sequence of random segments or sample paths, in each of which the state is driven from \mathbf{x}_t to \mathbf{x}_{t+1} by \mathbf{u}_t and ξ_t . Then, we can resort to the concept of *stochastic approximation* (SA) described in Sect. 5.3.3. This means that the average in (7.46) is replaced by sample paths, which follow one after the other. So, (7.46) can be replaced by

$$J^\mu(\mathbf{x}_t) \simeq h[\mathbf{x}_t, \mu(\mathbf{x}_t), \xi_t] + \alpha J^\mu(\mathbf{x}_{t+1}), \quad (7.47)$$

which in [54] is called (with the equality) the *deterministic Bellman equation*. In (7.47), $J^\mu(\mathbf{x}_t)$ may be considered as an estimate of the cost-to-go at the state \mathbf{x}_t and $\alpha J^\mu(\mathbf{x}_{t+1})$ may be regarded as a prediction of it at the next state \mathbf{x}_{t+1} (moving from \mathbf{x}_t to \mathbf{x}_{t+1} , the acquired data are given by \mathbf{x}_t , \mathbf{x}_{t+1} , and the transition cost $h(\mathbf{x}_t, \mathbf{u}_t, \xi_t)$). Therefore, the following *temporal difference error* is defined:

$$e_t = h(\mathbf{x}_t, \mathbf{u}_t, \xi_t) + \alpha J^\mu(\mathbf{x}_{t+1}) - J^\mu(\mathbf{x}_t). \quad (7.48)$$

The methods of *temporal difference learning* aim at making the average of the error (7.48) smaller and smaller. As we said before, this should be achieved by SA that uses the deterministic Bellman equation (7.47) stage after stage until, on the average, the solution of the Bellman equation (7.46) is reached.

In order to obtain the optimal control function $\mathbf{u}_t^* = \mu^*(\mathbf{x}_t)$ in an analytical form, let us make the following simplifying assumptions:

(i) State Equation (7.1) takes the form

$$\mathbf{x}_{t+1} = \tilde{\mathbf{f}}(\mathbf{x}_t) + F(\mathbf{x}_t)\mathbf{u}_t, \quad t = 0, 1, \dots, \quad (7.49)$$

(ii) the transition cost in (7.44) is given by

$$h(\mathbf{x}_t, \mathbf{u}_t, \xi_t) = Q(\mathbf{x}_t) + \mathbf{u}_t^\top R \mathbf{u}_t, \quad (7.50)$$

where $Q(\mathbf{x}_t) > 0$ and $R = R^\top > 0$. Note that in (7.49) and (7.50) the random vector ξ_t is not present. Its presence is not essential because we are considering the deterministic Bellman equation (7.47).

We now introduce an important approximation, which should be familiar after what we stated in Chap. 2 and, in particular, in Sects. 2.6 and 2.7. We assume that the cost \hat{J}^μ can be approximated arbitrarily well by some type of FSP functions (see Definitions 2.5, 2.6, and Remark 2.4), possibly by ε -pc approximating FSP functions (see Definition 2.8). For simplicity (see (64) of [54]), let each FSP function be a linear combination of fixed-basis functions. Then, it takes the form (3.1) with $m = 1$, i.e.,

$$\tilde{J}^\mu(\mathbf{x}) \triangleq \sum_{i=1}^n c_i \varphi_i(\mathbf{x}) = \mathbf{w}^\top \varphi(\mathbf{x}), \quad (7.51)$$

where $\mathbf{w} = \text{col}(c_1, \dots, c_n)$ and $\varphi(\mathbf{x}) = \text{col}(\varphi_1(\mathbf{x}), \dots, \varphi_n(\mathbf{x}))$.

In the *policy iteration algorithm*, described in [54, p. 91], we have the following steps.

1. Initialization

Set $j = 0$ and iterate on j until convergence. Choose an admissible⁵ control function $\mu_0(\mathbf{x}_t)$.

⁵We assume that the system (7.49) is “stabilizable” on some set $\Omega \subset \mathbb{R}^d$. In qualitative terms, this means that there exists a control function $\mu(\mathbf{x}_t)$ such that the closed-loop system $\mathbf{x}_{t+1} = \tilde{\mathbf{f}}(\mathbf{x}_t) +$

2. Policy Evaluation Step

Let us impose the temporal difference error e_t to be zero. This means that the weight vector \mathbf{w}_{j+1} verifies the following equation (see (7.48), use (7.50) and (7.51), and remove ξ_t in the transition cost):

$$h[\mathbf{x}_t, \mu_j(\mathbf{x}_t)] + \alpha \mathbf{w}_{j+1}^\top \varphi(\mathbf{x}_{t+1}) - \mathbf{w}_{j+1}^\top \varphi(\mathbf{x}_t) = 0$$

and then

$$\mathbf{w}_{j+1}^\top [\varphi(\mathbf{x}_t) - \alpha \varphi(\mathbf{x}_{t+1})] = h[\mathbf{x}_t, \mu_j(\mathbf{x}_t)]. \quad (7.52)$$

Note that every coefficient in (7.52) is known, as we can measure \mathbf{x}_t , \mathbf{x}_{t+1} , and $h[\mathbf{x}_t, \mu_j(\mathbf{x}_t)]$. However, (7.52) is a scalar equation whereas $\dim(\mathbf{w}_{j+1}) = n$. So, we need at least n sample paths to obtain a linear system of n equations and n unknowns. Clearly, it is better to measure the data from more than n subsequent sample paths and to determine \mathbf{w}_{j+1} by a batch least-squares procedure. Alternatively, a recursive least-squares technique or a gradient descent tuning algorithm can be used. See [54, p. 91] for details.

3. Policy Improvement Step

Once the vector \mathbf{w}_{j+1} has been computed, the “improved” control function μ_{j+1} is derived by minimizing the right-hand side of (7.47) with respect to \mathbf{u}_t for any admissible \mathbf{x}_t , with J^μ replaced by its approximation \tilde{J}^μ from (7.51), that is,

$$\begin{aligned} \mu_{j+1}(\mathbf{x}_t) &= \min_{\mathbf{u}_t} [h(\mathbf{x}_t, \mathbf{u}_t) + \alpha \tilde{J}^\mu(\mathbf{x}_{t+1})] \\ &= \min_{\mathbf{u}_t} \left\{ Q(\mathbf{x}_t) + \mathbf{u}_t^\top R \mathbf{u}_t + \alpha \mathbf{w}_{j+1}^\top \varphi[\tilde{f}(\mathbf{x}_t) + F(\mathbf{x}_t) \mathbf{u}_t] \right\}. \end{aligned}$$

Setting to zero the gradient with respect to \mathbf{u}_t of the expression inside the curly brackets, we obtain

$$\begin{aligned} 2R\mathbf{u}_t + \alpha \sum_{i=1}^n c_i \left[\frac{\partial \varphi_i(\mathbf{x}_{t+1})}{\partial \mathbf{u}_t} \right]^\top \\ = 2R\mathbf{u}_t + \alpha \sum_{i=1}^n c_i \left\{ \left[\frac{\partial \varphi_i(\mathbf{x}_{t+1})}{\partial \mathbf{x}_{t+1}} \frac{\partial (\tilde{f}(\mathbf{x}_t) + F(\mathbf{x}_t) \mathbf{u}_t)}{\partial \mathbf{u}_t} \right]^\top \right\} = \mathbf{0}. \end{aligned}$$

Then,

$$\mathbf{u}_t = \mu_{j+1}(\mathbf{x}_t) = -\frac{\alpha}{2} R^{-1} F(\mathbf{x}_t)^\top \left(\frac{\partial \varphi(\mathbf{x}_{t+1})}{\partial \mathbf{x}_{t+1}} \right)^\top \mathbf{w}_{j+1}. \quad (7.53)$$

The entire procedure is repeated until convergence to the optimal control function.

$F(\mathbf{x}_t)\mu(\mathbf{x}_t)$ is asymptotically stable on Ω . A control function $\mu(\mathbf{x}_t)$ is said to be “admissible” if it is stabilizing and yields a finite cost-to-go $J^\mu(\mathbf{x}_t)$.

A method similar to that described above, based on policy iteration, is given by the *value iteration algorithm*. See for details [54, p. 91]. It is worth noting that the optimal control function takes the form (7.53), where the knowledge of the function \tilde{f} is not required but the matrix $F(\mathbf{x}_t)$ must be known.

7.4.5 Q-Learning

Among several RL techniques, we illustrate some concepts of *Q-learning* (without any purpose of generality and completeness). To do this, it may be useful to remain in the same context as in the previous section and to continue considering an infinite-horizon optimal control problem with State Equation (7.49) and Transition Cost (7.50). We have seen that the knowledge of the matrix $F(\mathbf{x}_t)$ is necessary to determine the optimal control function (7.53). Then, it is interesting to understand what can be done to deal with cases in which $F(\mathbf{x}_t)$ is not known (besides $\tilde{f}(\mathbf{x}_t)$) and, in general, if there is no knowledge on the system dynamics. To this end, define the Q-function [76, 77, 85, 86]

$$\hat{Q}^\mu = \underset{\xi_t}{\mathbb{E}} \left[h(\mathbf{x}_t, \mathbf{u}_t, \xi_t) + \alpha \hat{J}_{t+1}^\mu(\mathbf{x}_{t+1}) \right], \quad (7.54)$$

where μ is a given control function (the letter Q derives from “quality function”). The substantial difference between the Q-function \hat{Q}^μ and the cost-to-go function \hat{J}^μ is the dependence of the former function on both the state and the control vectors (there are similarities between $\tilde{J}_t(\mathbf{x}_t^{(l)}, \mathbf{u}_t)$ (see (7.23)) and the Q-function). The Q-function is at the basis of Q-learning. This method was developed by Watkins [85, 86] and Werbos (see the edited volumes [13, 88] and also [87, 89]). Werbos called it “action-dependent heuristic dynamic programming” (ADHDP) because of its dependence on the control vector \mathbf{u}_t .

For a time-invariant control function $\mathbf{u}_t = \mu(\mathbf{x}_t)$, we have

$$\hat{J}^\mu(\mathbf{x}_t) = \hat{Q}^\mu[\mathbf{x}_t, \mu(\mathbf{x}_t)].$$

Then, the Q-function can be expressed as in Eq. (7.46), i.e.,

$$\hat{Q}^\mu[\mathbf{x}_t, \mu(\mathbf{x}_t)] = \underset{\xi_t}{\mathbb{E}} \left\{ h[\mathbf{x}_t, \mu(\mathbf{x}_t), \xi_t] + \alpha \hat{Q}^\mu[\mathbf{x}_{t+1}, \mu(\mathbf{x}_{t+1})] \right\}.$$

By resorting to SA as in the previous section, we again consider a sequence of sample paths along the state trajectory. Each of them is given by

$$Q^\mu(\mathbf{x}_t, \mathbf{u}_t) \simeq h(\mathbf{x}_t, \mathbf{u}_t) + \alpha Q^\mu[\mathbf{x}_{t+1}, \mu(\mathbf{x}_{t+1})].$$

Then, as in (7.48), we define the temporal difference error

$$e_t \triangleq h(\mathbf{x}_t, \mathbf{u}_t) + \alpha Q^\mu[\mathbf{x}_{t+1}, \boldsymbol{\mu}(\mathbf{x}_{t+1})] - Q^\mu(\mathbf{x}_t, \mathbf{u}_t). \quad (7.55)$$

Now, we assume that $\hat{Q}^\mu(\mathbf{x}, \mathbf{u})$ can be approximated arbitrarily well by an LCFBF (3.1) with $m = 1$, i.e., like $\tilde{J}^\mu(\mathbf{x})$ in (7.51). Of course, the argument is given by $\mathbf{z} \triangleq \text{col}(\mathbf{x}, \mathbf{u})$, that is,

$$\tilde{Q}^\mu(\mathbf{x}, \mathbf{u}) \triangleq \mathbf{w}^\top \boldsymbol{\varphi}(\mathbf{z}). \quad (7.56)$$

In order to determine the unknown parameter vector \mathbf{w} and the optimal policy $\boldsymbol{\mu}^\circ$, we may use the policy iteration or of the value iteration approach. As in the previous section, we choose the former and we present the related iterative procedure (we continue to refer to [54]).

1. Initialization Step

Choose a control function $\boldsymbol{\mu}_0$. Then, starting from $j = 0$, iterate on j until convergence.

2. Q -function Evaluation Step

Substituting (7.56) into (7.55), we have

$$e_t = h(\mathbf{x}_t, \mathbf{u}_t) + \alpha \mathbf{w}_{j+1}^\top \boldsymbol{\varphi}(\mathbf{z}_{t+1}) - \mathbf{w}_{j+1}^\top \boldsymbol{\varphi}(\mathbf{z}_t).$$

Let us impose

$$\mathbf{w}_{j+1}^\top [\boldsymbol{\varphi}(\mathbf{z}_t) - \alpha \boldsymbol{\varphi}(\mathbf{z}_{t+1})] = h(\mathbf{x}_t, \mathbf{u}_t). \quad (7.57)$$

To determine \mathbf{w}_{j+1} , a number of scalar equations (7.57) is needed at least equal to $d + m$. The online computation of \mathbf{w}_{j+1} can be implemented by batch or recursive least-squares techniques. Note that all the coefficients in (7.57) are known. Indeed, \mathbf{x}_t , \mathbf{x}_{t+1} , and $h(\mathbf{x}_t, \mathbf{u}_t)$ can be observed. So, $\mathbf{z}_t = \text{col}(\mathbf{x}_t, \mathbf{u}_t)$ is known. $\mathbf{z}_{t+1} = \text{col}(\mathbf{x}_{t+1}, \mathbf{u}_{t+1})$ can also be determined by letting $\mathbf{u}_{t+1} = \boldsymbol{\mu}_j(\mathbf{x}_{t+1})$, where $\boldsymbol{\mu}_j$ is the control function at the iteration j .

3. Policy Improvement Step

The control function $\boldsymbol{\mu}_{j+1}$ is computed by minimizing (7.56) with respect to \mathbf{u}_t for any admissible \mathbf{x}_t , i.e.,

$$\boldsymbol{\mu}_{j+1}(\mathbf{x}_t) = \arg \min_{\mathbf{u}_t} [\mathbf{w}_{j+1}^\top \boldsymbol{\varphi}(\mathbf{x}_t, \mathbf{u}_t)].$$

The minimization is performed by computing explicitly $\partial[\mathbf{w}_{j+1}^\top \boldsymbol{\varphi}(\mathbf{x}_t, \mathbf{u}_t)]/\partial \mathbf{u}_t$, using some gradient-based algorithm.

The entire procedure is repeated until convergence.

The Q-learning based on the value iteration approach can be performed in a rather similar way. As we have seen, Q-learning, unlike RL, does not require the knowledge

of the matrix $F(\mathbf{x}_t)$. For this reason, it allows the online computation of the optimal control function for a completely unknown dynamic system.

Finally, we repeat that our discussion on RL and Q-learning has been quite qualitative. For example, issues on the persistence of excitation should be discussed. Another fundamental problem is given by the stability properties of the control system designed via the RL and particularly the Q-learning (as we already said, the right place to discuss the stability should be Chap. 10, where infinite-horizon optimal control problems are addressed). For recent results on stability, see, for instance, [73] and the references cited therein, where the analysis is focused on the context of ADHDP. Such results show better control performance as compared with the ones previously appeared in the literature. For example, nice numerical simulations are reported where the very popular benchmark of the cart-pole balancing problem is considered: the proposed controller is able to stabilize the system even though the initial deviation of the pole angle is quite large.

7.5 Approximate Solution of Problems C2 and C2' by the ERIM

We have the possibility of applying the ERIM extensively for the first time in the book. This is also made stimulating by the fact that we shall have the opportunity of comparing a classic methodology (i.e., the ADP approach examined in the previous sections) with a new one. It will be possible to make the comparison while giving special attention to the basic computational aspects related to the various types of the curse of dimensionality.

Indeed, Problems C2 and C2' offer an appropriate framework in which ADP and the ERIM can be compared. This is due to the fact that \mathbf{x}_t is perfectly measurable. In the next chapter, we shall address a much more difficult stochastic optimal control problem for which the assumption of the perfect measurability of the state is no longer verified. This makes ADP difficult or even impossible to apply. Instead, the ERIM can still be used provided that some suitable approximations are accepted. Similar situations will occur in the remainder of the book for IDO problems like team optimal control problems. In all these cases, ADP ceases to be an applicable tool, whereas the ERIM does not lead to too great difficulties.

7.5.1 Approximation of the IDO Problem C2 by the Nonlinear Programming Problems C_{2n}

As we previously explained (see Sect. 2.2.2 and the chain of FSP functions in Fig. 7.2 of Sect. 7.6.2), the ERIM consists in constraining the closed-loop control functions (7.4) to take on the structure of FSP functions, that is,

$$\mathbf{u}_t = \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t}), \quad t = 1, \dots, T-1. \quad (7.58)$$

In (7.58), we point out that both the parameter vector \mathbf{w}_{tn_t} and its dimension, which is related to the model complexity n_t , may vary over time. The dependence of n_t on t is due to the fact that the regularity properties of the function $\mu_t^\circ(\mathbf{x}_t)$ to be approximated may be time-varying; hence, the model complexity may be time-varying, too. We shall go on assuming that the FSP functions will remain of the same type, stage after stage. We also recall that, in Problem C2, no FSP function is needed at the stage $t = 0$, since the initial state is the fixed vector $\mathbf{x}_0 = \hat{\mathbf{x}}$. Then, we just have to optimize the vector \mathbf{u}_0 .

Let us now replace the control functions $\mu_1(\mathbf{x}_1), \dots, \mu_{T-1}(\mathbf{x}_{T-1})$ with the FSP functions (7.58) in the cost functional (7.5). Similarly, let us replace the control vectors $\mathbf{u}_1, \dots, \mathbf{u}_{T-1}$ with the functions (7.58) in the state equation (7.1). Then, let us eliminate the state vectors $\mathbf{x}_1, \dots, \mathbf{x}_T$ in Cost (7.5) by repeatedly using the state equation. After these operations, Cost (7.5) is no longer a functional. Indeed, it does not depend any more on the control functions μ_1, \dots, μ_{T-1} but it turns out to be a function of the variables that escaped from the elimination. We can write the cost in the form

$$\underset{\xi_0, \xi_1, \dots, \xi_{T-1}}{\mathbb{E}} J(\mathbf{u}_0, \mathbf{w}_{1n_1}, \dots, \mathbf{w}_{T-1,n_{T-1}}, \mathbf{x}_0, \xi_0, \xi_1, \dots, \xi_{T-1}) \quad (7.59)$$

or, in a more compact form,

$$\hat{J}(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0) \triangleq \underset{\xi}{\mathbb{E}} J(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi), \quad (7.60)$$

where

$$\xi \triangleq \text{col}(\xi_0, \xi_1, \dots, \xi_{T-1}) \quad (7.61)$$

and

$$\mathbf{w}_n \triangleq \text{col}(\mathbf{w}_{1n_1}, \dots, \mathbf{w}_{T-1,n_{T-1}}). \quad (7.62)$$

(For notational simplicity, we use the same symbol J in (7.3), (7.59), and (7.60).) The subscript n in \mathbf{w}_n denotes the *global model complexity* of the chain of FSP functions as described in Sect. 2.2.2. As in Sect. 2.2.2, we reduced the functional optimization or IDO Problem C2 to an NLP Problem P_n (see (2.28)).

Now, to emphasize the derivation of Problem P_n from Problem C2, let us replace the term P_n with $C2_n$ and, for a given n , state the following problem.

Problem C2_n. Find the vectors $\mathbf{u}_0^\circ, \mathbf{w}_n^\circ$ that minimize the expected cost

$$\hat{J}(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0) \quad (7.63)$$

under the constraints $\mathbf{u}_0 \in U_0(\mathbf{x}_0)$ and $\mathbf{w}_n \in W_n(\mathbf{u}_0)$. \triangleleft

Note that the set $W_n(\mathbf{u}_0)$ is parametrized by \mathbf{u}_0 , consistently with the intuition and as will be clarified in Remark 7.4, where its construction will be provided.

For every n , the solution of Problems $C2_n$ yields the FSP optimal control law

$$\mathbf{u}_0^\circ, \mathbf{u}_t^\circ = \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t}^\circ), \quad t = 1, \dots, T - 1. \quad (7.64)$$

Of course, each Problem $C2_n$ should be considered within the framework of an infinite sequence of NLP problems that approximate Problem C2 better and better with increasing values of n . In practice, one implements a trial-and-error procedure. This means that one chooses a “reasonable” model complexity n ; then, roughly speaking, one increases or decreases n until values of n are found for which the variations of the minimum expected cost $\hat{J}(\mathbf{u}_0^\circ, \mathbf{w}_n^\circ, \mathbf{x}_0)$ and the elements of the optimal control law (7.64) are led below given thresholds. Then, the minimum of such values of the expected cost is taken.

Remark 7.4 Constraints (7.2) and constraints on the state vector may be present. As we said in Sect. 7.1, it is reasonable to take the latter into consideration if the random vectors ξ_t take their values on bounded domains. If such constraints are not present, then Problem $C2_n$ is an unconstrained NLP problem for which several powerful solution algorithms exist (see Sect. 5.3). However, if those constraints are present, Problem $C2_n$ may become a very difficult NLP problem. As we said in Remark 5.2, this happens if one has to transform the constraints of Problem C2 in the constraints set $W_n(\mathbf{u}_0)$ of Problem $C2_n$. Such a transformation is similar to the one by which we derived Cost (7.63). To get more insight, suppose that there are constraints expressed by the inequalities⁶

$$\mathbf{g}_t(\mathbf{u}_t) \geq \mathbf{0}, \quad t = 1, \dots, T - 1. \quad (7.65)$$

Note that Inequalities (7.65) only constrain the control vectors but that the state \mathbf{x}_t appears when one replaces \mathbf{u}_t in (7.65) with the FSP functions (7.58), i.e.,

$$\mathbf{g}_t[\tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t})] \geq \mathbf{0}, \quad t = 1, \dots, T - 1. \quad (7.66)$$

Constraints of the type (1.15), which depend explicitly on the state vector \mathbf{x}_t , can be considered in the same way as we shall do in the following paragraphs. In order to make the constraint set W_n explicit, one has to implement a procedure similar to the one that led to Cost (7.63). This means that in Inequalities (7.66) the state vector is eliminated by repeatedly applying the state equation, thus obtaining, in case of random disturbances with domains independent from the state and control vectors,

$$\begin{aligned} \tilde{\mathbf{g}}_t(\mathbf{u}_0, \mathbf{w}_{1n_1}, \dots, \mathbf{w}_{tn_t}, \mathbf{x}_0, \xi_0, \xi_1, \dots, \xi_{t-1}) &\geq \mathbf{0}, \quad t = 1, \dots, T - 1, \\ \xi_0 \in \Xi_0, \xi_1 \in \Xi_1, \dots, \xi_{t-1} \in \Xi_{t-1}. \end{aligned} \quad (7.67)$$

⁶See Constraints (1.15) for the case of continuous-time optimal control problems: the equalities $\mathbf{g}_t(\mathbf{u}_t) = \mathbf{0}$ can be converted into the inequalities $\mathbf{g}_t(\mathbf{u}_t) \geq \mathbf{0}$ and $-\mathbf{g}_t(\mathbf{u}_t) \geq \mathbf{0}$.

W_n is then defined as the set of all vectors \mathbf{w}_n that verify Inequalities (7.67) for *any* vector $\xi \in \Xi_0 \times \dots \times \Xi_{T-1}$. Note that W_n depends on $\mathbf{u}_0 \in U_0(\mathbf{x}_0)$ and \mathbf{x}_0 (which is fixed). This dependence can be highlighted by writing $W_n(\mathbf{u}_0)$, as we did in the statement of Problem C2_n. What was anticipated in Remark 5.2 follows, i.e., W_n has a formal but not practical meaning. Indeed, the functions \tilde{g}_t are too complex to allow the computation of W_n . Note that, instead, the functions g_t may have a simple form. For example, they may be linear. The functions \tilde{g}_t , on the contrary, are composite functions containing FSP functions which are generally complicated. Note also that, in general, it is not allowed to interpret Inequalities (7.67) as parametrized by random variables. Sometimes, in stochastic contexts, constraints may be interpreted in a probabilistic form. For example, from [70, p. 87] we report “... imposing constraints on probabilistic events is particularly appropriate whenever high uncertainty is involved and reliability is a central issue”. In other cases, the constraints can be satisfied on average or with high probability or by replacing the random variables with their expected values (“certainty equivalence”). In fact, Constraints (7.65) are deterministic and cannot be violated while the randomness comes from the stochastic state equations.

To sum up, the difficulties in determining the set W_n will lead us to solve the constrained optimal control problems encountered in the examples of the book by resorting to suitable penalty functions. This will allow us to solve the related unconstrained NLP problems by the algorithms described in Sect. 7.6. Finally, it is worth noting that quite common inequalities take on the form $u_{t,\min}^{(j)} \leq u_t^{(j)} \leq u_{t,\max}^{(j)}$, where $u_t^{(j)}$ is the j th component of the vector \mathbf{u}_t . Such inequalities are satisfied if one chooses FSP feedforward neural control functions with the output layer containing the activation functions $(u_{t,\max}^{(j)} - u_{t,\min}^{(j)}) \tanh z_j$ (see Example 7.4 in this chapter). \triangleleft

7.5.2 Approximation of Problems C2' by the Nonlinear Programming Problems C2'_n

In the previous section, we saw that the ERIM enables one to approximate Problem C2 by a sequence of NLP Problems C2_n. Let us now do the same in order to approximate Problem C2' by a suitable sequence of NLP problems that we define as Problems C2'_n (for simplicity, suppose that n has been derived as we said in the previous section after the statement of Problem C2_n).

We recall that, in Problem C2', the DM does not know the value of \mathbf{x}_0 before the decision process starts. It only knows that \mathbf{x}_0 belongs to a given region $X_0 \subseteq \mathbb{R}^d$. As soon as the process starts, the DM can perfectly observe \mathbf{x}_0 . In the ideal case, at time $t = 0$, the DM should have at its disposal the optimal control function

$$\mathbf{u}_0^\circ = \mu_0^\circ(\mathbf{x}_0), \quad \mathbf{x}_0 \in X_0, \quad (7.68)$$

i.e., the initial function of the optimal control law that solves Problem C2' via the “exact” DP Eq. (7.13).

In practice, we recall that, in applying ADP, Function (7.68) can only be determined (in an approximate way) pointwise for every discretized vector $\mathbf{x}_0^{(l)}$ ($l = 1, \dots, L_0$) of the set X'_0 . Then, by solving Problem 7.2 for $t = 0$, one obtains the FSP optimal approximation of (7.68), that is (see (7.36)),

$$\mathbf{u}_0^\circ = \tilde{\mu}(\mathbf{x}_0, \mathbf{w}_{0n_0\Sigma_0}^{\mu\circ}), \quad \mathbf{x}_0 \in X_0. \quad (7.69)$$

Function (7.69) can be computed offline, thus avoiding the online reoptimization procedure at $t = 0$ specified by (7.33).

In the ERIM’s framework, a procedure must be implemented similar to that described in the previous section. This means that a control function $\mathbf{u}_0 = \mu_0(\mathbf{x}_0)$ (see (7.4)) has to be introduced and optimized according to the statement of Problem C2'. Then, we constrain it to take on the structure of an FSP function, i.e.,

$$\mathbf{u}_0 = \tilde{\mu}(\mathbf{x}_0, \mathbf{w}_{0n_0}), \quad \mathbf{x}_0 \in X_0, \quad (7.70)$$

so we must optimize an additional FSP function for $t = 0$ that replaces the vector \mathbf{u}_0 in Problems C2_n (see (7.63)). It follows that we have to optimize the sequence of FSP functions

$$\mathbf{u}_t = \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t}), \quad t = 0, 1, \dots, T - 1. \quad (7.71)$$

By using State Equation (7.1) and FSP Control Functions (7.71) (as we did in Sect. 7.5.1 to derive Costs (7.59) and (7.60)), in the cost (7.3) the state and control vectors can be eliminated. The expected value becomes

$$\hat{J}(\mathbf{w}_n, \mathbf{x}_0) = \underset{\xi}{\mathbb{E}} J(\mathbf{w}_n, \mathbf{x}_0, \xi), \quad \mathbf{x}_0 \in X_0, \quad (7.72)$$

where

$$\mathbf{w}_n \triangleq \text{col}(\mathbf{w}_{0n_0}, \mathbf{w}_{1n_1}, \dots, \mathbf{w}_{T-1,n_{T-1}}). \quad (7.73)$$

As Cost (7.72) is parametrized by \mathbf{x}_0 , we should minimize it (with respect to \mathbf{w}_n) for each of the infinite possible initial states $\mathbf{x}_0 \in X_0$, i.e., we should solve an infinite number of Problems C2_n. Obviously, this is impossible. Therefore, we can consider the discretized set X'_0 derived from X_0 and, for each discretized state $\mathbf{x}_0^{(l)} \in X'_0$, we can solve a specific Problem C2_n. So, we determine L_0 optimal samples of the FSP optimal functions of the form (7.70), that is,

$$\mathbf{u}_0^{(l)\circ} = \tilde{\mu}(\mathbf{x}_0, \mathbf{w}_{0n_0}^{(l)\circ}), \quad l = 1, \dots, L_0. \quad (7.74)$$

Clearly, the initial state \mathbf{x}_0 generally does not coincide with any discretized state $\mathbf{x}_0^{(l)}$. In principle, one may accept the approximation of \mathbf{x}_0 by the nearest discretized state $\mathbf{x}_0^{(l)}$ or the use of a least-squares criterion at the stage $t = 0$ as in Problems 6.5

and 7.3. However, this implies the solution of a suitably large number L_0 of NLP problems to determine the optimal control vectors $\mathbf{u}_0^{(l)\circ}$. Note, however, that such NLP problems are computationally quite cumbersome as we have to find the optimal values of the entire parameter vectors \mathbf{w}_n defined in (7.73).

Therefore, we resort to the following approach. Given that in any case we have to perform the average of the cost with respect to ξ (see (7.72)), we interpret $\mathbf{x}_0 \in X_0$ as if it would be a random vector that is unknown to the DM before the decision process starts and becomes known to it at the stage $t = 0$. Then, we also average the cost (7.72) with respect to \mathbf{x}_0 over X_0 and we define the cost

$$\hat{J}(\mathbf{w}_n) = \underset{\mathbf{x}_0, \xi}{\text{E}} J(\mathbf{w}_n, \mathbf{x}_0, \xi). \quad (7.75)$$

Since in stating Problem C2' we only required that an optimal vector \mathbf{u}_0° be associated to any initial state vector $\mathbf{x}_0 \in X_0$, there is no reason for privileging certain regions of X_0 over others. As a consequence, it is reasonable to assume that \mathbf{x}_0 is uniformly distributed on X_0 .

Note that, as in Remark 7.4, possible constraints expressed by Inequalities (7.65) (with the additional inequalities $\mathbf{g}_0(\mathbf{u}_0) \geq \mathbf{0}$) can be transformed into inequalities of the form (7.67) (with the additional inequalities of the form $\tilde{\mathbf{g}}_0(\mathbf{w}_{0n_0}, \mathbf{x}_0) \geq \mathbf{0}$). W_n is no longer parametrized by \mathbf{u}_0 . Besides ξ , \mathbf{x}_0 appears as a random variable in all the functions $\tilde{\mathbf{g}}_t$ (see (7.67)). Again, the computation of W_n is generally impossible. However, we postpone the discussion on the issue of the constraints and we continue to keep into account the set W_n . Therefore, although aware of the “theoretical” meaning of W_n , for a given value of n , we state the following constrained NLP problem.

Problem C2'_n. Find the vector \mathbf{w}_n° that minimizes the expected cost (7.75) under the constraint $\mathbf{w}_n \in W_n$. \triangleleft

The solution of Problem C2'_n gives the FSP optimal control law

$$\mathbf{u}_t^\circ = \hat{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t}^\circ), \quad t = 0, 1, \dots, T - 1. \quad (7.76)$$

Remark 7.5 The optimal control Problem C2' can be intrinsically stochastic with respect to $\mathbf{x}_0 \in X_0$. This occurs whenever the DM does not know \mathbf{x}_0 until the instant $t = 0$ is reached but is given a priori information about the value that \mathbf{x}_0 can take on X_0 . If such information can be expressed as a probability density $p_0(\mathbf{x}_0)$, then this density can replace the assumption that \mathbf{x}_0 is uniformly distributed on X_0 . As we shall see in Sect. 7.8, optimal control problems characterized by the presence of parameters or variables that only become known to the DM at the instant $t = 0$ can be easily restated in terms of Problem C2'. \triangleleft

Remark 7.6 As we already explained in Chap. 1, the ERIM aims at providing approximate solutions for IDO problems. The T -stage optimal control Problems C1 and C2 (and their variants C1' and C2') are just important examples of IDO problems that benefit from the possibility of being solved by DP, which exploits the “divide

et impera” concept. The ERIM, on the other hand, requires the solution of a single “large” NLP problem (see Vectors (7.62) and (7.73)). Moreover, in Problems C2 and C2', the computation of the expected cost (7.75) must be performed with respect to the “big” vector (7.61). As a consequence, we have to pay attention to the dimensions of the vectors \mathbf{w}_n and ξ . The dimension of ξ is $T \cdot \dim(\xi_t)$ (we shall come back on the issue of the dimension of ξ in Remark 7.7). If we assume that $\dim(\mathbf{w}_{tn_i})$ is approximately constant along the decision stages, the dimension of \mathbf{w}_n is $T \cdot \dim(\mathbf{w}_{tn_i})$ for Vector (7.73) and $(T - 1) \cdot \dim(\mathbf{w}_{tn_i})$ for Vector (7.62). Obviously, what has been said above has consequences for computing time, memory requirements, and accuracy, depending on whether ADP or the ERIM is used. An example with a numerical comparison between the two methods will be presented in Sect. 7.7. \triangleleft

Remark 7.7 Once one has decided to replace the control functions μ_t with the FSP functions $\tilde{\mu}(\cdot, \mathbf{w}_t)$, the presence of the initial state \mathbf{x}_0 in the process cost function is unavoidable (see (7.75)). Whenever \mathbf{x}_0 is not known a priori, as in Problem C2', this presence is undesirable and must be removed. An important example of removal of \mathbf{x}_0 by interpreting this vector as a random variable and averaging the cost with respect to it is reported in [51]. In this work, a deterministic continuous-time optimal control problem, stated over a finite-time horizon $[0, t_f]$, is considered under LQ assumptions (the state is perfectly measurable). As is well known, the optimal solution to this problem is given by the control law $\mathbf{u}(t) = -L^\circ(t)\mathbf{x}(t)$, $0 \leq t \leq t_f$, where the gain matrix is determined by solving a Riccati differential equation.

Implementing a controller with a time-varying gain matrix may be quite a hard task. That being the case, a suboptimal controller $\mathbf{u}(t) = -L(t)\mathbf{x}(t)$, $0 \leq t \leq t_f$, is proposed where $L(t)$ has a suitable simple structure. For example, $L(t)$ may be constrained to be piecewise constant over the process length $[0, t_f]$, which is subdivided into M time intervals. As M increases, $L(t)$ approximates $L^\circ(t)$ better and better. If the $d \times d$ matrix $T_L(t)$ is the solution of a certain linear matrix differential equation containing $L(t)$ (the subscript L in $T_L(t)$ means that this matrix depends on $L(t)$), then the process cost is given by

$$\mathbf{x}(0)^\top T^\circ(0)\mathbf{x}(0) \leq \mathbf{x}(0)^\top T_L(0)\mathbf{x}(0),$$

where $T^\circ(0)$ is the initial value of the matrix $T^\circ(t)$, which is the unique symmetric positive-definite solution of the Riccati equation related to the problem at hand.

Suppose now that we want to approximate $L(t)$ with a gain matrix, which has to be constant over $[0, t_f]$, i.e., $L(t) = L$. Then, what we did in stating Problem C2'_n is similar to the approach proposed in [51]. The optimal constant gain matrix L^* is obtained by solving the following NLP problem:

$$L^* = \arg \min_L \text{tr}[T_L(0)],$$

where $\text{tr}[T_L(0)]$ is independent of $\mathbf{x}(0)$ and can be interpreted as follows. Suppose $\mathbf{x}(0)$ to be a random vector uniformly distributed over the surface of a d -dimensional unit hypersphere centered in the origin. It is easy to show that minimizing $\text{tr}[T_L(0)]$ is equivalent to minimizing the expected value of $\mathbf{x}(0)^\top T_L(0)\mathbf{x}(0)$. \triangleleft

7.6 Solution of Problems C2_n and C2'_n

For the solution of Problems C2_n and C2'_n there are the NLP methods described briefly in Sect. 2.3 and more extensively in Sect. 5.3 about the solution of Problem P_n. In this section, we first address some of these methods by referring to Problem C2_n, since all that is said about this problem can be immediately extended to Problem C2'_n (see Sect. 7.6.1). Indeed, the latter is simpler in the notation. Then, in Sect. 7.6.2, we shall concentrate on gradient-based algorithms and we shall detail the algebra for computing the gradient $\nabla_{\mathbf{u}_0, \mathbf{w}_n} J(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi)$ (see (7.60)). Particular attention is paid to stochastic approximation that enables one to avoid the computation of the expected cost \hat{J} and of its gradient. Finally, in Sect. 7.6.3 the general formulas for computing $\nabla_{\mathbf{u}_0, \mathbf{w}_n} J(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi)$ shall be specialized to the case in which feedforward multilayer neural networks are adopted as FSP functions.

7.6.1 Solution of Problem C2_n by Nonlinear Programming

In Remark 7.4, we highlighted the difficulties in computing the set W_n . In this section and in the first part of the next one we shall assume Problems C2_n and C2'_n to be unconstrained. This assumption will be removed at the end of Sect. 7.6.2. We note at first that, for a given value of n , the cost $\tilde{F}(\mathbf{w}_n) = \underset{\mathbf{z}}{\mathbb{E}} J(\mathbf{w}_n, \mathbf{z})$ in Problem P_n (see (2.31)) corresponds to the cost $\hat{J}(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0) = \underset{\xi}{\mathbb{E}} J(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi)$ in Problem C2_n (see (7.60)).

The preliminary choice for minimizing \hat{J} is between a *direct search method* and a *gradient-based method*. As regards the former, let Assumption 1.2 be verified for the cost $J(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi)$. The validity of this assumption is based on the fact that State Equation (7.1), Cost (7.3), and the probability density functions are perfectly known. Once Assumption 1.2 is satisfied, one must be sure that the expected value $\hat{J}(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0)$ can be computed with sufficient accuracy. As this expected value (generally) cannot be computed except numerically, one has to resort to MC or quasi-MC techniques. Such techniques give estimates of the form

$$\hat{J}(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0) \simeq \frac{1}{L_{\text{tot}}} \sum_{l=1}^{L_{\text{tot}}} J(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi^{(l)}), \quad (7.77)$$

where L_{tot} is the total number of evaluations of $J(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \boldsymbol{\xi})$ at the sample vectors $\boldsymbol{\xi}^{(1)}, \dots, \boldsymbol{\xi}^{(L_{\text{tot}})}$ along the T -stage stochastic process. Suppose that the number of samples of $\boldsymbol{\xi}_t$ is the same for each stage $t = 0, 1, \dots, T - 1$ and equal to L . Then, $L_{\text{tot}} = T \cdot L$. Note also that $d_{\boldsymbol{\xi}} = \dim(\boldsymbol{\xi}) = T \cdot \dim(\boldsymbol{\xi}_t)$. Both L_{tot} and $d_{\boldsymbol{\xi}}$ may be very large in problems with many decision stages, even if they grow only linearly with T . However, MC and quasi-MC techniques enable one to achieve reasonable levels of accuracy in the estimate of the expected values by generating a number L_{tot} of samples that is not too large and is also independent of the dimension of $\boldsymbol{\xi}$. Let us comment briefly on this issue.

Remark 7.8 As we saw in Sects. 5.1 and 5.2, if MC methods are used, under suitable assumptions, the estimates of integrals and expected values of functions dependent on random variables are characterized by errors of order $O(L_{\text{tot}}^{-1/2})$ as stated in Theorem 5.1. For the same estimates obtained by low-discrepancy sequences in the quasi-MC framework, more favorable bounds of order $O(L_{\text{tot}}^{-1} (\ln(L_{\text{tot}})^{d_{\boldsymbol{\xi}}-1})$ can be achieved (see Koksma–Hlawka Inequality (5.19) and Theorem 4.6; see also (4.106)). In practice, however, for the reasons reported in Sect. 5.14, often the superiority of low-discrepancy sequences is not verified. Anyway, in both MC and quasi-MC sequences, what matters is that the upper bounds on the errors are nearly independent of the dimension $d_{\boldsymbol{\xi}}$ (see (5.20), however, where $c(d)$ is not an absolute constant). Therefore, to achieve a given accuracy, the number L_{tot} of samples is almost independent of $d_{\boldsymbol{\xi}}$. So, as regards the dimension of $\boldsymbol{\xi}_t$ and the number of stages, the ERIM should mitigate the curse of dimensionality. \triangleleft

Of course, the availability of the gradient $\nabla_{\mathbf{u}_0, \mathbf{w}_n} \hat{J}$ (if it exists) can allow one to solve Problem C2_n by more efficient gradient-based algorithms. Some of them were mentioned in Sect. 5.3.2. In Sect. 2.3 and, in more detail, in Sect. 5.3.2, we pointed out that if J is a function of class \mathcal{C}^1 with respect to \mathbf{u}_0 , \mathbf{w}_n , and $\boldsymbol{\xi}$ and if some regularity conditions are verified, then \hat{J} is also a function of class \mathcal{C}^1 with respect to the same variables. More specifically, we have to suppose that Assumption 2.5 is satisfied (in this assumption, replace \mathbf{w}_n with the pair of vectors \mathbf{u}_0 and \mathbf{w}_n , as defined in (7.62), and z with $\boldsymbol{\xi}$, as defined in (7.61)). Then, some other conditions are stated in Theorem 5.2.

We summarize some of the conditions required in the next assumption.

Assumption 7.1 For every $\mathbf{x}_0 \in X_0$, the functions $f_0(\mathbf{x}_0, \mathbf{u}_0, \boldsymbol{\xi}_0)$ and $h_0(\mathbf{x}_0, \mathbf{u}_0, \boldsymbol{\xi}_0)$ are of class \mathcal{C}^1 with respect to \mathbf{u}_0 and $\boldsymbol{\xi}_0$. The functions $f_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t)$ and $h_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t)$, $t = 1, \dots, T - 1$ are of class \mathcal{C}^1 with respect to \mathbf{x}_t , \mathbf{u}_t , and $\boldsymbol{\xi}_t$. The function $h_T(\mathbf{x}_T)$ is of class \mathcal{C}^1 . The FSP functions $\tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t})$, $t = 1, \dots, T - 1$, are of class \mathcal{C}^1 . \triangleleft

If Assumption 7.1 and all the other conditions required by Theorem 5.2 are verified (hence, both J and \hat{J} are \mathcal{C}^1 functions), what we said in Sect. 5.3.2 for Problem P_n can be repeated for Problem C2_n. This means that one can compute both the gradient $\nabla_{\mathbf{u}_0, \mathbf{w}_n} \hat{J}(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0)$ (in general, approximately) and the gradient $\nabla_{\mathbf{u}_0, \mathbf{w}_n} J(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \boldsymbol{\xi})$. Then, if the former gradient is used, one can choose among

gradient-based algorithms; if the latter gradient is used, one can resort to stochastic programming techniques. (In both cases, at the beginning of this section, we have assumed Problem C2_n to be unconstrained.)

By choosing the first of the two options mentioned above (i.e., by using the gradient of \hat{J}), one can resort to the wide class of gradient-based algorithms of the type (5.30), that is,

$$\begin{aligned} \text{col} [\mathbf{u}_0(k+1), \mathbf{w}_n(k+1)] &= \text{col} [\mathbf{u}_0(k), \mathbf{w}_n(k)] \\ -\alpha_k S_k \nabla_{\mathbf{u}_0, \mathbf{w}_n} \underset{\xi}{\mathbb{E}} J[\mathbf{u}_0(k), \mathbf{w}_n(k), \mathbf{x}_0, \xi], \quad k &= 0, 1, \dots \end{aligned} \quad (7.78)$$

For the meaning of α_k , S_k , and some comments on the algorithms of this type, we refer to Sect. 5.3.2. Of course, owing to the very general framework in which Problem C2_n has been stated, we are unable to analytically compute the gradient $\nabla_{\mathbf{u}_0, \mathbf{w}_n} \underset{\xi}{\mathbb{E}} J[\mathbf{u}_0(k), \mathbf{w}_n(k), \mathbf{x}_0, \xi]$ and, in some cases, even the matrix S_k . However, following (5.32) and (5.33), from (7.77) we obtain

$$\begin{aligned} \text{col} [\mathbf{u}_0(k+1), \mathbf{w}_n(k+1)] &= \text{col} [\mathbf{u}_0(k), \mathbf{w}_n(k)] \\ -\alpha_k S_k \frac{1}{L_{\text{tot}}} \sum_{l=1}^{L_{\text{tot}}} \nabla_{\mathbf{u}_0, \mathbf{w}_n} J[\mathbf{u}_0(k), \mathbf{w}_n(k), \mathbf{x}_0, \xi^{(l)}], \quad k &= 0, 1, \dots \end{aligned} \quad (7.79)$$

(we assumed that the sample set $\{\xi^{(l)}\}_{l=1}^{L_{\text{tot}}}$ is the same at any iteration k). As mentioned in Sect. 5.3.2, S_k can range from $S_k = I$ (then the simple steepest descent method is used) to a progressive approximation of the inverse of the Hessian matrix HJ , obtained by using two successive iterations of a quasi-Newton algorithm.

The second choice for the solution of Problem C2_n (i.e., by using the gradient of J instead of the gradient of \hat{J}) consists in resorting to *stochastic gradient algorithms*, which allow one to avoid the approximate computation of the expected cost in (7.78). We recall that this second choice implies switching from a batch processing mode to a sequential one. This means that, instead of (7.79), we use the updating algorithm

$$\begin{aligned} \text{col} [\mathbf{u}_0(k+1), \mathbf{w}_n(k+1)] &= \text{col} [\mathbf{u}_0(k), \mathbf{w}_n(k)] \\ -\alpha_k \nabla_{\mathbf{u}_0, \mathbf{w}_n} J[\mathbf{u}_0(k), \mathbf{w}_n(k), \mathbf{x}_0, \xi(k)], \quad k &= 0, 1, \dots, \end{aligned} \quad (7.80)$$

where the sequence $\xi(k)$, $k = 0, 1, \dots$, is generated randomly according to the known probability density function of ξ , and α_k is a suitably decreasing positive step-size. Algorithm (7.80) is the widely used stochastic gradient algorithm (see (5.50)). Sufficient conditions for the convergence of the general stochastic algorithm (5.45) (with probability 1) are given in Theorem 5.3 and are specialized for the stochastic gradient algorithm (7.80) in Sect. 5.3.8. Actually, the conditions for the convergence of the algorithm are related to the shape of the cost surface $\underset{\xi}{\mathbb{E}} J(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi)$ and are very difficult to assess because of the high complexity of such a surface.

In the numerical examples presented later in this chapter, we take the stepsize

$$\alpha_k = c_1/(c_2 + k), c_1, c_2 > 0 \quad (7.81)$$

(see (5.85) and the discussion in Sect. 5.3.7). In these examples, a momentum term $\beta [\mathbf{w}_n(k) - \mathbf{w}_n(k-1)]$ is added to the right-hand side of (7.80). Some comments on the role played by the momentum are reported in Sect. 5.3.8.

Several techniques exist that accelerate the convergence of Stochastic Gradient Algorithms (7.80). Probably, they would allow a faster convergence in the examples given later.

7.6.2 Computation of the Gradient $\nabla_{\mathbf{u}_0, \mathbf{w}_n} J(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi)$

As we have seen in the previous section, the computation of $\nabla_{\mathbf{u}_0, \mathbf{w}_n} J$ is needed in both the batch processing algorithm (7.79) and the stochastic gradient algorithm (7.80). In the following paragraphs, we shall concentrate on Algorithm (7.80) for notational simplicity. Then, we shall have to compute the components of the row vector $\partial J / \partial \mathbf{u}_0$ and the partial derivatives

$$\frac{\partial J}{\partial w_{tn_t}^i}, \quad t = 1, 2, \dots, T-1, \quad i = 1, 2, \dots, \mathcal{N}(n_t), \quad (7.82)$$

where $w_{tn_t}^i$ is the i th component of the vector \mathbf{w}_{tn_t} and $\mathcal{N}(n_t)$ is its dimension, that is, the number of parameters to be optimized in each FSP function (7.58).

To simplify the notation, in the following formulas we shall drop the subscript n_t in the functions $\tilde{\mu}_{n_t}$ and in the vectors \mathbf{w}_{tn_t} . We shall also drop the index k in Algorithms (7.79) and (7.80). Let us use similar definitions to the ones given in Sect. 6.7.1, that is, $\mathbf{u}_t^{T-1} \triangleq \text{col}(\mathbf{u}_t, \dots, \mathbf{u}_{T-1})$, $\tilde{\mu}_t^{T-1} \triangleq \text{col}(\tilde{\mu}_t, \dots, \tilde{\mu}_{T-1})$, and $\xi_t^{T-1} \triangleq \text{col}(\xi_t, \dots, \xi_{T-1})$. Define and decompose the cost-to-go functions as follows:

$$\begin{aligned} J_t(\mathbf{x}_t, \tilde{\mu}_t^{T-1}, \xi_t^{T-1}) &\triangleq h_t[\mathbf{x}_t, \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t), \xi_t] + \sum_{j=t+1}^{T-1} h_j[\mathbf{x}_j, \tilde{\mu}(\mathbf{x}_j, \mathbf{w}_j), \xi_j] \\ &\quad + h_T(\mathbf{x}_T), \quad t = 0, 1, \dots, T-1. \end{aligned} \quad (7.83)$$

Then, by using the chain rule, we have

$$\begin{aligned} \frac{\partial J}{\partial \mathbf{u}_0} &= \frac{\partial J_0(\mathbf{x}_0, \mathbf{u}_0^{T-1}, \xi_0^{T-1})}{\partial \mathbf{u}_0} = \frac{\partial h_0(\mathbf{x}_0, \mathbf{u}_0, \xi_0)}{\partial \mathbf{u}_0} \\ &\quad + \frac{\partial J_1(\mathbf{x}_1, \mathbf{u}_1^{T-1}, \xi_1^{T-1})}{\partial \mathbf{x}_1} \frac{\partial f_0(\mathbf{x}_0, \mathbf{u}_0, \xi_0)}{\partial \mathbf{u}_0}, \end{aligned} \quad (7.84)$$

and

$$\begin{aligned}
\frac{\partial J}{\partial w_t^i} &= \frac{\partial J_t(\mathbf{x}_t, \mathbf{u}_t^{T-1}, \boldsymbol{\xi}_t^{T-1})}{\partial w_t^i} = \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t)}{\partial \mathbf{u}_t} \frac{\partial \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)}{\partial w_t^i} \\
&\quad + \frac{\partial J_{t+1}(\mathbf{x}_{t+1}, \mathbf{u}_{t+1}^{T-1}, \boldsymbol{\xi}_{t+1}^{T-1})}{\partial \mathbf{x}_{t+1}} \frac{\partial f_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t)}{\partial \mathbf{u}_t} \frac{\partial \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)}{\partial w_t^i} \\
&= \left[\frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t)}{\partial \mathbf{u}_t} + \frac{\partial J_{t+1}(\mathbf{x}_{t+1}, \mathbf{u}_{t+1}^{T-1}, \boldsymbol{\xi}_{t+1}^{T-1})}{\partial \mathbf{x}_{t+1}} \frac{\partial f_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t)}{\partial \mathbf{u}_t} \right] \\
&\quad \cdot \frac{\partial \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)}{\partial w_t^i}, \quad t = 1, \dots, T-1, \quad i = 1, \dots, \mathcal{N}(n_t). \quad (7.85)
\end{aligned}$$

Note that $\frac{\partial J}{\partial w_t^i} = \frac{\partial J_t}{\partial w_t^i}$. The same holds true for the derivatives of J and J_t with respect to other variables.

In order to visually follow the various steps, the chain of blocks depicted in Fig. 7.2a may be helpful, as it renders the alternation of the FSP functions and of the state equation explicit stage after stage. In the neural networks literature (see, for instance [60]), this chain is often called “*unfolded*” control structure. The usual control scheme is shown in Fig. 7.2b (compare it with that in Fig. 7.1).

The partial derivatives of the functions h_t , f_t , and $\tilde{\mu}$ in (7.85) can be computed along the control and state trajectories generated by $\mathbf{u}_0(k)$, $\mathbf{u}_t = \tilde{\mu}[\mathbf{x}_t, \mathbf{w}_t(k)]$, $t = 0, 1, \dots, T-1$, and $\boldsymbol{\xi}_t(k)$, $t = 1, \dots, T-1$, for each step k of Algorithm (7.80). The same can be done for the partial derivatives

$$\lambda_t^\top \triangleq \frac{\partial J_t(\mathbf{x}_t, \mathbf{u}_t^{T-1}, \boldsymbol{\xi}_t^{T-1})}{\partial \mathbf{x}_t} = \frac{\partial J}{\partial \mathbf{x}_t}, \quad t = 1, \dots, T. \quad (7.86)$$

By applying the chain rule again and taking into account (7.83), we obtain

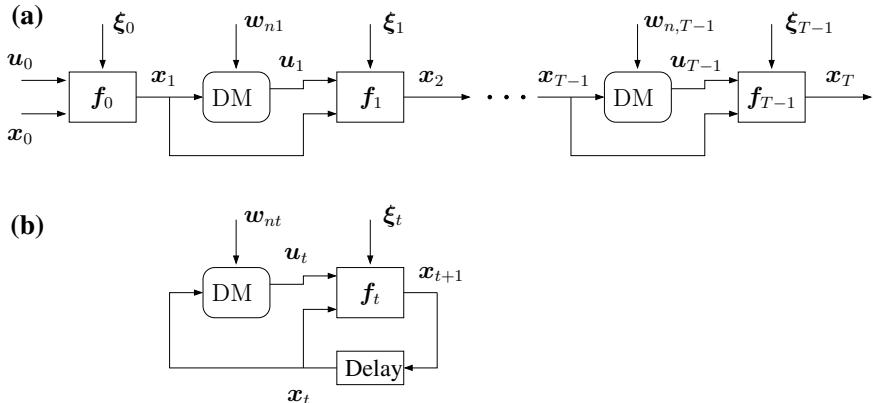


Fig. 7.2 **a** “Unfolded” control scheme; **b** Control scheme

$$\begin{aligned}
\boldsymbol{\lambda}_t^\top &= \frac{\partial h_t[\mathbf{x}_t, \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t), \xi_t]}{\partial \mathbf{x}_t} + \frac{\partial J_{t+1}}{\partial \mathbf{x}_{t+1}} \frac{\partial f_t[\mathbf{x}_t, \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t), \xi_t]}{\partial \mathbf{x}_t} \\
&= \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{x}_t} + \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{u}_t} \frac{\partial \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)}{\partial \mathbf{x}_t} \\
&\quad + \boldsymbol{\lambda}_{t+1}^\top \left[\frac{\partial f_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{x}_t} + \frac{\partial f_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{u}_t} \frac{\partial \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)}{\partial \mathbf{x}_t} \right] \\
&= \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{x}_t} + \boldsymbol{\lambda}_{t+1}^\top \frac{\partial f_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{x}_t} \\
&\quad + \left[\frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{u}_t} + \boldsymbol{\lambda}_{t+1}^\top \frac{\partial f_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{u}_t} \right] \frac{\partial \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)}{\partial \mathbf{x}_t}, \\
&\quad t = 1, \dots, T-1. \tag{7.87}
\end{aligned}$$

By performing the same computation as in (7.84) for $t = 1, \dots, T-1$, we have

$$\frac{\partial J_t(\mathbf{x}_t, \mathbf{u}_t^{T-1}, \xi_t^{T-1})}{\partial \mathbf{u}_t} = \frac{\partial J}{\partial \mathbf{u}_t} = \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{u}_t} + \boldsymbol{\lambda}_{t+1}^\top \frac{\partial f_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{u}_t}, \\
t = 0, 1, \dots, T-1. \tag{7.88}$$

To sum up, the partial derivatives (7.82) needed for Algorithms (7.79) and (7.80) can be computed by (7.84) and the following relationships, derived by substituting (7.88) into (7.85) and (7.87),

$$\frac{\partial J}{\partial w_t^i} = \frac{\partial J}{\partial \mathbf{u}_t} \frac{\partial \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)}{\partial w_t^i}, \quad i = 1, \dots, \mathcal{N}(n_t), \quad t = 1, \dots, T-1, \tag{7.89}$$

$$\boldsymbol{\lambda}_t^\top = \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{x}_t} + \boldsymbol{\lambda}_{t+1}^\top \frac{\partial f_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{x}_t} + \frac{\partial J}{\partial \mathbf{u}_t} \frac{\partial \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)}{\partial \mathbf{x}_t}, \quad t = T-1, \dots, 1, \tag{7.90a}$$

$$\boldsymbol{\lambda}_T^\top = \frac{\partial h_T(\mathbf{x}_T)}{\partial \mathbf{x}_T}, \tag{7.90b}$$

where (7.90b) follows from Definition (7.86).

Therefore, Algorithm (7.80) develops up to a possible convergence according to the steps of the following procedure.

Procedure 7.1

- Guess the initial values of the control $\mathbf{u}_0(0)$ and of the parameter vector $\mathbf{w}(0)$. Set $k = 0$.
- (*) Randomly generate the vector $\xi(k)$ according to its probability density function.

- Starting from $\mathbf{x}_0 = \hat{\mathbf{x}}$, compute the control trajectory $\mathbf{u}_1, \dots, \mathbf{u}_{T-1}$ and the state trajectory $\mathbf{x}_1, \dots, \mathbf{x}_T$ by using \mathbf{u}_0 , the control functions $\tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)$ ($t = 1, \dots, T - 1$), and the state equation (7.1) (*forward pass*).
- Along such trajectories, determine the partial derivatives $\frac{\partial h_t}{\partial \mathbf{x}_t}, \frac{\partial f_t}{\partial \mathbf{x}_t}, \frac{\partial h_t}{\partial \mathbf{u}_t}, \frac{\partial f_t}{\partial \mathbf{u}_t}, \frac{\partial \tilde{\mu}}{\partial \mathbf{x}_t}$.
- Compute λ_T by (7.90b).
- Compute $\lambda_{T-1}, \dots, \lambda_1$ by (7.90a) (*backward pass*).
- Compute the partial derivatives $\frac{\partial J}{\partial \mathbf{u}_t}$ by (7.88); then, $\frac{\partial J}{\partial \mathbf{u}_0}$ is obtained.
- Compute the partial derivatives $\frac{\partial J}{\partial w_t^i}$ by (7.89). Then, all the components of the gradient $\nabla_{\mathbf{u}_0, \mathbf{w}_n} J[\mathbf{u}_0(k), \mathbf{w}_n(k), \mathbf{x}_0, \xi(k)]$ are available.
- Compute the “new” vectors $\mathbf{u}_0(k+1)$ and $\mathbf{w}_n(k+1)$ by using Algorithm (7.80).
- If some termination criterion is satisfied, then stop, otherwise
- Set $k \leftarrow k + 1$ and go to (*). \triangleleft

With some minor changes, the above procedure can be adapted to Algorithm (7.79).

For the solution of a Problem C2_n', i.e., to compute the FSP optimal control functions

$$\mathbf{u}_t^\circ = \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{t_n}^\circ), \quad \mathbf{x}_0 \in X_0, \quad t = 0, 1, \dots, T - 1, \quad (7.91)$$

one may resort again to algorithms similar to (7.79) and (7.80), replacing \mathbf{u}_0 with the weight vector \mathbf{w}_{0n_0} . In the case of Stochastic Gradient Algorithm (7.80) (mostly used in the remainder of chapter), we have

$$\mathbf{w}_n(k+1) = \mathbf{w}_n(k) - \alpha_k \nabla_{\mathbf{w}_n} J[\mathbf{w}_n(k), \xi(k), \mathbf{x}_0(k)], \quad k = 0, 1, \dots, \quad (7.92)$$

where the sequences $\{\xi(k)\}$ and $\{\mathbf{x}_0(k)\}$ are randomly generated on the basis of their probability density functions. As regards the components of the gradient $\nabla_{\mathbf{w}_n} J(\mathbf{w}_n, \xi, \mathbf{x}_0)$, they can be derived by following the same computations that led to Procedure 7.1. Of course, one has now to compute the partial derivatives $\frac{\partial J}{\partial \mathbf{w}_{0n_0}}$ instead of $\frac{\partial J}{\partial \mathbf{u}_0}$. These can be obtained by including the step $t = 0$ in (7.89) and in the other relationships connected with (7.89).

As anticipated at the beginning of Sect. 7.6.1, we now consider the case in which the discrete-time versions of Equalities and Inequalities (1.15) are present. For simplicity, we focus on Inequalities (7.65) addressing Problem C2_n. Among the most suitable methods for the particular structure of this problem, there are certainly those based on *penalty functions*. So, to facilitate the exposition, we shall consider such methods without too much loss of generality. Of course, this means interpreting the constraints in a “soft” way.

Let $P_t(\mathbf{u}_t)$ be a penalty function at the stage t on the vector \mathbf{u}_t . Among the many possibilities, a penalty function frequently used is

$$P_t(\mathbf{u}_t) \triangleq \frac{1}{2} \sum_{j=1}^{\tilde{m}_t} \max \left[0, g_t^{(j)}(\mathbf{u}_t) \right]^2, \quad t = 1, \dots, T-1, \quad (7.93)$$

where $\mathbf{g}_t(\mathbf{u}_t) = \text{col}[g_t^{(1)}(\mathbf{u}_t), \dots, g_t^{(\tilde{m}_t)}(\mathbf{u}_t)]$. Then, we add the penalty function (7.93), weighted by a positive scalar c to the summation of Cost (7.5) and we replace the control function $\boldsymbol{\mu}_t(\mathbf{x}_t)$ with the FSP function (7.58). The new cost is given by

$$\hat{J}_P(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, c) \triangleq \underset{\xi}{\mathbb{E}} J_P(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi, c), \quad (7.94)$$

(compare with (7.60)), where

$$\begin{aligned} J_P(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi, c) = & h_0(\mathbf{x}_0, \mathbf{u}_0, \xi_0) + \sum_{t=1}^{T-1} \{ h_t[\mathbf{x}_t, \tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{w}_{tn_t}), \xi_t] \\ & + c P_t[\tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{w}_{tn_t})] \} + h_T(\mathbf{x}_T). \end{aligned} \quad (7.95)$$

Compare (7.95) with (7.83) for $t = 0$. As is known, c is the “penalty parameter.” This positive parameter must be repeatedly monotonically increased (starting from a relatively small value) until the “new” cost is “close” to the previous one.

One can see that the penalty terms inside the summation do not change the structure of Cost (7.95) with respect to the cost $J(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi)$. Then, what is said in Sects. 7.5 and 7.6 does not require conceptual modifications. For example, if one wants to analytically compute the gradient $\nabla_{\mathbf{u}_0, \mathbf{w}_n} J_P(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi, c)$ to use it in Algorithms (7.79) and (7.80) (of course, we have to suppose, among other assumptions, that the penalty functions (7.93) are of class \mathcal{C}^1 with respect to \mathbf{u}_t ; see Assumption 7.1), a variation of Eq. (7.90) is needed. This equation becomes

$$\begin{aligned} \boldsymbol{\lambda}_t^\top = & \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{x}_t} + c \frac{\partial P_t(\mathbf{u}_t)}{\partial \mathbf{u}_t} \frac{\partial \tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{w}_{tn_t})}{\partial \mathbf{x}_t} \\ & + \boldsymbol{\lambda}_{t+1}^\top \frac{\partial f_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{x}_t} + \frac{\partial J}{\partial \mathbf{u}_t} \frac{\partial \tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{w}_{tn_t})}{\partial \mathbf{x}_t}, \\ & t = T-1, \dots, 1, \\ \boldsymbol{\lambda}_T^\top = & \frac{\partial h_T(\mathbf{x}_T)}{\partial \mathbf{x}_T}. \end{aligned}$$

Similar reasoning can be repeated if one chooses *barrier functions*. Remaining in the case of a “soft” interpretation of the constraints, one can also resort to *augmented Lagrangian methods*. If the equality constraints $\mathbf{g}_t(\mathbf{u}_t) = \mathbf{0}$ are involved, this implies adding terms of the form

$$\tilde{\lambda}_t^\top \mathbf{g}_t(\mathbf{u}_t) + \frac{1}{2} c |\mathbf{g}_t(\mathbf{u}_t)|^2 \quad (7.96)$$

(quadratic penalties are chosen) in the summation of Cost (7.5) (compare with (7.95)). The notation $\tilde{\lambda}_t^\top$ has been used to avoid confusion with the vectors (7.86). If inequalities are present, they can be converted into equalities by introducing additional squared slack variables. After replacing the control functions $\mu(\mathbf{x}_t)$ with the FSP functions (7.58), Cost (7.60) takes on the form

$$\hat{J}_L(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, c, \tilde{\lambda}) = \underset{\xi}{\mathbb{E}} J_L(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi, c, \tilde{\lambda}),$$

where $\tilde{\lambda} \triangleq \text{col}(\tilde{\lambda}_1, \dots, \tilde{\lambda}_{T-1})$ and J_L is the cost (7.95) with the term (7.96) in place of a penalty function of type (7.93). For the generation of the penalty parameters $\{c^k\}$ and the multipliers $\{\tilde{\lambda}^k\}$ to approach the minimum, we refer, for instance, to [9, Sect. 4.2].

If one chooses batch algorithms as (7.79), leaves out approaches like Procedure 7.1, and prefers to interpret the constraints in a “hard” way, then efficient methods exist based on *exact penalty functions*. One of such methods is *sequential quadratic programming*. Other methods were mentioned in Sect. 5.3.2. We do not deepen the issue of the hard interpretation of the constraints as, in the examples of the book, we shall use methods based on simple penalty functions.

Remark 7.9 It is worth noting that (7.90) is the classic *adjoint equation* for T -stage deterministic optimal control (see (6.87)) with the addition of one term (the third in (7.90a)) taking into account the introduction of the FSP control functions. Typically, terms of this kind will appear in other IDO problems approximately solved by the ERIM. This will be clear in the next chapters. The presence of this term is consistent with intuition. If we consider Definition (7.86) and give a look at the scheme in Fig. 7.2a, we see that a variation of \mathbf{x}_t affects the block \mathbf{f}_t in two ways: one is direct, and the other is associated with the block $\tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t})$. Analogously, a variation of \mathbf{x}_t affects the transition cost $h_t[\mathbf{x}_t, \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t}), \xi_t]$, and hence the cost J_t in the same way. Moreover, bear in mind that, when computing the partial derivatives (7.82) at the iteration step k of Algorithms (7.80) and (7.92), the random vectors $\xi_0, \xi_1, \dots, \xi_{T-1}$ have to be considered as constant at the values generated by their probability density functions.

Therefore, the partial derivatives of the cost J with respect to \mathbf{x}_t (keeping the vector \mathbf{x}_t fixed in $\tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t})$) yield the first two terms in (7.90a), which are identical to the right-hand side of Adjoint Eq. (6.87a) (apart from the presence of the constant vector ξ_t). The partial derivatives of J with respect to \mathbf{x}_t have to be determined by taking into account the dependence of J on \mathbf{x}_t also through the control function $\tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t})$. Such derivatives give rise to the third term in (7.90a). Its computation can be easily performed once the kind of FSP function has been chosen. In the next section, we shall compute this third term in the case of multi-hidden-layer (MHL) networks. \triangleleft

Remark 7.10 In deriving (7.89), notice that the mutual independence of the random vectors $\xi_0, \xi_1, \dots, \xi_{T-1}$ is not required. Actually, if the random vectors are not mutually independent, we are in the presence of a problem that differs from Problems C2 or C2'. For this different problem, the ERIM is able to provide a suboptimal solution without any change in its computational procedure. Clearly, the solution may be suboptimal because, at the stage t , the DM does not exploit all the information it has acquired up to this stage, which may be no longer summarized by \mathbf{x}_t (then \mathbf{x}_t would cease to be a state vector). Indeed, the control functions should take on the form $\mathbf{u}_t = \mu_t(\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_t)$ instead of $\mathbf{u}_t = \mu_t(\mathbf{x}_t)$ and the related FSP control functions should be $\mathbf{u}_t = \tilde{\mu}(\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_t, \mathbf{w}_{tn_t})$. \triangleleft

7.6.3 Computation of the Gradient $\nabla_{\mathbf{u}_0, \mathbf{w}_n} J(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi)$ When MHL Networks Are Used

In this section, we specialize the computation of $\nabla_{\mathbf{u}_0, \mathbf{w}_n} J(\mathbf{u}_0, \mathbf{w}_n, \mathbf{x}_0, \xi)$ for the case in which MHL networks are used as FSP functions. We consider the solution of Problem C2_n again. As we said in the previous section, the obvious changes that have to be made, when Problem C2'_n is addressed, imply only replacing \mathbf{u}_0 with the MHL network $\tilde{\mu}(\mathbf{x}_0, \mathbf{w}_{0n_0})$, $\mathbf{x}_0 \in X_0$.

The partial derivatives (7.82) can be determined by (7.88), (7.89), and (7.90). The structures of such relationships are completely independent of the choice of the FSP functions, whose presence in (7.89) and (7.90) is enhanced by the Jacobian matrices $\frac{\partial \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)}{\partial w_t^i}$ and $\frac{\partial \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)}{\partial \mathbf{x}_t}$, respectively. The computation of these matrices may involve a tedious algebra that sometimes can be avoided, at least in part, by the particular architecture of the chosen FSP functions. This is the case for MHL networks, as we shall see in this section.

We address the general multi-hidden-layer case because, as mentioned previously, MHL networks may provide better performances in comparison with OHL neural networks. Assume, for simplicity, that each of the $T - 1$ MHL networks, implementing the control functions $\tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t})$, $t = 1, \dots, T - 1$, is composed of L layers. The structure of each network has been described in Sect. 3.3. We describe such a structure again in order to introduce the stages t and to make the dependence between the state \mathbf{x}_t and the control \mathbf{u}_t explicit. Then, Input/Output Mappings (3.12) and (3.13) are replaced by

$$y_q^t(s) = h[z_q^t(s)], \quad s = 1, \dots, L, \quad q = 1, \dots, n_s, \quad (7.97)$$

and

$$z_q^t(s) = \sum_{p=1}^{n_{s-1}} w_{pq}^t(s) y_p^t(s-1) + w_{0q}^t(s). \quad (7.98)$$

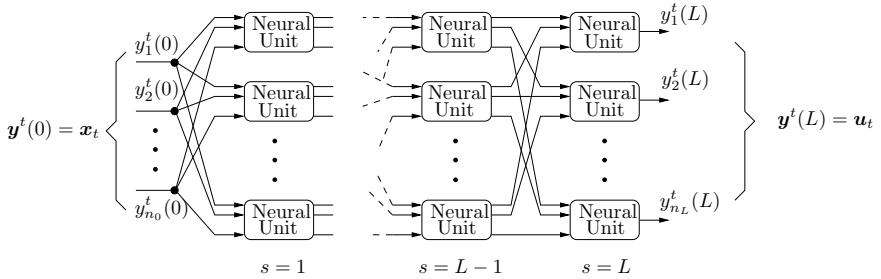


Fig. 7.3 Scheme of a multi-hidden-layer neural network generating the control vector \mathbf{u}_t as a function of the state \mathbf{x}_t

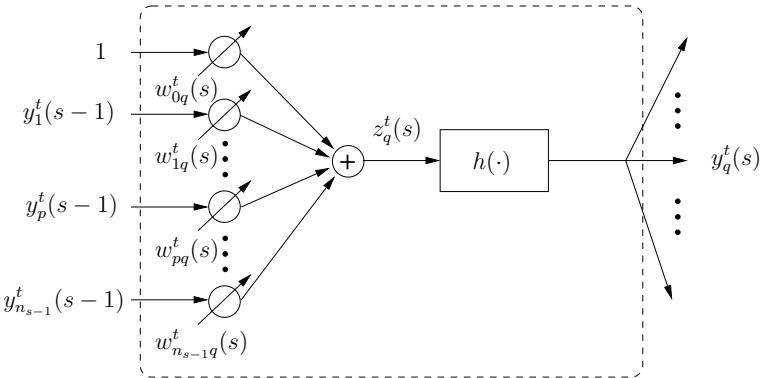


Fig. 7.4 A neural unit showing variable and weight notation

Figures 3.2 and 3.3 are replaced by Figs. 7.3 and 7.4, respectively.

By implementing the control functions $\tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn})$ by MHL networks, we have (see Fig. 7.2a)

$$\mathbf{u}_t = \mathbf{y}^t(L) \triangleq \text{col} [y_1^t(L), \dots, y_m^t(L)], \quad t = 1, \dots, T-1, \quad (7.99)$$

$$\mathbf{x}_t = \mathbf{y}^t(0) \triangleq \text{col} [y_1^t(0), \dots, y_n^t(0)], \quad t = 1, \dots, T-1, \quad (7.100)$$

where $m = \dim(\mathbf{u}_t) = n_L$ and $d = \dim(\mathbf{x}_t) = n_0$

A computational advantage of MHL networks consists in the possibility of directly deriving each component $\frac{\partial J}{\partial w_{pq}^t(s)}$ of the gradient $\nabla_{\mathbf{u}_0, \mathbf{w}_n} J$ by using the chain rule in a scalar way, without the need to explicitly determine the Jacobian matrices $\frac{\partial \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)}{\partial w_t^i}$ and $\frac{\partial \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)}{\partial \mathbf{x}_t}$, as required in (7.89) and (7.90a), respectively. Let us see how this can be carried out. From (7.98) we can write

$$\frac{\partial J}{\partial w_{pq}^t(s)} = \frac{\partial J_t}{\partial w_{pq}^t(s)} = \frac{\partial J}{\partial z_q^t(s)} \frac{\partial z_q^t(s)}{\partial w_{pq}^t(s)} = y_p^t(s-1) \delta_q^t(s), \quad (7.101)$$

where

$$\delta_q^t(s) \triangleq \frac{\partial J}{\partial z_q^t(s)}, \quad t = 1, \dots, T-1, \quad s = 1, \dots, L, \quad q = 1, \dots, n_s. \quad (7.102)$$

Of course, (7.101) implies that the partial derivatives with respect to the bias weights $w_{0q}^t(s)$ are obtained by setting the corresponding inputs to one. From (7.97) we also have

$$\delta_q^t(s) = \frac{\partial J}{\partial y_q^t(s)} \frac{\partial y_q^t(s)}{\partial z_q^t(s)}. \quad (7.103)$$

If we pass from layer s to layer $s+1$ and we take into account (7.98) for this layer, we obtain

$$\frac{\partial J}{\partial y_q^t(s)} = \sum_{k=1}^{n_{s+1}} \frac{\partial J}{\partial z_k^t(s+1)} \frac{\partial z_k^t(s+1)}{\partial y_q^t(s)} = \sum_{k=1}^{n_{s+1}} \delta_k^t(s+1) w_{qk}^t(s+1). \quad (7.104)$$

Substitution of (7.104) into (7.103) yields (see (7.103))

$$\delta_q^t(s) = h'[z_q^t(s)] \sum_{k=1}^{n_{s+1}} \delta_k^t(s+1) w_{qk}^t(s+1), \\ t = 1, \dots, T-1, \quad s = 1, \dots, L, \quad q = 1, \dots, n_s. \quad (7.105a)$$

$$\delta_q^t(L) = h'[z_q^t(L)] \frac{\partial J}{\partial y_q^t(L)}, \quad t = 1, \dots, T-1, \quad q = 1, \dots, n_s. \quad (7.105b)$$

Therefore, the knowledge of the partial derivative $\frac{\partial J}{\partial y^t(L)} = \frac{\partial J}{\partial \mathbf{u}_t}$ (see (7.99)) and the use of (7.105) enable one to compute all the variables $\delta_q^t(s)$ and then the components $\frac{\partial J}{\partial w_{pq}^t(s)}$ of the gradient $\nabla_{\mathbf{u}_0, \mathbf{w}_n} J$ backward, through each MHL network (for $t = 1, \dots, T-1$). Equations (7.105) are the formulas of the popular *backpropagation* procedure. The origin of this procedure is to be credited to several authors. See also [83] for the derivation of the partial derivatives of the cost with respect to weight parameters in complicated architectures made of several MHL networks. The method proposed to compute each term $\frac{\partial J}{\partial w_{pq}^t(s)}$ obtains the partial derivatives via block diagram manipulation rules without a single chain rule expansion.

In order to compute $\frac{\partial J}{\partial \mathbf{u}_t}$, we can use (7.88), which, in turn, requires the derivation of the vectors $\lambda_1, \dots, \lambda_T$. Such vectors can be determined by using (7.90). As we

remarked previously, it is important to note that the third term of (7.90a) denotes the sensitivity of the cost J to the variation of \mathbf{x}_t when this vector causes a variation of \mathbf{u}_t by acting as input to the MHL network $\tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)$ (indeed, a variation of \mathbf{x}_t also produces a variation of J by acting as argument of the function $\mathbf{x}_{t+1} = \mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)$; this can be visually appreciated in Fig. 7.2). Then, we can write the third term of (7.90a) as follows:

$$\frac{\partial J}{\partial \mathbf{u}_t} \frac{\partial \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)}{\partial \mathbf{x}_t} = \frac{\partial J}{\partial \mathbf{y}^t(0)}, \quad t = 1, \dots, T-1. \quad (7.106)$$

Equation (7.90) becomes

$$\boldsymbol{\lambda}_t^\top = \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{x}_t} + \boldsymbol{\lambda}_{t+1}^\top \frac{\partial \mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{x}_t} + \frac{\partial J}{\partial \mathbf{y}^t(0)}, \quad t = 1, \dots, T-1, \quad (7.107a)$$

$$\boldsymbol{\lambda}_T^\top = \frac{\partial h_T(\mathbf{x}_T)}{\partial \mathbf{x}_T}. \quad (7.107b)$$

The term $\frac{\partial J}{\partial \mathbf{y}^t(0)}$ does not depend on the choice of the FSP control function $\tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)$, provided that $\mathbf{y}^t(0)$ is interpreted as the input vector to the block representing such a function. From this, however, one can see the computational efficiency of MHL networks. By using (7.104) and Definition (7.102), one can write the p th component of $\frac{\partial J}{\partial \mathbf{y}^t(0)}$ as

$$\frac{\partial J}{\partial y_p^t(0)} = \sum_{q=1}^{n_1} \frac{\partial J}{\partial z_q^t(1)} \frac{\partial z_q^t(1)}{\partial y_p^t(0)} = \sum_{q=1}^{n_1} \delta_q^t(1) w_{pq}^t(1), \quad t = 1, \dots, T-1, \quad p = 1, \dots, n_0, \quad (7.108)$$

where the variables $\delta_q^t(1)$ can be obtained at the last step of Backpropagation Eq. (7.105) for $s = 1$.

The computation of the gradient $\nabla_{\mathbf{u}_0, \mathbf{w}_n} J$ is completed by the determination of the partial derivative $\frac{\partial J}{\partial \mathbf{u}_0}$. This computation can be performed by using (7.88) for $t = 0$ once the vector $\boldsymbol{\lambda}_1$ has been derived.

Let us now specialize Procedure 7.1 to the case of MHL networks and suppose that Stochastic Gradient Algorithm (7.80) is used. The computational steps, according to which this algorithm is implemented at each iteration step k , are shown pictorially in Fig. 7.5.

Procedure 7.2

- Guess initial values of the control $\mathbf{u}_0(0)$ and of the parameter vector $\mathbf{w}(0)$. Set $k = 0$.

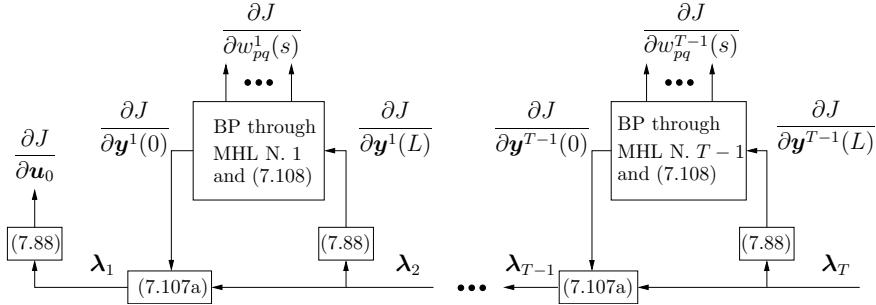


Fig. 7.5 Computation of the gradient $\nabla_{\mathbf{u}_0, \mathbf{w}_n} J[\mathbf{u}_0(k), \mathbf{w}_n(k), \mathbf{x}_0, \boldsymbol{\xi}(k)]$ at the iteration step k of Stochastic Gradient Algorithm (7.80)

- Randomly generate the vector $\boldsymbol{\xi}(k)$ according to its probability density function.
- Starting from $\mathbf{x}_0 = \hat{\mathbf{x}}$, compute the control trajectory $\mathbf{u}_1, \dots, \mathbf{u}_{T-1}$ and the state trajectory $\mathbf{x}_1, \dots, \mathbf{x}_T$ by using the control functions $\tilde{\mu}(\mathbf{x}_t, \mathbf{w}_t)$, $t = 1, \dots, T - 1$, and State Equation (7.1) (*forward phase*).
- Along such trajectories, determine the partial derivatives $\frac{\partial h_t}{\partial \mathbf{x}_t}, \frac{\partial f_t}{\partial \mathbf{x}_t}, \frac{\partial h_t}{\partial \mathbf{u}_t}, \frac{\partial f_t}{\partial \mathbf{u}_t}$, and the variables $y_q^t(s), z_q^t(s)$ in the MHL networks.
- Compute λ_T by (7.107b) (beginning of the *backward phase*).

(*) Set $t = T - 1$.

(**) Compute $\frac{\partial J}{\partial \mathbf{y}^t(L)} = \frac{\partial J}{\partial \mathbf{u}_t}$ by (7.88).

- Apply Backpropagation Eq. (7.105) through the MHL network t , compute all the variables $\delta_q^t(s)$ and then the gradient components $\frac{\partial J}{\partial w_{pq}^t(s)}$ by (7.89).

- Compute $\frac{\partial J}{\partial \mathbf{y}^t(0)}$ by (7.108).
- Compute λ_t by (7.107a).
- If $t > 1$, then set $t \leftarrow t - 1$ and go to step (**).
- Compute λ_1 by (7.107a).
- Compute $\frac{\partial J}{\partial \mathbf{u}_0}$ by (7.88). At this point, all the components of the gradient $\nabla_{\mathbf{u}_0, \mathbf{w}_n} J[\mathbf{u}_0(k), \mathbf{w}_n(k), \mathbf{x}_0, \boldsymbol{\xi}(k)]$ are available.
- Compute the “new” vectors $\mathbf{u}_0(k+1)$ and $\mathbf{w}_n(k+1)$ by using Algorithm (7.80).
- If some termination criterion is satisfied, then stop, otherwise
- Set $k \leftarrow k + 1$ and go to (*).

▫

If one chooses to apply Algorithm (7.79) instead of (7.80), Procedure 7.2 can be used after some appropriate modifications.

We now present an example of Problem C2'_n solved by the ERIM, where an LQ optimal control problem is addressed to evaluate the capacity of OHL neural

networks to derive approximate solutions [62]. It is well known that LQ optimal control problems can be solved analytically. Then, in this case, we have the possibility of comparing the solution provided by the ERIM with the exact one.

Example 7.1 Consider the following linear dynamic system:

$$\mathbf{x}_{t+1} = \begin{bmatrix} 0.65 & -0.19 \\ 0 & 0.83 \end{bmatrix} \mathbf{x}_t + \begin{bmatrix} 7 \\ 7 \end{bmatrix} u_t, \quad t = 0, 1, \dots, T-1,$$

where $\mathbf{x}_t \triangleq \text{col}(x_t, y_t)$. The cost function is

$$J = \sum_{t=0}^{T-1} u_t^2 + v_T |\mathbf{x}_T|^2, \quad (7.109)$$

where $v_T = 40$, $T = 10$.

The problem consists in driving the dynamic system from *any* initial state \mathbf{x}_0 , which is assumed to be uniformly distributed on the rectangle (see Fig. 7.6)

$$X_0 = \{(x, y) \in \mathbb{R}^2 : 2.5 \leq x \leq 3.5, -1 \leq y \leq 1\}$$

to the origin. Therefore, the problem is of type C2'. The minimum value of Cost (7.109) can be computed analytically by DP, which gives

$$J_0^\circ(\mathbf{x}_0) = \mathbf{x}_0^\top T_0^\circ \mathbf{x}_0, \quad \forall \mathbf{x}_0 \in X_0. \quad (7.110)$$

The optimal control is generated by the linear feedback law

$$u_t^\circ = -L_t^\circ \mathbf{x}_t, \quad t = 0, 1, \dots, T-1. \quad (7.111)$$

The matrices T_0° and $L_0^\circ, L_1^\circ, \dots, L_{T-1}^\circ$ are determined by solving a discrete-time Riccati equation.

To evaluate the appropriateness of the ERIM for a problem admitting an analytical solution, we considered OHL neural networks (see (7.58))

$$u_t = \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t}), \quad t = 0, 1, \dots, T-1. \quad (7.112)$$

All the networks (7.112) were implemented by using 20 sigmoidal basis functions given by $h(z) = \tanh z$ (see (7.97)). Then, the model complexities n_t were kept time-invariant. We performed several simulations for increasing values of the model complexity n until we noted that the expected cost \hat{J} did not decrease any more. We applied the stochastic gradient algorithm with the momentum term:

$$\begin{aligned} \mathbf{w}_n(k+1) &= \mathbf{w}_n(k) - \alpha_k \nabla_{\mathbf{w}_n} J[\mathbf{w}_n(k), \mathbf{x}_0(k)] \\ &\quad + \beta[\mathbf{w}_n(k) - \mathbf{w}_n(k-1)], \quad k = 1, 2, \dots, \end{aligned}$$

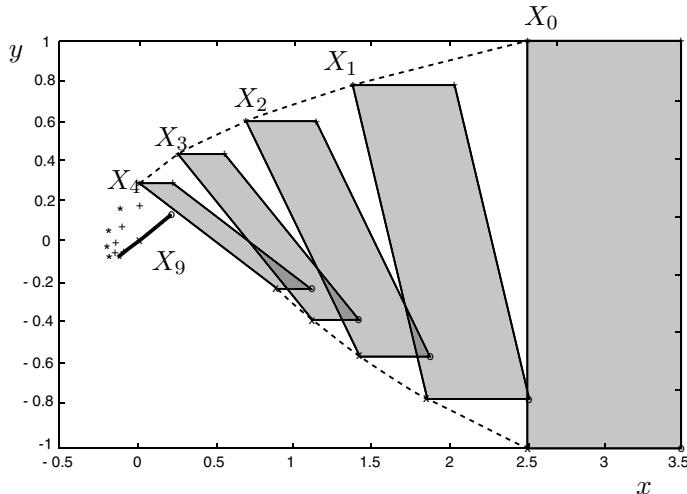


Fig. 7.6 Optimal “neural” state trajectories from X_0 to the region X_7 “neighboring” the origin.
©1994 IEEE. Reprinted, with permission, from [62]

(see (7.80) and (5.89)) and we set $c_1 = 100$, $c_2 = 100000$, and $\beta = 0.8$. The parameters c_1 , c_2 , and β were derived experimentally. More specifically, they were chosen so as to obtain a reasonable trade-off between the convergence speed and a “regular” behavior of the learning algorithm (i.e., a steep descent in the initial part of the learning procedure, low sensitivity to the randomly chosen initial values of the weight vector, etc.). A similar criterion was used to choose such parameters for the examples given later. The algorithm converged to the optimal solution w_n^o after a number of iterations between 10^4 and $2 \cdot 10^4$.

Two optimal neural state trajectories starting from two vertices of the initial region X_0 are shown in Fig. 7.6. In the figure, the region X_0 is mapped into the region X_1 , then X_1 into X_2 , and so on, up to the region X_9 . For clarity, only the first regions are shown.

Since, in Fig. 7.6, the optimal neural and the analytically derived trajectories practically coincide, X_0 , X_4 , and X_9 are plotted in enlarged form in Fig. 7.7 to allow comparisons between the approximate results and the analytical ones. The continuous lines represent the optimal neural control functions (7.112) for different constant values of the control variable u_t (“iso-control” lines), and the dashed lines represent the optimal control functions (7.111). As can be seen, the optimal neural control functions approximate the optimal ones in a very satisfactory way. This occurs not only inside the sets X_0 , X_4 , and X_9 but also outside these regions, thus pointing out the nice generalization properties of the optimal neural control functions. \triangleleft

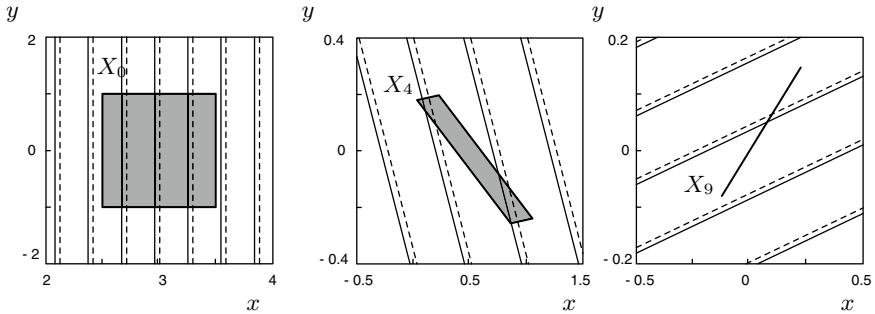


Fig. 7.7 Comparison between the optimal neural control functions (continuous lines) and the optimal control functions (dashed lines). ©1994 IEEE. Reprinted, with permission, from [62]

7.7 Comparison Between the ERIM and ADP

We now consider the optimal management of two water reservoir systems (see Figs. 7.8 and 7.9). The problem consists in determining the optimal releases of water from N_{res} interconnected basins of a network. The cost to be minimized is defined on the basis of suitable objectives such as power generation, irrigation, satisfaction of a minimal level of water in the reservoirs, etc., over a finite horizon of T time intervals or stages. This is a classical problem, which has been solved very often in literature through techniques related to DP [19, 34, 46, 64, 90].

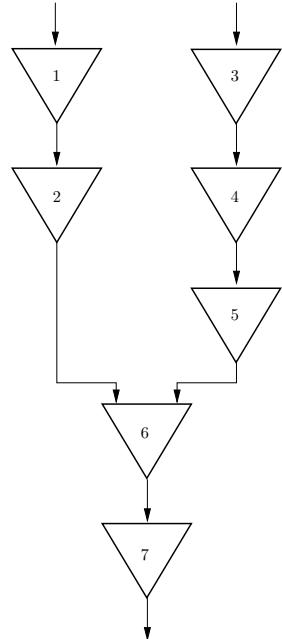
7.7.1 The Model of the Reservoir System

The evolution of the system of reservoirs during the time horizon can be modeled by a dynamic system, in which the amounts of water in the reservoirs are the components of the state vector, and the state equation for the single basin reflects the flow balance between the water that enters and the water that is released in the time interval t . For this purpose, the model must take into account not only the water from the upstream basins but also external inflows due to rain and rivers, which can be modeled in general by dynamic systems with stochastic inputs.

In the following paragraphs, we shall use the notation described below:

- x_t^j : amount of water present in the reservoir $j = 1, \dots, N_{res}$, at the beginning of the stage $t = 0, 1, \dots, T$.
- x_{max}^j : maximum capacity of the reservoir j .
- u_t^j : amount of water released from the reservoir j in the stage $t = 0, 1, \dots, T - 1$. Define $\mathbf{u}_t \triangleq \text{col}(u_t^1, \dots, u_t^{N_{res}})$.
- u_{max}^j : maximum amount of water released from the reservoir j .
- u_{des}^j : desired amount of water released at regime from the reservoir j .

Fig. 7.8 A seven-reservoir system



- $(s_t^j)^+$: amount of external inflows that enter the reservoir j , modeled by a “thresholded” random variable s_t^j (here, $(s_t^j)^+$ denotes $\max(s_t^j, 0)$).
- ξ_t^j : normal random variable acting at stage $t = 0, 1, \dots, T - 1$, $j = 1, \dots, N_{res}$. This variable is the input of the state equation modeling the behavior of the inflow $(s_t^j)^+$, as specified later.
- I_j^+ : set of indexes corresponding to the reservoirs that release water directly into the j th one, $j = 1, \dots, N_{res}$.

1. State Equation

Various models with different formulations of the state equation and the cost function were proposed in literature (see, e.g., [90] and the references therein). Here, we consider a model used in [34] for the optimal management of a water reservoir network, which was also used in [3] in order to compare ADP and the ERIM. The state equation for the j th reservoir has the following form:

$$x_{t+1}^j = \min(x_t^j - u_t^j + \sum_{h \in I_j^+} u_t^h + (s_t^j)^+, x_{max}^j), \\ t = 0, 1, \dots, T - 1, j = 1, \dots, N_{res}. \quad (7.113)$$

Note that the “min” in (7.113) implies that each reservoir has a floodway. This way, the amount of water x_{t+1}^j cannot exceed the maximum value x_{max}^j .

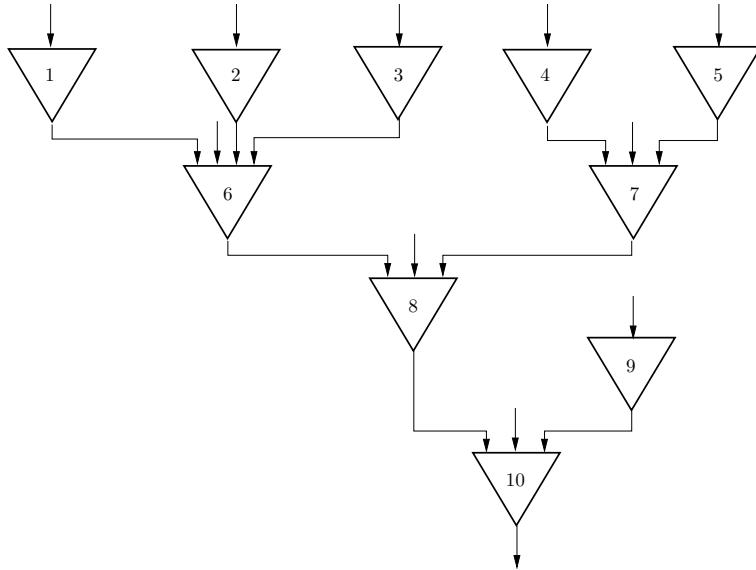


Fig. 7.9 A 10-reservoir system

The stochastic inflows are modeled as autoregressive linear processes of order 2 driven by normal random variables ξ_t^j , followed by a thresholding operation, to obtain nonnegative inflows. Then, we write

$$s_t^j = a_t^j s_{t-1}^j + b_t^j s_{t-2}^j + c_t^j + d_t^j \xi_t^j, \quad t = 0, 1, \dots, T-1, \quad j = 1, \dots, N_{res}, \quad (7.114)$$

where $a_t^j \in \mathbb{R}$, $b_t^j \in \mathbb{R}$, $c_t^j \in \mathbb{R}$, $d_t^j \in \mathbb{R}$.

For a given reservoir j , the random variables ξ_t^j are mutually independent for $t = 0, 1, \dots, T-1$. For a given t , the random variables ξ_t^j for $j = 1, \dots, N_{res}$ are not necessarily mutually independent. This is the case for reservoirs that are considered “close” to each other. The coefficients a_t^j , b_t^j , c_t^j , and d_t^j are derived from [38]. Such coefficients are time-dependent so as to make their dependence on the stages explicit. Thus, we have to add other $2N_{res}$ equations to (7.113). We let $x_t^{N_{res}+j} = s_{t-2}^j$ and $x_t^{2N_{res}+j} = s_{t-1}^j$. Therefore,

$$x_{t+1}^{N_{res}+j} = x_t^{2N_{res}+j}, \quad (7.115)$$

$$\begin{aligned} x_{t+1}^{2N_{res}+j} &= a_t^j x_t^{2N_{res}+j} + b_t^j x_t^{N_{res}+j} + c_t^j + d_t^j \xi_t^j, \\ t &= 0, 1, \dots, T-1, \quad j = 1, \dots, N_{res}. \end{aligned} \quad (7.116)$$

More complex inflow dynamics can be included. For instance, autoregressive linear models of a given order k can be used in order to include the inflows of the past k time periods (see, e.g., [19, 38, 67]). This simply leads to a higher dimensional state vector.

To simplify matters and without loss of generality, in (7.113) we assume that the superficial spill from each reservoir, which occurs when the corresponding storage overflows the maximum storage x_{\max}^j , does not reach the downstream reservoirs.

Grouping Eqs. (7.113), (7.115), and (7.116) and defining $\mathbf{x}_t \triangleq \text{col}(x_t^1, \dots, x_t^{3N_{res}})$, we get the overall nonlinear state equation

$$\mathbf{x}_{t+1} = \mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t), \quad t = 0, 1, \dots, T-1, \quad (7.117)$$

where the initial state $\mathbf{x}_0 = \hat{\mathbf{x}}$ is given. All the components of \mathbf{x}_t are assumed to be perfectly measurable. Therefore, the optimal control problem is a Problem C2.

2. Constraints

Each water release must verify the inequalities

$$0 \leq u_t^j \leq u_{\max}^j, \quad t = 0, 1, \dots, T-1, \quad j = 1, \dots, N_{res}. \quad (7.118)$$

As pointed out in Sect. 7.2, possible difficulties may be encountered in setting constraints on the state when random variables are present. Such constraints take on the form $\mathbf{x}_t \in X_t \subseteq \mathbb{R}^d$, $t = 1, \dots, T-1$. In this example, the amount of water must verify the inequalities

$$0 \leq x_{t+1}^j \leq x_{\max}^j, \quad t = 0, 1, \dots, T, \quad j = 1, \dots, N_{res}. \quad (7.119)$$

The inequalities $x_{t+1}^j \leq x_{\max}^j$ hold true, thanks to the presence of the floodways. Consider now the constraints $x_{t+1}^j \geq 0$. From Eqs. (7.113), we can see that the trajectories of the state variables currently being considered have a stochastic behavior owing to the presence of the random inflows $(s_t^j)^+$. We can also see that the nonnegativity of x_{t+1}^j is ensured by a release verifying the following inequalities:

$$0 \leq u_t^j \leq \min \left(x_t^j + \sum_{h \in I_j^+} u_t^h + (s_t^j)^+, u_{\max}^j \right) \quad (7.120)$$

for any value of $(s_t^j)^+$. In particular, “conservative” or “cautious” DM control functions are obtained by letting $(s_t^j)^+ = 0$ in (7.120). This gives the inequalities

$$0 \leq u_t^j \leq \min \left(x_t^j + \sum_{h \in I_j^+} u_t^h, u_{max}^j \right). \quad (7.121)$$

3. Cost Function

According to [64], at each stage we want to keep the releases as close as possible to a given time-invariant value u_{des}^j , which is assumed to be the correct regime of water amount needed, e.g., for power generation and/or irrigation. Furthermore, we introduce the following requirement: at the end of the control horizon the amount of water in the various reservoirs must be as close as possible to a desired value \tilde{x}_T^j to satisfy, for example, domestic demands. Then, the cost h_t for the single stage has the following form:

$$h_t(x_t, u_t, \xi_t) = \sum_{j=1}^{N_{res}} c_p^j (u_t^j - u_{des}^j)^2, \quad (7.122)$$

where $c_p^j > 0$. The final cost is given by

$$h_T(x_T) = \sum_{j=1}^{N_{res}} (x_T^j - \tilde{x}_T^j)^2.$$

Then, the total cost function, which takes into account the the whole horizon, is

$$J = \sum_{t=0}^{T-1} \sum_{j=1}^{N_{res}} c_p^j (u_t^j - u_{des}^j)^2 + \sum_{j=1}^{N_{res}} (x_T^j - \tilde{x}_T^j)^2. \quad (7.123)$$

The problem of optimal management of reservoir systems can be formalized as a Problem C2: find the optimal control vector \mathbf{u}_0° and the optimal control functions $\mathbf{u}_t^\circ = \mu_t^\circ(\mathbf{x}_t)$, $t = 1, \dots, T - 1$, that minimize the expected value of Cost Function (7.123) subject to Constraints (7.117), (7.119), and (7.121) (Constraint (7.118) is not needed in the problem formulation, since it is implied by Constraint (7.121)). Other commonly used costs can be found in [90]. They do not affect the applicability of the proposed methods.

7.7.2 Computational Aspects

In order to present numerical results concerning the application of ADP and the ERIM, we consider two examples based on the water reservoir model described in

Table 7.1 Examples 7.2 and 7.3: features of the dynamic systems and the solving algorithms for the ERIM

	N_{res}	Inflow model	d	ERIM opt. algorithm
Example 7.2	7	White process	7	Batch (stoch. grad. in [3])
Example 7.3	10	Autoregr. process	30	Batch

Table 7.2 \tilde{x}_T^j : desired amount of water in a reservoir. x_{\max}^j : maximum reservoir capacity. u_{des}^j : desired regime of water release. c_P^j : coefficient in Transition Cost (7.122). u_{max}^j : maximum water release

(a) Example 7.2							(b) Example 7.3					
Res. j	\tilde{x}_T^j	x_{\max}^j	u_{des}^j	c_P^j	u_{max}^j	Res. j	\tilde{x}_T^j	x_{\max}^j	u_{des}^j	c_P^j	u_{max}^j	
1	5	12	1	1.1	12	1	5	12	1	1.1	12	
2	5	12	1	1.2	12	2	5	12	1	1.2	12	
3	5	12	1	1.0	12	3	5	12	1	1.0	12	
4	7	12	1	1.3	12	4	7	12	1	1.3	12	
5	7	12	1	1.1	12	5	7	12	1	1.1	12	
6	7	12	1	1.0	12	6	7	12	1	1.0	12	
7	7	12	1	1.0	12	7	7	12	1	1.0	12	
						8	9	12	1	1.0	12	
						9	9	12	1	1.0	12	
						10	9	12	1	1.0	12	

the previous section. For clarity, the features of the dynamic systems and the solving algorithms for the ERIM are shown in Table 7.1 for Examples 7.2 and 7.3.

The first example (Example 7.2) is simpler: the number of reservoirs is $N_{res} = 7$ and the random inflows before thresholding are not modeled by Linear Autoregressive Processes (7.114) but are given by white normal random processes. Then, the dimension of the state in (7.117) is 7.

In the second example (Example 7.3), the number of basins is $N_{res} = 10$. The random inflows are assumed to be generated by Linear Autoregressive Processes (7.114) of order two, followed by thresholding. Then, $\dim(\mathbf{x}_t) = 10 + 2 \cdot 10 = 30$. The layouts of the two networks of reservoirs are shown in Figs. 7.8 and 7.9, respectively. The main features of the two examples are shown in Table 7.2. In both examples, the number of decision stages is $T = 6$ and the distributions of the white noises ξ_t^j are assumed to be time-invariant. The parameters characterizing the state equations and the cost functions are reported in Table 7.2a and b.

In the seven-reservoir example, we assume that the random inflows are only present in the upstream reservoirs, that is, $\xi_t^j = 0$ for $j \neq 1, 3$. The random variables ξ_t^1 and ξ_t^3 have Gaussian probability densities with $p_t(\xi_t^1) = \mathcal{N}(2, 1.5)$ and $p_t(\xi_t^3) = \mathcal{N}(4, 2.5)$. In the 10-reservoir example, on the contrary, all basins have the stochastic inflows $(s_t^j)^+$. In this case, $p_t(\xi_t^j) = \mathcal{N}(2, 1)$, $j = 1, \dots, 10$. Of course, to avoid

negative values of the inflows, the Gaussians are truncated and their profiles modified so that areas remain unitary.

In applying ADP, OHL neural networks with sigmoidal activation functions $h(z) = \tanh z$ were used to approximate the (approximate) optimal cost-to-go functions \bar{J}_t° , $t = 1, \dots, 5$ (see Problem 7.1). At each stage, each network was trained for this approximation by a sampling scheme of the state space based on low-discrepancy sequences (specifically, Sobol' sequences). For the application of the ERIM, the FSP control functions $\tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn})$, $t = 1, \dots, 5$ (see (7.58)), were also given the form of OHL neural networks with $h(z) = \tanh z$.

Note that, to apply ADP, we have to solve NLP Problems 7.1 and 7.2 and, to apply the ERIM, the NLP Problem C2_n. The ERIM was applied by using a batch NLP algorithm that minimizes the expected value of Cost (7.123). Remember that such an expected value can be approximated by using MC and quasi-MC estimates as explained in Sects. 5.2 and 7.6.1; see (7.77) and (7.79). Note, however, that the constraints given by State Equations (7.113) introduce non-differentiability. This drawback can be easily removed by smoothing techniques (see what was reported in Example 7.2). The *sequential quadratic programming* algorithm (mentioned in Sect. 5.3.2) was used in both Examples 7.2 and 7.3.

For both examples and both ADP and the ERIM, 100 white normal random sequences ξ_t^{jl} , $t = 0, 1, \dots, T - 1$, $l = 1, \dots, 100$, were generated to model the external inflows $(s_t^j)^+$. The simulations were performed for several numbers of neural units in the OHL networks (each network was given the same number n of units). For all sequences ξ_t^{jl} , $l = 1, \dots, 100$, of the simulation runs, the costs J_A^l , $l = 1, \dots, 100$ (obtained by applying the controls derived from ADP), and J_B^l , $l = 1, \dots, 100$ (obtained by applying the ERIM) were derived. Then, the mean costs

$$\hat{J}_A = \frac{1}{100} \sum_{l=1}^{100} J_A^l$$

and

$$\hat{J}_B = \frac{1}{100} \sum_{l=1}^{100} J_B^l$$

were computed, and finally minimized with respect to the parameters of the respective FSP functions. We denote their obtained optimal values, respectively, by \hat{J}_A° and \hat{J}_B° .

The experimental results are shown below. The simulations were performed using an Intel Core2 Duo PC with 2GB of RAM.

Example 7.2 This example refers to the seven-reservoir system depicted in Fig. 7.8. The numerical results are summarized in Table 7.3, where the obtained optimal mean costs \hat{J}_A° and \hat{J}_B° and the corresponding computation times are reported for four values of the number n of neural units. In applying both ADP and the ERIM (to approximate the optimal cost-to-go and the control functions, respectively) we have assumed that the structures of the approximating OHL neural networks are time-invariant. Then, n

is constant. Note that, in the backward phase of ADP, we have to solve Problem 7.1 for the stages $t = T - 1, \dots, 1$, where $T = 6$. Each OHL network has $d = \dim(\mathbf{x}_t) = 7$ and 1 output (the cost-to-go $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t})$). Then, the total number of weight parameters to be optimized (we omit counting the optimization of \mathbf{u}_0) is given by

$$\mathcal{N}_A(n) = (T - 1)[(d + 1)n + (n + 1)] = 45n + 5. \quad (7.124)$$

In applying the ERIM and in comparing it with ADP, we consider a number n of units for each OHL neural control function equal to that used in applying ADP. Taking into account that the number of inputs to each control network is again $d = 7$ and that $\dim(\mathbf{u}_t) = m = 7$, we have

$$\mathcal{N}_B(n) = (T - 1)[(d + 1)n + (n + 1)m] = 75n + 35. \quad (7.125)$$

Table 7.3 shows that, at least in correspondence with the four considered values of n , the obtained optimal mean costs \hat{J}_B° are lower than the obtained optimal mean costs \hat{J}_A° , whereas the computation grows faster with n for the ERIM than for ADP. This may be due to Relationships (7.124) and (7.125), where the coefficient multiplying n is greater in (7.125) than in (7.124).

The boxplots of the costs $J_A^{\circ l}$ and $J_B^{\circ l}$ associated with \hat{J}_A° and \hat{J}_B° , respectively, are shown in Fig. 7.10, for $n = 20$. The boxplots indicate the 25th and the 75th percentiles of the sets of 100 trials, whereas the segment inside the boxplot marks the median. Outliers are also reported outside the boxes. The median for the ERIM is slightly lower than the median for ADP. Similar results can be shown for the other values of n .

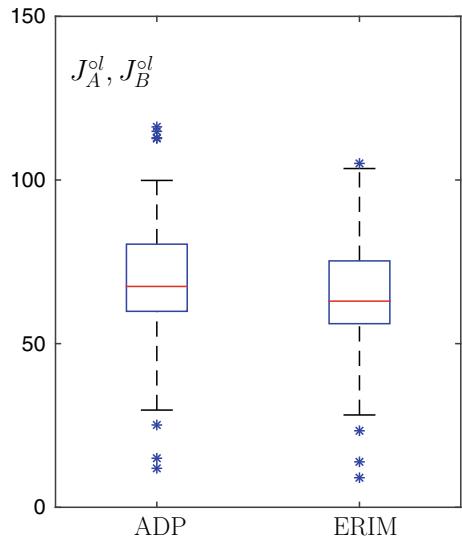
It is interesting to compare Example 7.2 with the results obtained in [3], where the same seven-reservoir configuration was considered. In [3] the ERIM was applied using Stochastic Gradient Algorithm (7.80) instead of a batch algorithm of type (7.79). The stochastic gradient algorithm was applied by computing the components $\partial J / \partial w_{pq}^t(s)$ via Procedure 7.2. This procedure requires the continuous differentiability of the cost J with respect to all the variables on which it depends. Then, as stated previously, the stochastic gradient algorithm can be used in the ERIM after removing the discontinuity of State Equations (7.113) by a simple third-order polynomial approximation (see [3] for details).

The trajectories of two components of the state \mathbf{x}_t ($x_t^{(1)}$ and $x_t^{(3)}$) and of the control \mathbf{u}_t ($u_t^{(1)}$ and $u_t^{(3)}$) are shown in Fig. 7.11 in correspondence with two sequences ξ_t , $t = 0, \dots, 5$, taken among the 100 generated randomly. Note that the trajectories of the state components have a quite similar profile. It is not so for the trajectories of the control components. This might be due to the fact that the neural cost surfaces may be quite “flat” around the minimizing weight vectors (as pointed out previously, the neural surfaces have several global or local minima). Then, computational inaccuracies may cause rather large shifts of such minimizing vectors and thus changes in the OHL networks and in the control variables they generate. \triangleleft

Table 7.3 Summary of the numerical results for some values of the number n of neural units (Example 7.2: 7 reservoirs)

(a) ADP			(b) ERIM		
	Opt. mean costs \hat{J}_A°	Comp. times (s)		Opt. mean costs \hat{J}_B°	Comp. times (s)
$n = 5$	108.4	$1.55 \cdot 10^4$	$n = 5$	69.15	$8.42 \cdot 10^3$
$n = 10$	76.8	$1.95 \cdot 10^4$	$n = 10$	67.07	$2.91 \cdot 10^4$
$n = 15$	71.11	$2.41 \cdot 10^4$	$n = 15$	68.07	$7.40 \cdot 10^4$
$n = 20$	69.77	$2.62 \cdot 10^4$	$n = 20$	64.17	$2.53 \cdot 10^5$

Fig. 7.10 Boxplots of the costs J_A^{ol} and J_B^{ol} , $l = 1, \dots, 100$, associated with \hat{J}_A° and \hat{J}_B° , respectively, for $n = 20$



Example 7.3 This example is again aimed at comparing ADP and the ERIM by referring to the reservoir network with a larger number of state components, namely, the 10-reservoir system depicted in Fig. 7.9. In this case, Problems 7.1 and 7.2 were solved by applying the *Levenberg–Marquardt* algorithm and *sequential quadratic programming*, respectively. The latter algorithm was applied in the ERIM as in Example 7.2.

On the basis of the new values of d and m , Functions (7.124) and (7.125) become

$$\mathcal{N}_A(n) = 160n + 5, \quad (7.126)$$

$$\mathcal{N}_B(n) = 205n + 50. \quad (7.127)$$

The numerical results are reported in Table 7.4, in Figs. 7.12 and 7.13.

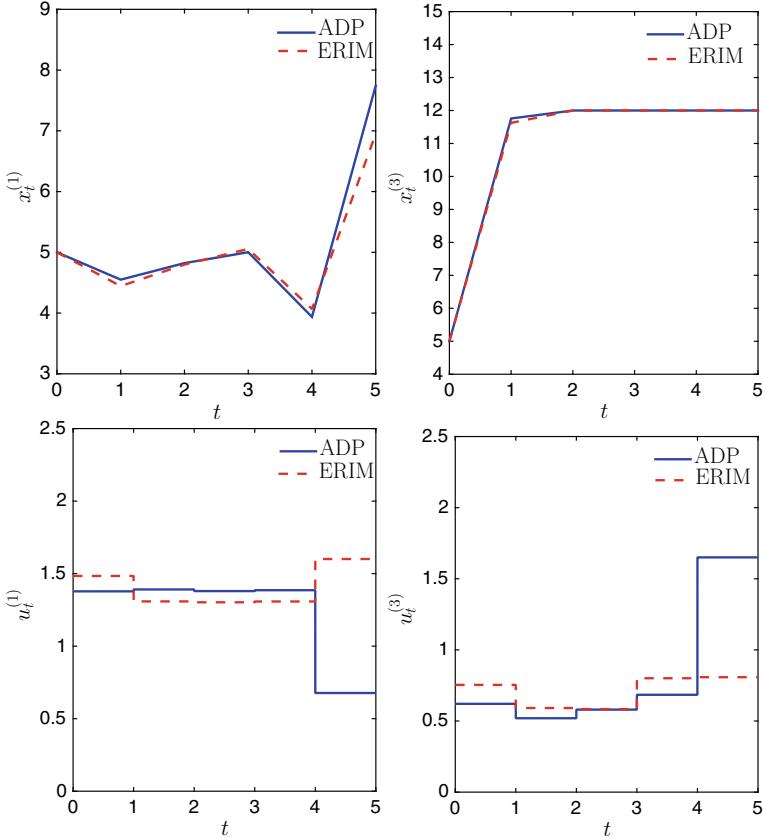


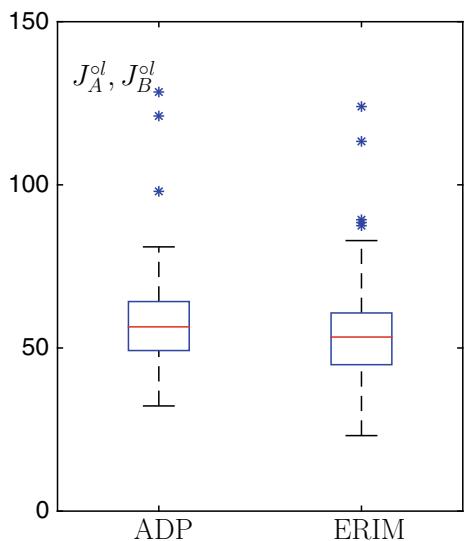
Fig. 7.11 Trajectories of the state components $x_t^{(1)}$ and $x_t^{(3)}$ (amount of water in the reservoir 1 and 3) and of the control components $u_t^{(1)}$ and $u_t^{(3)}$ (amount of water released from the reservoirs 1 and 3) obtained applying ADP and the ERIM, respectively

Considerations similar to those of Example 7.2 can be made. Note, in particular, that in Fig. 7.12 the median of the ERIM is a little lower than the median of ADP but the box is slightly larger. As in Example 7.2, more or less the same was observed for the other values of n . The optimal control of a similar network of 10 reservoirs and 30 state variables is addressed in [19], where ADP was applied and several designs of discretization of the state space were compared, e.g., orthogonal arrays, Latin hypercubes, Sobol, and Niederreiter–Xing low-discrepancy sequences. \triangleleft

Table 7.4 Summary of the numerical results for some values of the number n of neural units (Example 7.3: 10 reservoirs)

(a) ADP			(a) ERIM		
	Opt. mean costs \hat{J}_A°	Comp. times (s)		Opt. mean costs \hat{J}_B°	Comp. times (s)
$n = 5$	82.4	$3.79 \cdot 10^4$	$n = 5$	62.4	$6.15 \cdot 10^4$
$n = 10$	77.6	$3.80 \cdot 10^4$	$n = 10$	60.5	$6.44 \cdot 10^4$
$n = 15$	62.6	$3.94 \cdot 10^4$	$n = 15$	56.3	$9.39 \cdot 10^4$
$n = 20$	57.3	$4.82 \cdot 10^4$	$n = 20$	53.9	$1.24 \cdot 10^5$

Fig. 7.12 Boxplots of the costs J_A^{ol} and J_B^{ol} , $l = 1, \dots, 100$, associated with \hat{J}_A° and \hat{J}_B° , respectively, for $n = 20$



7.7.3 Concluding Comments

On the basis of Examples 7.2 and 7.3, in the following paragraphs we summarize some considerations on the comparison between the ERIM and ADP. Of course, great caution is required since there are only two, rather specific, examples. Moreover, the comparison is limited to the use of OHL neural networks.

From the boxplots in Figs. 7.10 and 7.12, it can be seen that the median is lower in the ERIM but the variance is a little larger. As regards the computation time, Tables 7.3 and 7.4 show that the NLP algorithms converge faster in ADP than in the ERIM. The ERIM should be slower because the number of weight parameters to be optimized to solve Problem C2_n is larger than the number of parameters to solve Problem 7.1 (for simplicity and not to just “privilege” the ERIM vs. ADP, we omit the calculations related to Problem 7.3). The faster growth of the number of parameters in the ERIM with respect to ADP can be seen in (7.124), (7.125),

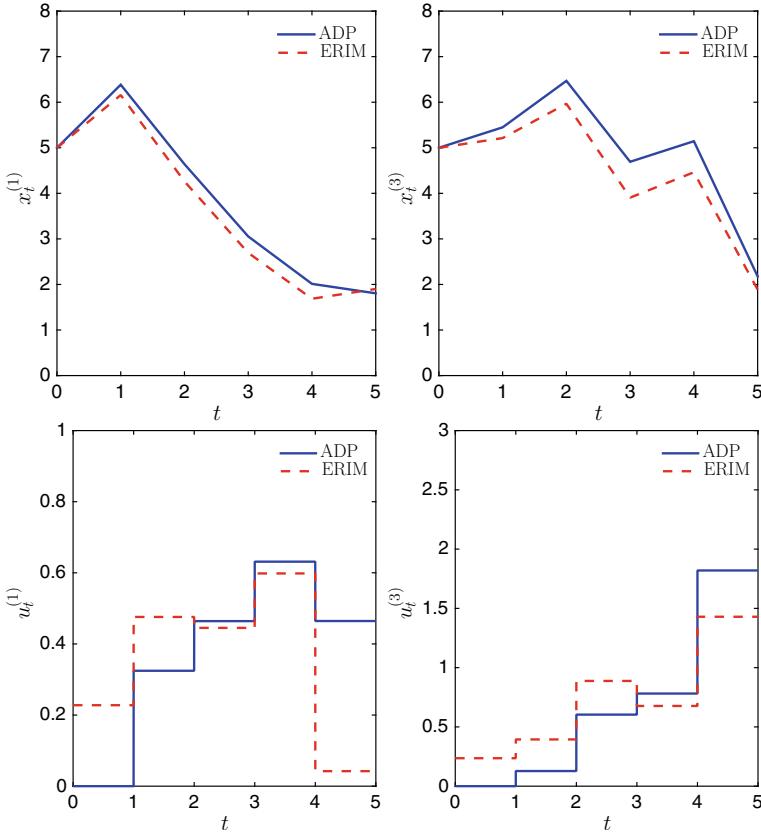


Fig. 7.13 Trajectories of the state components $x_t^{(1)}$ and $x_t^{(3)}$ (amount of water in the reservoirs 1 and 3) and of the control components $u_t^{(1)}$ and $u_t^{(3)}$ (amount of water released from the reservoirs 1 and 3), obtained applying ADP and the ERIM, respectively

(7.126), and (7.127) and is essentially due to the fact that in ADP $m = 1$ whereas in the ERIM $m = \dim(\mathbf{u}_t)$ (in Example 7.2 $m = 7$ and in Example 7.3 $m = 10$). Anyway, the affine dependence of the number of weight parameters on n is a fairly reassuring fact but it does not guarantee that the computation time does not grow too rapidly with n . In general, we can be content to say that the NLP algorithms used are provided with a good order of rate of convergence typical of the quasi-Newton methods.

In applying the ERIM, all the above refers to the case of batch algorithms. It may be interesting to make some comments about [3] where the stochastic gradient algorithm was used. The same model from Example 7.2 is considered, i.e., with seven reservoirs and white normal sequences of external inflows (before thresholding). For some sequences of disturbances, the cost incurred is actually larger in the ERIM than in ADP. If this result were true in general, this fact would be unpleasant.

Fortunately, we have successfully used the stochastic gradient algorithm in the other numerical examples of the book. Of course, possible improvements can be obtained with respect to this algorithm. An improvement may lie in replacing the stochastic gradient algorithm with the *second-order stochastic gradient* (2OSG) algorithm (compare with (7.80) for the case of Problem C2_n)

$$\mathbf{w}_n(k+1) = \mathbf{w}_n(k) - \alpha_k S_k \nabla_{\mathbf{w}_n} J[\mathbf{w}_n(k), \mathbf{x}_0, \boldsymbol{\xi}(k)], \quad k = 0, 1, \dots,$$

where $S_k = S_k^\top > 0$ is a matrix tending to approximate the inverse of the Hessian of J (see Sect. 5.3.2). Convergence conditions of Stochastic Gradient Algorithm (7.80) or (5.50) were considered in Sect. 5.3.3. For the rate of convergence of this algorithm as well as for that of the 2OSG one, we refer, for example, to [18].

In recent decades, stochastic gradient algorithms attracted the attention of a large literature in the area of statistical machine learning. What emerges is that such algorithms may perform better than the ones of type (7.79) (of batch structure) if the computing time creates more difficulties than the need to handle large sample sizes. In [18], all this is summarized by the recommendation “use stochastic gradient when training time is the bottleneck.”

7.8 Stochastic Optimal Control Problems Equivalent to Problem C2'

In applications, it often occurs that other random variables influence the dynamic system and the cost function. Such variables may be measured by the DM or not. If they are measured, we assume the measures to be perfect, i.e., noise-free. For example, it may happen that the DM can take perfect measures on the current random vector $\boldsymbol{\xi}_t$. There is also the possibility that, at a certain stage t , the DM gets to know the values that $\boldsymbol{\xi}_t$ will take on at future instants.

In the next three sections, we address three possible situations: (i) the random variables are time-invariant (i.e., they do not change during the T decision stages), (ii) the random variables are given by $\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_{T-1}$, which are assumed to be measurable, and (iii) the random variables are given by a reference vector to be tracked. For these situations, we consider the application of both ADP and the ERIM. We address Problem C2' as it is better suited to address the various cases. Anyway, transferring the related results to the case of Problem C2 is immediate.

Up to now, the random variables $\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_{T-1}$ have always been considered as mutually independent. It is worth noting that sometimes a possible mutual dependence among such random variables may be modeled by defining an “augmented” dynamic system that includes, besides the basic one (i.e., (7.1)), another dynamic system generating the random variables. The state of the additional system is not always measurable. When it is not, we are no longer in the context of Problem C2'.

Of course, the examples above are only three among a very large number of problems reducible to Problem C2'. We leave the others to the “imagination” of the reader.

7.8.1 Measurable Time-Invariant Random Variables

We address two cases, those in which the time-invariant random variables are given by (i) a vector π of unknown parameters present in the state equation and/or in the cost functional; and (ii) a vector $\mathbf{x}^* \in \mathbb{R}^d$ that the dynamic system has to reach at the final stage T . We shall see that both cases can be stated in terms of Problem C2'.

1. Presence of unknown parameters

An unknown parameter vector π may be present in State Equation (7.1) and/or in Cost Function (7.3). They take on the form

$$\mathbf{x}_{t+1} = \mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t, \pi), \quad \mathbf{x}_0 \in X_0, \quad t = 0, 1, \dots, T-1, \quad (7.128)$$

$$J = \sum_{t=0}^{T-1} h_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t, \pi) + h_T(\mathbf{x}_T, \pi). \quad (7.129)$$

$\pi \in \Pi \subseteq \mathbb{R}^p$ is considered as a random vector with a known probability density function, independent of \mathbf{x}_0 and $\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_{T-1}$. It may be given by physical parameters that influence the dynamic system. In the cost function, π may represent different operational conditions that must be taken into account during the decision process. For example, if the constraints are embedded in Cost (7.3) as penalty functions, different constraints on the admissible controls (see (7.2)) may be included. Analogously to \mathbf{x}_0 , we assume that π is unknown a priori and that it becomes known to the DM at the time $t = 0$. This assumption – as well as other similar ones that will be considered in the following paragraphs – gives reasons for the introduction of Problem C2' instead of Problem C2.

2. The final state \mathbf{x}_T has to approach a vector \mathbf{x}^*

Let us assume that, at the final stage T , the system state has to be driven as close as possible to any desired vector \mathbf{x}^* belonging to a region $X_T \subset \mathbb{R}^d$. As in the previous example, \mathbf{x}^* becomes known to the DM only at the stage $t = 0$. If a cost (7.3) of general form has to be paid, one may add suitable penalty terms, i.e.,

$$J = \sum_{t=0}^{T-1} h_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t) + h_T(\mathbf{x}_T) + \sum_{t=1}^T |\mathbf{x}^* - \mathbf{x}_t|_{V_t}^2, \quad (7.130)$$

where $V_t = V_t^\top \geq 0, t = 1, \dots, T$, are weight matrices that impose the desired accuracy in approaching \mathbf{x}^* . It is reasonable to assume V_t to be diagonal matrices, whose

elements specify different requirements to each component of \mathbf{x}_t . If one is not interested in approaching \mathbf{x}^* before the final stage T , one may set $V_1 = \dots = V_{T-1} = 0$. The hypotheses on \mathbf{x}^* are the same as the ones on \mathbf{x}_0 and $\boldsymbol{\pi}$. This means that $\mathbf{x}^* \in X^* \triangleq X_T \subset \mathbb{R}^d$ is a random vector, independent of the other random vectors, with a known probability density function.

Let us suppose that both vectors $\boldsymbol{\pi}$ and \mathbf{x}^* are present in the optimal control problem. Then, Costs (7.129) and (7.130) are replaced by

$$J = \sum_{t=0}^{T-1} h_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t, \boldsymbol{\pi}) + h_T(\mathbf{x}_T) + \sum_{t=1}^T \|\mathbf{x}^* - \mathbf{x}_t\|_{V_t}^2. \quad (7.131)$$

The presence of the vectors $\boldsymbol{\pi}$ and \mathbf{x}^* can be easily treated by introducing the auxiliary state equation

$$\mathbf{z}_{t+1} = \mathbf{z}_t \triangleq \text{col}(\boldsymbol{\pi}, \mathbf{x}^*), \quad t = 0, 1, \dots, T-1, \quad (7.132)$$

and by defining an “augmented” vector

$$\mathbf{x}_t^A \triangleq \text{col}(\mathbf{x}_t, \mathbf{z}_t). \quad (7.133)$$

The control functions take on the form

$$\mathbf{u}_t = \boldsymbol{\mu}_t(\mathbf{x}_t^A), \quad t = 0, 1, \dots, T-1. \quad (7.134)$$

It follows that the optimal control problem with the a priori unknown vectors \mathbf{x}_0 , $\boldsymbol{\pi}$, and \mathbf{x}^* can be stated as a Problem C2' (we point out once again the “flexibility” deriving from having introduced this problem). Then, if ADP is used, Control Functions (7.134) can be optimized as usual. Alternatively, if the ERIM is applied, then Control Functions (7.134) are constrained to have the structure of FSP functions, that is,

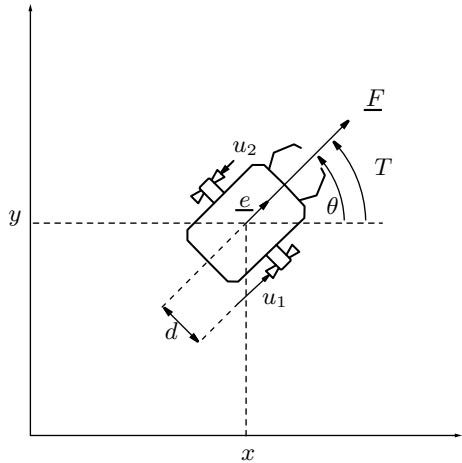
$$\mathbf{u}_t = \tilde{\boldsymbol{\mu}}(\mathbf{x}_t^A, \mathbf{w}_{tn_t}), \quad t = 0, 1, \dots, T-1. \quad (7.135)$$

Then, by eliminating the state and control vectors in Cost (7.131), we obtain the cost function

$$\hat{J}(\mathbf{w}_n) = \underset{\boldsymbol{\xi}, \mathbf{x}_0, \boldsymbol{\pi}, \mathbf{x}^*}{\mathbb{E}} J(\mathbf{w}_n, \boldsymbol{\xi}, \mathbf{x}_0, \boldsymbol{\pi}, \mathbf{x}^*), \quad (7.136)$$

which has to be minimized with respect to the vector \mathbf{w}_n given by (7.73). Then, if we choose to use a stochastic gradient algorithm similar to (7.92), we have

$$\begin{aligned} \mathbf{w}_n(k+1) &= \mathbf{w}_n(k) - \alpha_k \nabla_{\mathbf{w}_n} J[\mathbf{w}_n(k), \boldsymbol{\xi}(k), \mathbf{x}_0(k), \boldsymbol{\pi}(k), \mathbf{x}^*(k)], \\ k &= 0, 1, \dots \end{aligned}$$

Fig. 7.14 The space robot

In the following example, a non-LQ optimal control problem, for which it is impossible to derive analytical solutions, is dealt with.

Example 7.4 Consider the space robot shown in Fig. 7.14, which is assumed to move in the plane of the coordinate system of the figure. The robot's position is described by the Cartesian coordinates x, y and by the angle ϑ of its axis of symmetry (oriented in the direction of the vector e of unit length) with respect to the axis x . Two couples of thrusters, aligned with this axis, are mounted on the robot sides. Their thrusts, u_1 and u_2 , can be controlled so as to obtain the desired intensity of the force F and the desired torque C . We assume the mass m and the moment of inertia J to remain constant during the manoeuvre described below. Then, we write

$$\mathbf{F} = (u_1 + u_2) \mathbf{e} = m \frac{d\mathbf{v}}{dt} \quad (7.137)$$

$$C = (u_1 - u_2) d = J \frac{d\omega}{dt}, \quad (7.138)$$

where d is the distance between the thrusters and the axis of symmetry, \mathbf{v} is the robot velocity, and ω is the robot angular velocity (this robot was considered in the examples reported in [62] and in other related works). Let $x_1 = x, x_2 = \dot{x}, x_3 = y, x_4 = \dot{y}, x_5 = \vartheta, x_6 = \dot{\vartheta}$, and

$$\mathbf{x} \triangleq \text{col}(x_i : i = 1, \dots, 6).$$

Then, from (7.137) and (7.138), we derive the nonlinear differential dynamic system

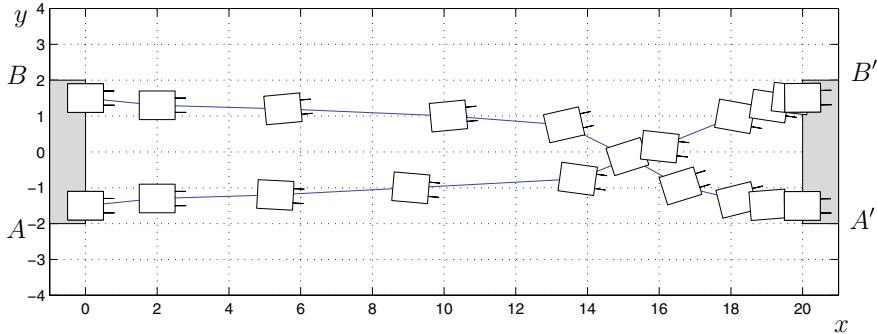


Fig. 7.15 Two optimal trajectories of the space robot from the interval AB to the interval $A'B'$

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{1}{m} (u_1 + u_2) \cos x_5 \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= \frac{1}{m} (u_1 + u_2) \sin x_5 \\ \dot{x}_5 &= x_6 \\ \dot{x}_6 &= \frac{d}{J} (u_1 - u_2)\end{aligned}\tag{7.139}$$

with the constraints

$$|u_1| \leq U, |u_2| \leq U,\tag{7.140}$$

where U is the maximum thrust value allowed.

The space robot is required to start from *any* point (specified by the variable $x_3(0)$) of a segment AB (see Fig. 7.15: AB might be the edge of a space platform) and to reach, at a fixed instant t_f , *any* given point (specified by $x_3(t_f)$) belonging to the segment $A'B'$ (the edge of another platform). $T = 10$ control stages are allowed. When the robot reaches the segment $A'B'$, it must stop with the angle $\vartheta = 0$. Then, the initial and final state sets are given by

$$X_0 = \{\mathbf{x} \in \mathbb{R}^6 : x_1 = 0, x_2 = 0, -2 \leq x_3 \leq 2, x_4 = 0, x_5 = 0, x_6 = 0\}$$

and

$$X_T = \{\mathbf{x} \in \mathbb{R}^6 : x_1 = 20, x_2 = 0, -2 \leq x_3 \leq 2, x_4 = 0, x_5 = 0, x_6 = 0\},$$

respectively. We assume $x_3(0)$ and $x_3(t_f)$ to be mutually independent random variables uniformly distributed on the two intervals. The manoeuvring problem can be viewed as a Problem C2' and, more specifically, as an example of Case 2 addressed in this section.

Since the fuel consumption is a part of the cost J , Function (7.130) can be rewritten as follows:

$$J = \sum_{t=0}^{T-1} \left[c(u_{1t}) + c(u_{2t}) + |\mathbf{x}^* - \mathbf{x}_{t+1}|_{V_{t+1}}^2 \right],$$

where $\mathbf{x}_t \triangleq \mathbf{x}(t\Delta t)$, $t = 0, 1, \dots, T$, $\mathbf{u}_t \triangleq \mathbf{u}(t\Delta t)$, $t = 0, 1, \dots, T-1$, and $\Delta t = t_f/T$. For brevity, we do not write the discretized version of the differential dynamic system (7.139), as it is simply given by a first-order Euler approximation of the state equation. Moreover, $V_t \triangleq \text{diag}[1, 0.1, 40, 0.1, 40, 0.1]$, $t = 1, \dots, T-1$, and $V_T \triangleq \text{diag}[40, 40, 40, 40, 40, 40]$. The cost of the fuel consumption is taken into account by the transition costs

$$c(u_{it}) = K \left[\frac{1}{a} \ln(2 + e^{au_{it}} + e^{-au_{it}}) - \frac{1}{a} \ln(4) \right], \quad i = 1, 2, \quad (7.141)$$

which (for large enough values of the parameter a) approximate the non-differentiable costs $K|u_{it}|$ (see Fig. 7.16). Such costs are realistic because it is reasonable to assume the fuel consumption to be proportional to the thrust. We chose $a = 50$, $K = 0.01$. The matrices V_t and the constant K are chosen so as to obtain a reasonable compromise between the “attractiveness” of the final states \mathbf{x}^* to be reached and the fuel consumption. Two hidden-layer networks are used as FSP functions. The first hidden

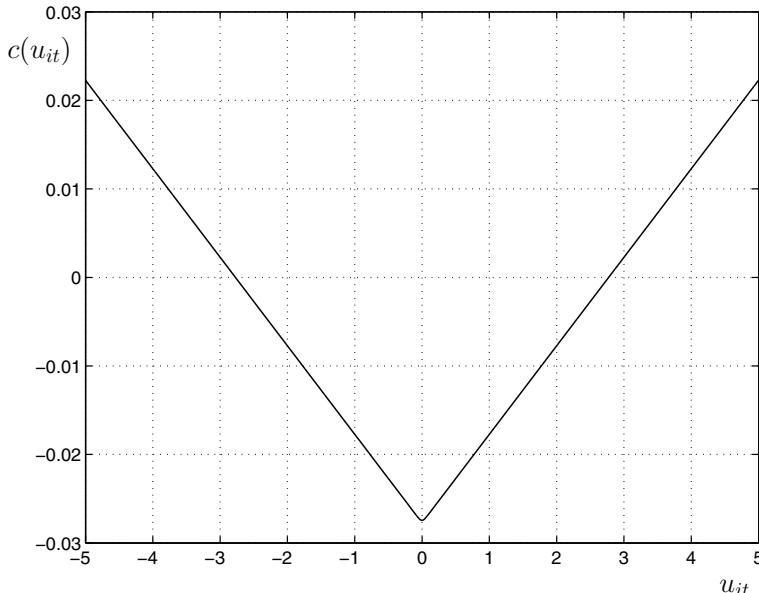


Fig. 7.16 Approximate Cost Function (7.141) with $a = 50$, $K = 0.01$

layer $s = 1$ has 80 sigmoidal activation functions $h(z) = \tanh z$ (see Fig. 7.3). The second hidden layer $s = L = 2$ has two sigmoidal activation functions $h(z) = \tanh z$, multiplied by $U = 1$. Since these functions generate the control variables u_{1t} and u_{2t} , Constraints (7.140) cannot be violated and can be removed.

The FSP functions implementing Control Functions (7.135) take on the form

$$\boldsymbol{u}_t = \tilde{\boldsymbol{\mu}}(\boldsymbol{x}_t, \boldsymbol{x}^*, \boldsymbol{w}_n), \quad t = 0, 1, \dots, T - 1, \quad (7.142)$$

where $\dim(\text{col}(\boldsymbol{x}_t, \boldsymbol{x}^*)) = 12$. In applying the stochastic gradient algorithm, we took $c_1 = 0.0001$, $c_2 = 10000$ (see (5.85)), and $\beta = 0.9$. Two optimal trajectories are shown in Fig. 7.15. \triangleleft

7.8.2 The Random Variables $\xi_0, \xi_1, \dots, \xi_{T-1}$ are Measurable

In this section, we address two situations: (i) $\xi_0, \xi_1, \dots, \xi_{T-1}$ are mutually independent; and (ii) such variables are generated by a dynamic system driven by a white random sequence.

1. $\xi_0, \xi_1, \dots, \xi_{T-1}$ are mutually independent

The possibility of measuring the random vector ξ_t at each stage t enables the DM to generate the controls not only on the basis of the feedback information given by the state \boldsymbol{x}_t but also of the “feedforward information” obtained from the measures taken on ξ_t . Of course, the utilization of additional information is in general beneficial for the decrement of the expected value of the process cost.

ADP and the ERIM can be easily applied to optimize the control functions, which take on the form

$$\boldsymbol{u}_t = \boldsymbol{\mu}_t(\boldsymbol{x}_t, \boldsymbol{\xi}_t), \quad t = 0, 1, \dots, T - 1. \quad (7.143)$$

The optimal cost-to-go functions are defined as (see (7.8))

$$\begin{aligned} \hat{J}_t^\circ(\boldsymbol{x}_t, \boldsymbol{\xi}_t) &\triangleq \min_{\boldsymbol{\mu}_t, \dots, \boldsymbol{\mu}_{T-1}, \boldsymbol{\xi}_{t+1}, \dots, \boldsymbol{\xi}_{T-1}} \mathbb{E} \left\{ \sum_{k=t}^{T-1} h_k[\boldsymbol{x}_k, \boldsymbol{\mu}_k(\boldsymbol{x}_k, \boldsymbol{\xi}_k), \boldsymbol{\xi}_k] + h_T(\boldsymbol{x}_T) \right\}, \\ \boldsymbol{x}_t &\in \bar{X}_t, \quad t = 0, 1, \dots, T - 1. \end{aligned} \quad (7.144)$$

Then, the DP recursive equation can be easily derived and is given by (compare with (7.13))

$$\hat{J}_T^\circ(\boldsymbol{x}_T, \boldsymbol{\xi}_T) = h_T(\boldsymbol{x}_T), \quad (7.145a)$$

$$\hat{J}_t^\circ(\mathbf{x}_t, \boldsymbol{\xi}_t) = \min_{\mathbf{u}_t \in U_t(\mathbf{x}_t)} \left[h_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t) + \underset{\boldsymbol{\xi}_{t+1}}{\text{E}} \hat{J}_{t+1}^\circ(\mathbf{x}_{t+1}, \boldsymbol{\xi}_{t+1}) \right], \\ \mathbf{x}_t \in \bar{X}_t, \quad t = T - 1, T - 2, \dots, 0. \quad (7.145b)$$

Of course, for a given value of \mathbf{x}_0 , the expected process cost $\hat{J}_0^\circ(\mathbf{x}_0)$ without measurements on $\boldsymbol{\xi}_t$ (which is given by (7.13b) for $t = 0$) is not lower than the expected process cost with measurement on $\boldsymbol{\xi}_t$, that is,

$$\underset{\boldsymbol{\xi}_0}{\text{E}} \hat{J}_0^\circ(\mathbf{x}_0, \boldsymbol{\xi}_0) \leq \hat{J}_0^\circ(\mathbf{x}_0),$$

where $\hat{J}_0^\circ(\mathbf{x}_0)$ refers to the situation in which the random variables $\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_{T-1}$ are not measurable.

In economics, a difference between costs like

$$\hat{J}_0^\circ(\mathbf{x}_0) - \underset{\boldsymbol{\xi}_0}{\text{E}} \hat{J}_0^\circ(\mathbf{x}_0, \boldsymbol{\xi}_0) \geq 0$$

has been called the *expected value of perfect information* (EVPI); see, for instance, the classic work [65].

The ERIM can be applied as usual by constraining the control functions (7.143) to take on the structures of FSP functions, i.e.,

$$\mathbf{u}_t = \tilde{\mu}(\mathbf{x}_t, \boldsymbol{\xi}_t, \mathbf{w}_{tn}), \quad t = 0, 1, \dots, T - 1. \quad (7.146)$$

The control and the state vectors in the cost functional can be eliminated by using Functions (7.146) and State Equation (7.1), thus obtaining the cost function

$$\hat{J}(\mathbf{w}_n) = \underset{\mathbf{x}_0, \boldsymbol{\xi}}{\text{E}} J(\mathbf{w}_n, \mathbf{x}_0, \boldsymbol{\xi}), \quad (7.147)$$

where \mathbf{w}_n is given by (7.73). Then, an optimal weight vector \mathbf{w}_n° can be computed by solving the usual NLP problem. Note that Cost (7.147) depends on the same variables as Cost (7.75). Of course, the way the two costs depend on them is different since Functions (7.146) and (7.71) differ.

If, at the stage t , the DM is allowed to observe, besides $\boldsymbol{\xi}_t$, some future random vectors $\boldsymbol{\xi}_{t+1}, \dots, \boldsymbol{\xi}_{t+k}$, where k is a given integer, then ADP and the ERIM can be applied in a similar way.

2. $\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_{T-1}$ are generated by a dynamic system

A particularly simple case in which the vectors $\boldsymbol{\xi}_t$ are not mutually independent occurs when they constitute a Markov process generated by a dynamic system driven by white noise. This dynamic system is given by

$$\boldsymbol{\xi}_{t+1} = \psi_t(\boldsymbol{\xi}_t, \boldsymbol{\varepsilon}_t), \quad t = 0, 1, \dots, T - 1, \quad (7.148)$$

where $\mathbf{x}_0, \xi_0, \varepsilon_0, \varepsilon_1, \dots, \varepsilon_{T-1}$ are mutually independent random vectors with known probability densities $p_0(\mathbf{x}_0)$, $p_0(\xi_0)$, and $p_t(\varepsilon_t)$, $t = 0, 1, \dots, T-1$. More generally, the dependence of ξ_t on preceding random vectors $\xi_{t-1}, \dots, \xi_{t-k}$ can be expressed as follows:

$$\xi_{t+1} = \Psi_t(\xi_t, \dots, \xi_{t-k}, \varepsilon_t), \quad t = 0, 1, \dots, T-1. \quad (7.149)$$

Obviously, a suitable introduction of state variables enables one to represent Eq. (7.149) as a dynamic system of the form (7.148).

Let us address random vectors generated by (7.148) and note that the possibility of measuring ξ_t (see Fig. 7.17) makes the problem at hand a problem of type C2'. Indeed, it is sufficient to consider an “augmented” dynamic system with state $\mathbf{x}_t^A \triangleq \text{col}(\mathbf{x}_t, \xi_t)$ and with state equation given by (7.1) and (7.148). Then, ADP can be applied to optimize the control functions

$$\mathbf{u}_t = \mu_t(\mathbf{x}_t, \xi_t), \quad t = 0, 1, \dots, T-1. \quad (7.150)$$

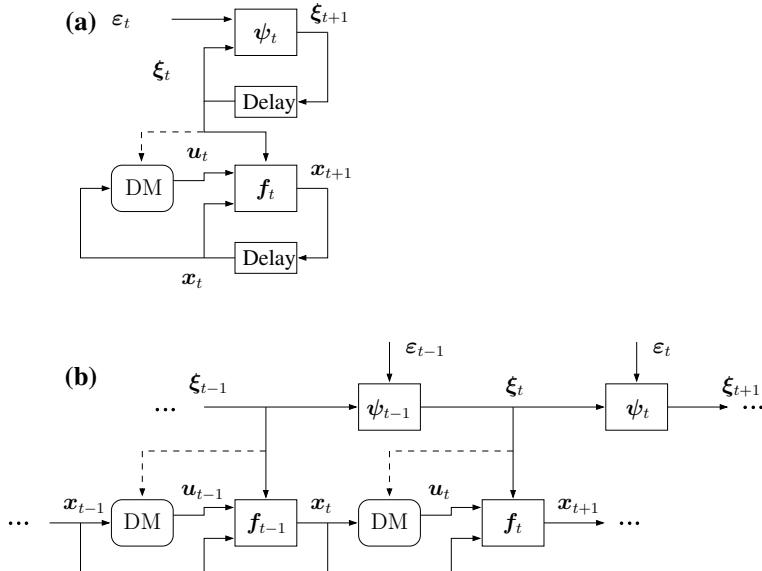


Fig. 7.17 The random vector ξ_t is generated by a dynamic system: (a) control scheme, (b) “unfolded” control scheme. In general, the “unfolded” schemes are useful for solving optimal control problems by the ERIM whenever the computation of the gradient is required (see Sects. 7.6.2, 7.6.3, and Fig. 7.2). The dashed lines are present if the DM can measure ξ_t . Then, we are in Case 2 of Sect. 7.8.2. Otherwise, we are in a situation addressed in Problem C3 (see Chap. 8) where the state \mathbf{x}_t^A cannot be observed completely

Relationships similar to (7.144) and (7.145) in which the averaging operations have to be performed with respect to the primitive random variables $\mathbf{x}_0, \xi_0, \varepsilon_0, \varepsilon_1, \dots, \varepsilon_{T-1}$ can be derived.

If the ERIM is used, Control Functions (7.150) are replaced by the FSP functions

$$\mathbf{u}_t = \tilde{\mu}(\mathbf{x}_t, \xi_t, \mathbf{w}_{tn_t}), \quad t = 0, 1, \dots, T-1. \quad (7.151)$$

Then, once again we eliminate the control and the state vectors (i.e., both \mathbf{u}_t and \mathbf{x}_t^A) in the cost functional by using Functions (7.151) and State Equations (7.1) and (7.148). Therefore, we reduce a Problem C2' to a Problem C2'_n. The cost function to be minimized is given by

$$\hat{J}(\mathbf{w}_n) = \underset{\mathbf{x}_0, \xi_0, \varepsilon}{\text{E}} J(\mathbf{w}_{tn_t}, \mathbf{x}_0, \xi_0, \varepsilon), \quad (7.152)$$

where $\varepsilon \triangleq \text{col}(\varepsilon_0, \varepsilon_1, \dots, \varepsilon_{T-1})$.

7.8.3 Tracking a Stochastic Reference

Let us require that the system state \mathbf{x}_t has to track a measurable random vector $\mathbf{x}_t^* \in \mathbb{R}^d$ given by the state of a dynamic system driven by white noise, i.e.,

$$\mathbf{x}_{t+1}^* = \varphi_t(\mathbf{x}_t^*, \varepsilon_t), \quad t = 0, 1, \dots, T-1. \quad (7.153)$$

$\mathbf{x}_0, \varepsilon_0, \varepsilon_1, \dots, \varepsilon_{T-1}$ are mutually independent random vectors with known probability density functions (compare (7.153) with State Equation (7.148)). Several cases of reference tracking problems in the LQ framework are reported, for instance, in [52].

To take the tracking requirement into account, we may add penalty terms to the cost (7.3). If, for example, we choose quadratic terms (this is not necessarily required as, in general, we need not to be in an LQ context), the cost becomes

$$J = \sum_{t=0}^{T-1} h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t) + h_T(\mathbf{x}_T) + \sum_{t=1}^T |\mathbf{x}_t^* - \mathbf{x}_t|_{V_t}^2, \quad (7.154)$$

where the matrices $V_t = V_t^\top \geq 0$ have the same meaning as in (7.130). The control scheme related to the reference tracking problem is given in Fig. 7.18. Clearly, the tracking problem dealt with in this section generalizes Case 2 of Sect. 7.8.1, where the target state \mathbf{x}^* was random but constant in time.

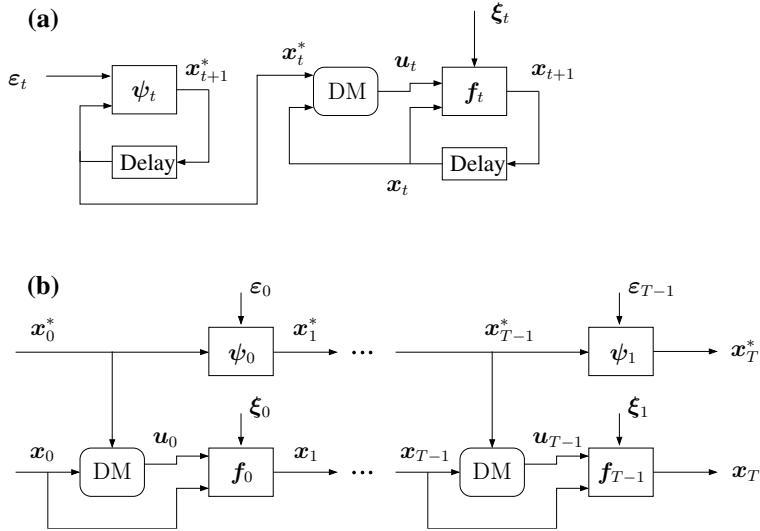


Fig. 7.18 The random reference vector \mathbf{x}_t^* is generated by a dynamic system: (a) control scheme, (b) “unfolded” control scheme

If ADP is applied, then an “augmented” dynamic system made of (7.1) and (7.153) has to be defined with a perfectly measurable state $\mathbf{x}_t^A \triangleq \text{col}(\mathbf{x}_t, \mathbf{x}_t^*)$. Then, the control law is given by

$$\mathbf{u}_t = \mu_t(\mathbf{x}_t, \mathbf{x}_t^*), \quad t = 0, 1, \dots, T-1. \quad (7.155)$$

After defining the optimal cost-to-go functions $\hat{J}_t^\circ(\mathbf{x}_t, \mathbf{x}_t^*)$, the control functions (7.155) are optimized via the recursive equation

$$\begin{aligned} \hat{J}_T^\circ(\mathbf{x}_T, \mathbf{x}_T^*) &= h_T(\mathbf{x}_T) + |\mathbf{x}_T^* - \mathbf{x}_T|_{V_T}^2, \\ \hat{J}_t^\circ(\mathbf{x}_t, \mathbf{x}_t^*) &= \min_{\mathbf{u}_t \in U_t(\mathbf{x}_t)} \mathbb{E}_{\xi_t, \varepsilon_t} [h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t) + |\mathbf{x}_{t+1}^* - \mathbf{x}_{t+1}|_{V_{t+1}}^2 \\ &\quad + \hat{J}_{t+1}^\circ(\mathbf{x}_{t+1}, \mathbf{x}_{t+1}^*)], \\ t &= T-1, T-2, \dots, 0. \end{aligned}$$

If the ERIM is applied, then Control Functions (7.155) take on the form of FSP functions, that is,

$$\mathbf{u}_t = \tilde{\mu}(\mathbf{x}_t, \mathbf{x}_t^*, \mathbf{w}_{tn_t}), \quad t = 0, 1, \dots, T-1. \quad (7.156)$$

The elimination of all the variables except the primitive random ones in Cost (7.154) yields the cost (compare with (7.147))

$$\hat{J}(\mathbf{w}_n) = \underset{\mathbf{x}_0, \mathbf{x}_0^*, \boldsymbol{\xi}, \boldsymbol{\varepsilon}}{\mathbb{E}} J(\mathbf{w}_n, \mathbf{x}_0, \mathbf{x}_0^*, \boldsymbol{\xi}, \boldsymbol{\varepsilon}) \quad (7.157)$$

to be minimized with respect to \mathbf{w}_n , where $\boldsymbol{\varepsilon} \triangleq \text{col}(\varepsilon_0, \varepsilon_1, \dots, \varepsilon_{T-1})$.

References

1. Archibald TW, McKinnon KIM, Thomas LC (1997) An aggregate stochastic dynamic programming model of multireservoir systems. *Water Resour Res* 33:333–340
2. Atkeson C, Stephens B (2008) Random sampling of states in dynamic programming. *IEEE Trans Syst Man Cybern Part B Cybern* 38:924–929
3. Baglietto M, Cervellera C, Sanguineti M, Zoppoli R (2010) Management of water resources systems in the presence of uncertainties by nonlinear approximators and deterministic sampling. *Comput Optim Appl* 47:349–376
4. Bellman R (1957) Dynamic programming. Princeton University Press
5. Bellman R, Dreyfus S (1959) Functional approximations and dynamic programming. *Math Tables Aids Comput* 13:247–251
6. Bellman R, Dreyfus SE (1962) Applied dynamic programming. Princeton University Press
7. Bellman R, Kalaba R, Kotkin B (1963) Polynomial approximation-a new computational technique in dynamic programming. *Math Comput* 17:155–161
8. Benveniste LM, Scheinkman JA (1979) On the differentiability of the value function in dynamic models of economics. *Econometrica* 47:727–732
9. Bertsekas DP (1999) Nonlinear programming, 2nd edn. Athena Scientific
10. Bertsekas DP (2000) Dynamic programming and optimal control, vol 2. Athena Scientific
11. Bertsekas DP (2005) Dynamic programming and optimal control, vol 1. Athena Scientific
12. Bertsekas DP (2005) Dynamic programming and suboptimal control: a survey from ADP to MPC. *Eur J Control* 11:310–334
13. Bertsekas DP, Borkar VS, Nedic A (2004) A review of design and modeling in computer experiments. In: Si J, Barto AG, Powell WB, Wunsch D (eds) *Handbook of learning and approximate dynamic programming*. IEEE Press, pp 233–257
14. Bertsekas DP, Shreve SE (1978) Stochastic optimal control-the discrete time case. Academic Press
15. Bertsekas DP, Tsitsiklis JN (1996) Neuro-dynamic programming. Athena Scientific
16. Buoniu L, Ernst D, De Schutter B, Babuk R (2010) Approximate dynamic programming with a fuzzy parameterization. *Automatica* 46:804–814
17. Busoniu L, Babuska R, De Schutter B, Ernst D (2010) Reinforcement learning and dynamic programming using function approximators. CRC Press
18. Buttou L (2012) Stochastic gradient descent tricks. In: Montavon G, Orr G, Müller K-R, (eds) *Neural networks: tricks of the trade*. Lecture notes in computer science, vol 7700. Springer, pp 421–436
19. Cervellera C, Chen VCP, Wen A (2006) Optimization of a large-scale water reservoir network by stochastic dynamic programming with efficient state space discretization. *Eur J Oper Res* 171:1139–1151
20. Cervellera C, Gaggero M, Macciò D (2012) Efficient kernel models for learning and approximate minimization problems. *Neurocomputing* 97:74–85
21. Cervellera C, Gaggero M, Macciò D (2014) Low-discrepancy sampling for approximate dynamic programming with local approximators. *Comput Oper Res* 43:108–115

22. Cervellera C, Gaggero M, Macciò D, Marcialis R (2013) Quasi-random sampling for approximate dynamic programming. In: Proceedings of the international joint conference on neural networks, pp 2567–2574
23. Cervellera C, Gaggero M, Macciò D, Marcialis R (2015) Efficient use of Nadaraya-Watson models and low-discrepancy sequences for approximate dynamic programming. In: Proceedings of the international joint conference on neural networks
24. Cervellera C, Macciò D (2011) A comparison of global and semi-local approximation in T -stage stochastic optimization. *Eur J Oper Res* 208:109–118
25. Cervellera C, Muselli M (2007) Efficient sampling in approximate dynamic programming. *Comput Optim Appl* 38:417–443
26. Cervellera C, Wen A, Chen VCP (2007) Neural network and regression spline value function approximations for stochastic dynamic programming. *Comput Oper Res* 34:70–90
27. Chaudhary SK (2005) American options and the LSM algorithm: quasi-random sequences and Brownian bridges. *J Comput Financ* 8:101–115
28. Chen VCP, Ruppert D, Shoemaker CA (1999) Applying experimental design and regression splines to high-dimensional continuous-state stochastic dynamic programming. *Oper Res* 47:38–53
29. Chen VCP, Tsui KL, Barton RR, Allen JK (2003) A review of design and modeling in computer experiments. In: Khattree R, Rao CR (eds) *Handbook in statistics: statistics in industry*, vol 22. Elsevier, pp 231–261
30. Dion M, Lecuyer P (2010) American option pricing with randomized quasi-Monte Carlo simulations. In: Johansson B, Jain S, Montoya-Torres J, Hugan J, Yücesan E (eds) *Proceedings of the 2010 winter simulation conference*, pp 2705–2720
31. Ernst D, Geurts P, Wehenkel L (2005) Tree-based batch mode reinforcement learning. *J Mach Learn Res* 6:503–556
32. Farahmand AM, Ghavamzadeh M, Szepesvári C, Mannor S (2009) Regularized fitted Q-iteration for planning in continuous-space Markovian decision problems. In: *Proceedings of the American control conference*, pp 725–730
33. Farahmand AM, Munos R, Szepesvári C (2010) Error propagation for approximate policy and value iteration. In: Lafferty J, Williams CKI, Shawe-Taylor J, Zemel RS, Culotta A (eds) *Advances in neural information processing systems 23*. MIT Press, pp 568–576
34. Foufoula-Georgiou E, Kitanidis PK (1988) Gradient dynamic programming for stochastic optimal control of multidimensional water resources systems. *Water Resour Res* 24:1345–1359
35. Gaggero M, Gnecco G, Sanguineti M (2013) Dynamic programming and value-function approximation in sequential decision problems: error analysis and numerical results. *J Optim Theory Appl* 156:380–416
36. Gaggero M, Gnecco G, Sanguineti M (2014) Approximate dynamic programming for stochastic N -stage optimization with application to optimal consumption under uncertainty. *Comput Optim Appl* 58:31–85
37. Gaggero M, Gnecco G, Sanguineti M (2014) Suboptimal policies for stochastic N -stage optimization problems: accuracy analysis and a case study from optimal consumption. In: El Ouardighi F, Kogan K (eds) *Models and methods in economics and management*. Springer, pp 27–50
38. Gal S (1979) Optimal management of a multireservoir water supply system. *Water Resour Res* 15:737–749
39. Gale D (1967) On optimal development in a multi-sector economy. *Rev Econ Stud* 34:1–18
40. Giuliani M, Quinn JD, Herman JD, Castelletti A, Reed PM (2018) Scalable multiobjective control for large scale water resources systems under uncertainty. *IEEE Trans Control Syst Technol* 26:1492–1499
41. Gnecco G, Sanguineti M (2010) Suboptimal solutions to dynamic optimization problems via approximations of the policy functions. *J Optim Theory Appl* 146:764–794
42. Gnecco G, Sanguineti M (2016) Neural approximations of the solutions to a class of stochastic optimal control problems. *J Neurotechnol* 1:1–16

43. Gnecco G, Sanguineti M (2018) Neural approximations in discounted infinite-horizon stochastic optimal control problems. *Eng Appl Artif Intell* 74:294–302
44. Guo W, Si J, Liu F, Mei S (2018) Policy approximation in policy iteration approximate dynamic programming for discrete-time nonlinear systems. *IEEE Trans Neural Netw Learn Syst* 29:2794–2807
45. Haykin S (2008) Neural networks and learning systems. Pearson Prentice-Hall
46. Johnson SA, Stedinger JR, Shoemaker C, Li Y, Tejada-Guibert JA (1993) Numerical solution of continuous-state dynamic programs using linear and spline interpolation. *Oper Res* 41:484–500
47. Judd K (1998) Numerical methods in economics. MIT Press
48. Kamalapurkar R, Walters P, Rosenfeld J, Dixon W (2018) Reinforcement learning for optimal feedback control. Springer
49. Kitanidis PK (1986) Hermite interpolation on an n -dimensional rectangular grid. Technical report, St. Anthony Falls Hydraulics Laboratory, University of Minnesota
50. Kiumarsi B, Vamvoudakis KG, Modares H, Lewis FL (2018) Optimal and autonomous control using reinforcement learning: a survey. *IEEE Trans Neural Netw Learn Syst* 29:2042–2062
51. Kleinman DL, Athans M (1968) The design of suboptimal linear time-varying systems. *IEEE Trans Autom Control* 13:150–159
52. Kwakernaak H, Sivan R (1972) Linear optimal control systems. Wiley
53. Larson RE (1968) State increment dynamic programming. Elsevier
54. Lewis FL, Vrabie D, Vamvoudakis KG (2012) Reinforcement learning and feedback control: using natural decision methods to design optimal adaptive controllers. *IEEE Control Syst Mag* 32:76–105
55. Montrucchio L (1987) Lipschitz continuous policy functions for strongly concave optimization problems. *J Math Econ* 16:259–273
56. Montrucchio L (1998) Thompson metric, contraction property and differentiability of policy functions. *J Econ Behav Organ* 33:449–466
57. Montrucchio L, Boldrin M (1986) On the indeterminacy of capital accumulation paths. *J Econ Behav Organ* 40:26–36
58. Munos R, Moore A (2002) Variable resolution discretization in optimal control. *Mach Learn* 49:291–323
59. Munos R, Szepesvári C (2008) Finite-time bounds for fitted value iteration. *J Mach Learn Res* 1:815–857
60. Nguyen DH, Widrow B (1990) Neural networks for self-learning control systems. *IEEE Control Syst Mag* 10:18–23
- 61.Ormoneit D, Sen S (2002) Kernel-based reinforcement learning. *Mach Learn* 49:161–178
62. Parisini T, Zoppoli R (1994) Neural networks for feedback feedforward nonlinear control systems. *IEEE Trans Neural Netw* 5:436–449
63. Powell WB (2011) Approximate dynamic programming: solving the curse of dimensionality, 2nd edn. Wiley
64. Philbrick CR Jr, Kitanidis PK (2001) Improved dynamic programming methods for optimal control of lumped-parameter stochastic systems. *Oper Res* 49:398–412
65. Raiffa H, Schlaifer R (1961) Applied statistical decision theory. Harvard University Press
66. Riedmiller M (2005) Neural fitted Q iteration. First experiences with a data efficient neural reinforcement learning method. In: Proceedings of the 16th European conference on machine learning, pp 317–328
67. Salas JD, Tabios GQ III, Bartolini P (1985) Approaches to multivariate modeling of water resources time series. *Water Resour Bull* 21:683–708
68. Samuel AL (1959) Some studies in machine learning using the game of checkers. *IBM J Res Dev* 3:211–229
69. Schweitzer PJ, Seidmann A (1985) Generalized polynomial approximations in Markovian decision processes. *J Math Anal Appl* 110:568–582
70. Shapiro A, Dentcheva D, Ruszczynski A (2009) Lectures on stochastic programming: modeling and theory. MOS-SIAM series on optimization

71. Si J, Barto AG, Powell WB, Wunsch D (eds) (2004) Handbook of learning and approximate dynamic programming. IEEE Press
72. Smith JE, McCardle KF (2002) Structural properties of stochastic dynamic programs. *Oper Res* 50:796–809
73. Sokolov Y, Kozma R, Werbos LD, Werbos PJ (2015) Complete stability analysis of a heuristic approximate dynamic programming control design. *Automatica* 59:9–18
74. Stokey NL, Lucas RE, Prescott EC (1989) Recursive methods in economic dynamics. Harvard University Press
75. Sutton RS (1988) Learning to predict by the method of temporal differences. *Mach Learn* 3:9–44
76. Sutton RS, Barto AG (1998) Reinforcement learning. MIT Press
77. Ten Hagen S, Kröse B (2003) Neural Q-learning. *Neural Comput Appl* 12:81–88
78. Teytaud O, Gelly S, Mary J (2007) Active learning in regression, with application to stochastic dynamic programming. In: Proceedings of the 4th international conference on informatics in control, automation, and robotics, intelligent control systems in optimization, pp 198–205
79. Thorndike E (1911) Animal intelligence: experimental studies. Macmillan
80. Tsai JCC, Chen VCP, Beck MB, Chen J (2004) Stochastic dynamic programming formulation for a wastewater treatment decision-making framework. *Ann Oper Res* 132:207–221
81. Tsitsiklis JN, Van Roy B (1996) Feature-based methods for large scale dynamic programming. *Mach Learn* 22:59–94
82. Turgeon A (1981) A decomposition method for the long-term scheduling of reservoirs in series. *Water Resour Res* 17:1565–1570
83. Wan EA, Benfaghi F (1996) Diagrammatic derivation of gradient algorithm for neural networks. *Neural Comput* 8:182–201
84. Wang D, He H, Liu D (2017) Adaptive critic nonlinear robust control: a survey. *IEEE Trans Cybern* 47:3429–3451
85. Watkins C (1988) Learning from delayed rewards. PhD thesis, Cambridge University, Cambridge, U.K
86. Watkins C, Dayan P (1992) Q-learning. *Mach Learn* 8:279–292
87. Werbos PJ (1989) Neural networks for control and system identification. In: Proceedings of the IEEE conference on decision and control, pp 260–265
88. Werbos PJ (1991) A menu of designs for reinforcement learning over time. In: Miller WTI, Sutton RS, Werbos PJ (eds) *Neural networks for control*. MIT Press, pp 67–95
89. Werbos PJ (1992) Approximate dynamic programming for real-time control and neural modeling. In: White DA, Sofge DA (eds) *Handbook of intelligent control*. Von Nostrand Reinhold
90. Yakowitz S (1982) Dynamic programming applications in water resources. *Water Resour Res* 18:673–696

Chapter 8

Stochastic Optimal Control with Imperfect State Information over a Finite Horizon



Whereas in Chaps. 6 and 7 we assumed that all the state components were measured perfectly by the decision-maker (DM), in this chapter we shall give up on this hypothesis, which is not always realistic. We shall consider a context where the DM has access to a vector \mathbf{y}_t of measures that only contains an imperfect information on the system state \mathbf{x}_t .

This may occur because the number of observation devices (for example, transducers in an industrial plant) is insufficient to measure all the state components and/or the devices are affected by noises. More specifically, we assume that \mathbf{y}_t is a p -dimensional vector coming out from an *observation or measurement channel* of the form

$$\mathbf{y}_t = \mathbf{g}_t(\mathbf{x}_t, \boldsymbol{\eta}_t), \quad t = 0, 1, \dots, T - 1, \quad (8.1)$$

where \mathbf{g}_t is a known function and $\boldsymbol{\eta}_t$ is a random vector.

As can be easily understood, the unavailability of perfect information on the system state makes the resulting stochastic optimal control problems much more difficult than problems in which there are perfect measures on \mathbf{x}_t , as in Problem C2. Moreover, it is well known that, in the classical design of closed-loop control systems, disturbances are to be considered less dangerous if they act on the controlled plant than if they act on the measurement channel.

The DM has then to retain in its memory the information collected up to the stage t in order to compute the optimal decision \mathbf{u}_t° by minimizing the cost functional. We assume that the DM's memory is “perfect.” The information is gathered in a vector that is termed the *information state vector* or, simply, the *information vector*. Then, this vector is made of all the measures acquired up to the stage t and all the controls generated up to the stage $t - 1$, i.e.,

$$\begin{aligned} \mathbf{I}_0 &\triangleq \mathbf{y}_0, \\ \mathbf{I}_t &\triangleq \text{col}(\mathbf{y}_0^t, \mathbf{u}_0^{t-1}), \quad t = 1, \dots, T-1, \end{aligned} \tag{8.2}$$

where we used the notation $\mathbf{y}_0^t \triangleq \text{col}(\mathbf{y}_0, \mathbf{y}_1, \dots, \mathbf{y}_t)$ and $\mathbf{u}_0^{t-1} \triangleq \text{col}(\mathbf{u}_0, \mathbf{u}_1, \dots, \mathbf{u}_{t-1})$. In the context of Problem C2, one gets $\mathbf{I}_t = \mathbf{x}_t$. In general, the state information vector summarizes the whole “history of the decision process.”

After stating the stochastic optimal control Problem C3 in Sect. 8.1, in Sect. 8.2 we shall show that the transition from \mathbf{I}_t to \mathbf{I}_{t+1} is described by a relationship that plays the role of a state equation for the information state. Of course, this new equation has to be distinguished from State Equation (7.1). Since \mathbf{I}_t is perfectly known, dynamic programming (DP) can be applied for the solution of Problem C3 in the same way as for Problem C2.

As is well known, if the LQG hypotheses are verified, then the optimal control functions can be computed in closed form by DP and are given by the product of time-varying gain matrices and the state estimates provided by the Kalman filter. However, all the optimal control problems addressed in this book are of a general type. This makes the application of DP an almost impossible task. The difficulties are essentially two: (i) the increase of the dimension of \mathbf{I}_t with time and (ii) the computation of the conditional probabilities $p_t(\mathbf{x}_t | \mathbf{I}_t)$ that are needed in the backward phase of DP. Unlike what we did for Problems C1 and C2, we do not continue the discussion on DP as a tool for solving Problem C3 but we focus on the extended Ritz method (ERIM). Indeed, the ERIM escapes the second difficulty. We shall show this in Sect. 8.3 where, as usual, the admissible control functions are constrained to take on the structure of fixed-structure parametrized (FSP) functions. The infinite-dimensional optimization (IDO) Problem C3 is therefore reduced to a nonlinear programming (NLP) Problem $C3_n$, which can be solved by a descent method or by stochastic approximation. As in the case of Problem $C2_n$, the gradient of the cost can be determined by solving a classical *adjoint equation of T-stage optimal control* with the addition of a term taking the introduction of the FSP functions into account. This term is computed for the case of multi-hidden-layer (MHL) networks.

Quite a natural approximation consists in truncating \mathbf{I}_t and in only retaining the “most recent past” in memory thus “forgetting,” at the stage t , the measures acquired and the controls generated before the stage $t - M + 1$, where M is a properly chosen integer. Then, as explained in Sect. 8.4, a “limited-memory information vector” \mathbf{I}_t^M is kept in memory. We propose to mitigate the abrupt truncation of \mathbf{I}_t by making use of a fixed-dimension vector \mathbf{s}_t (called “memory vector”) by means of which the DM recalls approximately the lost data. A suitable dynamic system is introduced to generate \mathbf{s}_t . The vectors \mathbf{I}_t^M and \mathbf{s}_t become the arguments of the control functions. Other approximate control schemes are objects of research in control theory. Some of them will be mentioned at the beginning of Sect. 8.5 but their description goes far beyond our purposes. We shall present therein an approximate solution of Problem C3 that is based not just on the use of the limited memory, but also on the application

of the certainty equivalence principle. As regards the suboptimal controls, which are functions of \mathbf{I}_t^M and s_t , we do not address any theoretical issue but interesting numerical results were obtained. Two examples will be shown in Sect. 8.6.

8.1 Statement of Problem C3

As in Chap. 7, we consider a discrete-time dynamic system whose state equation is given by

$$\mathbf{x}_{t+1} = \mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t), \quad t = 0, 1, \dots, T-1. \quad (8.3)$$

Since the state is measurable through Channel (8.1), we do not know the initial state \mathbf{x}_0 . We assume, however, that the probability density $p_0(\mathbf{x}_0)$ is known. We also assume that $\mathbf{x}_0, \boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_{T-1}, \boldsymbol{\eta}_0, \boldsymbol{\eta}_1, \dots, \boldsymbol{\eta}_{T-1}$ are mutually independent random vectors with known probability densities and that these densities do not depend on other variables. As we shall see later on, these assumptions enable us to compute the conditional probability density functions $p_t(\mathbf{x}_t | \mathbf{I}_t)$, $t = 0, 1, \dots, T-1$, which are fundamental to the application of DP. In some cases, these assumptions can be weakened. For example, we can assume, as in Problem C2, that $\boldsymbol{\xi}_t$ depends on \mathbf{x}_t and \mathbf{u}_t via the conditional probability density $p_t(\boldsymbol{\xi}_t | \mathbf{x}_t, \mathbf{u}_t)$. The dependence of the density on $\boldsymbol{\eta}_t$ or other variables as well as more complex structures of the measurement channel (e.g., $\mathbf{y}_t = \mathbf{g}_t(\mathbf{x}_t, \mathbf{u}_{t-1}, \boldsymbol{\eta}_t)$) are addressed, for example, in [8].

The control vector \mathbf{u}_t may be constrained to take values from a set U_t , that is,

$$\mathbf{u}_t \in U_t \subseteq \mathbb{R}^m, \quad t = 0, 1, \dots, T-1. \quad (8.4)$$

Since we do not have a perfect information on the current state, we assumed that U_t does not depend on \mathbf{x}_t (compare (8.4) with (7.2)). Not to get involved in matters that might be complicated in the considerations of this chapter, we leave out what was discussed in Sect. 7.2 about the constraint sets X_t (see (7.7)). Then, in the following sections, the state vector \mathbf{x}_t will be considered unbounded.

The cost function is the same as in Problem C2, that is,

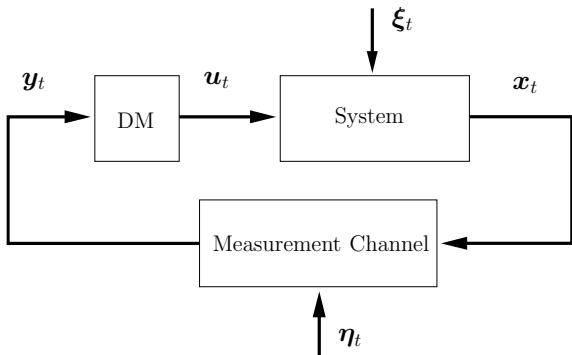
$$J \triangleq \sum_{t=0}^{T-1} h_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t) + h_T(\mathbf{x}_T). \quad (8.5)$$

As stated previously, since the DM cannot measure the state \mathbf{x}_t exactly, it has to retain the information state vector \mathbf{I}_t in its memory. It follows that the control functions take on the closed-loop form

$$\mathbf{u}_t = \boldsymbol{\mu}_t(\mathbf{I}_t), \quad t = 0, 1, \dots, T-1. \quad (8.6)$$

As usual, we define the sequence of *control functions* as a *control law*.

Fig. 8.1 Closed-loop structure of the control scheme



By analogy with Problem C2, we choose the minimization of the expected value of J as a suitable criterion to design the optimal control law. Thus, the functional to be minimized is given by

$$\hat{J} \triangleq \underset{\mathbf{x}_0, \xi_0^{T-1}, \eta_0^{T-1}}{\mathbb{E}} \left\{ \sum_{t=0}^{T-1} h_t[\mathbf{x}_t, \mu_t(\mathbf{I}_t), \xi_t] + h_T(\mathbf{x}_T) \right\}. \quad (8.7)$$

The chosen criterion leads us to state the following stochastic IDO problem, which is related to Problem PM (see Sect. 1.3.1).

Problem C3. *Find the optimal control functions $\mu_0^o(\mathbf{I}_0), \mu_1^o(\mathbf{I}_1), \dots, \mu_{T-1}^o(\mathbf{I}_{T-1})$ that minimize the functional \hat{J} subject to Constraints (8.1), (8.3), and (8.4). \diamond*

A first difficulty evident in the solution to Problem C3 comes from the dimension of the information vector \mathbf{I}_t , which grows as the number of decision stages increases. Besides the computational load, just think about the computer memory space necessary to store a “reasonable” number of samples of the vector \mathbf{I}_t . Except for the case of particularly simple problems, in general we are able to overcome such a difficulty only by approximate techniques, which lead us to suboptimal solutions. The closed-loop structure of the control scheme is depicted in Fig. 8.1.

8.2 Solution of Problem C3 by Dynamic Programming

Whereas the application of DP does not involve any significant conceptual difference between solving Problem C1 and Problem C2, an imperfect knowledge of the state vectors \mathbf{x}_t makes the use of DP much more complex, especially from the computational point of view. However, once the possibility of assigning the information state \mathbf{I}_t an identical role to the one played by the state \mathbf{x}_t in Problem C2 has been admitted, one can solve Problem C3 by applying DP in the same way as in Problem C2.

8.2.1 Reduction of Problem C3 to Problem C2

As stated previously, \mathbf{I}_t can be considered just as a state vector (*information state vector*) that evolves according to another state equation given by

$$\begin{aligned} \mathbf{I}_{t+1} &= \text{col}(\mathbf{I}_t, \mathbf{u}_t, \mathbf{y}_{t+1}), \quad t = 0, 1, \dots, T-2, \\ \mathbf{I}_0 &= \mathbf{y}_0. \end{aligned} \tag{8.8}$$

In (8.8), \mathbf{I}_t is the system state, \mathbf{u}_t is the control vector, and \mathbf{y}_{t+1} can be viewed as a random variable similar to ξ_t . Indeed, the conditional probability density of \mathbf{y}_{t+1} , given the information vector acquired up to the stage t , takes on the form $p_{t+1}(\mathbf{y}_{t+1} | \mathbf{I}_t, \mathbf{u}_t)$. This conditional density has the same form as the conditional probability density of ξ_t in Problems C2, that is, $p_t(\xi_t | \mathbf{x}_t, \mathbf{u}_t)$. Note that the cost per stage $E[h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)]$ can also be reformulated as a function of the variables of the new dynamic system (8.8). In fact, we can express the cost per stage as a function of the information vector \mathbf{I}_t and of the control vector \mathbf{u}_t , that is,

$$\bar{h}_t(\mathbf{I}_t, \mathbf{u}_t) \triangleq E_{\mathbf{x}_t, \xi_t} [h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t) | \mathbf{I}_t, \mathbf{u}_t], \quad t = 0, 1, \dots, T-1. \tag{8.9}$$

Once Problem C3 has been restated as Problem C2, the DP recursive equation can be directly written since the same steps can be followed that led us to write Eq. (7.13) for Problem C2.

Then, let us define the optimal cost-to-go functions for any possible information state \mathbf{I}_t (see (7.8)):

$$\begin{aligned} \hat{J}_{T-1}^\circ(\mathbf{I}_{T-1}) &\triangleq \min_{\mu_{T-1}} E \{ h_{T-1}[\mathbf{x}_{T-1}, \mu_{T-1}(\mathbf{I}_{T-1}), \xi_{T-1}] \\ &\quad + h_T(\mathbf{x}_T) | \mathbf{I}_{T-1} \}, \end{aligned} \tag{8.10}$$

$$\begin{aligned} \hat{J}_t^\circ(\mathbf{I}_t) &\triangleq \min_{\mu_t, \dots, \mu_{T-1}} E \left\{ \sum_{k=t}^{T-1} h_k[\mathbf{x}_k, \mu_k(\mathbf{x}_k), \xi_k] + h_T(\mathbf{x}_T) | \mathbf{I}_t \right\}, \\ t &= 0, 1, \dots, T-2. \end{aligned} \tag{8.11}$$

Even if Problem C3 has been reduced to the form of Problem C2, it is useful, for the subsequent developments, to go on writing the transition and the final costs in terms of the random variables \mathbf{x}_t and ξ_t . Then, we use the right-hand sides of (8.9). It follows that the information state \mathbf{I}_t , as a function of which we express the optimal cost-to-go functions (8.10) and (8.11), must appear as a conditioning variable in the expectations of the recursive equation of DP. Summing up, this equation takes on the form

$$\hat{J}_{T-1}^{\circ}(\mathbf{I}_{T-1}) = \min_{\mathbf{u}_{T-1} \in U_{T-1}} \mathbb{E}_{\mathbf{x}_{T-1}, \xi_{T-1}} \left[h_{T-1}(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}, \xi_{T-1}) + h_T(\mathbf{x}_T) \mid \mathbf{I}_{T-1}, \mathbf{u}_{T-1} \right], \quad (8.12a)$$

$$\hat{J}_t^{\circ}(\mathbf{I}_t) = \min_{\mathbf{u}_t \in U_t} \mathbb{E}_{\mathbf{x}_t, \xi_t, \mathbf{y}_{t+1}} \left[h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t) + \hat{J}_{t+1}^{\circ}(\mathbf{I}_{t+1}) \mid \mathbf{I}_t, \mathbf{u}_t \right], \\ t = T - 2, T - 3, \dots, 0. \quad (8.12b)$$

Recursive Equation (8.12) is formally similar to Eq. (7.13), where the system state \mathbf{x}_t is replaced by the information state \mathbf{I}_t . In order to perform the averaging operations, the conditional probability densities of the random vectors \mathbf{x}_t , ξ_t , and \mathbf{y}_{t+1} are required (the conditional density of \mathbf{y}_{t+1} is needed since $\mathbf{I}_{t+1} = \text{col}(\mathbf{I}_t, \mathbf{u}_t, \mathbf{y}_{t+1})$). These densities will be computed in the next section.

The optimal control law can be determined by the usual backward procedure of DP. $\mathbf{u}_{T-1}^{\circ} = \mu_{T-1}^{\circ}(\mathbf{I}_{T-1})$ and $\hat{J}_{T-1}^{\circ}(\mathbf{I}_{T-1})$ are obtained by minimizing the right-hand side of (8.12a) for any possible value of \mathbf{I}_{T-1} . Then, at the stage $T - 2$, $\mathbf{u}_{T-2}^{\circ} = \mu_{T-2}^{\circ}(\mathbf{I}_{T-2})$ and $\hat{J}_{T-2}^{\circ}(\mathbf{I}_{T-2})$ are computed by (8.12b). The procedure is repeated until $\mathbf{u}_0^{\circ} = \mu_0^{\circ}(\mathbf{I}_0)$ and $\hat{J}_0^{\circ}(\mathbf{I}_0)$ are determined. The minimum value of Cost (8.7) is given by

$$\hat{J}^{\circ} = \mathbb{E}_{\mathbf{y}_0} \hat{J}_0^{\circ}(\mathbf{y}_0).$$

Unless particular assumptions are verified (typically, the LQG hypotheses) or very simple versions of Problem C3 are considered, solving Eq. (8.12) exactly is practically an impossible task. This is for two main reasons: (i) the difficulty in computing the aforementioned conditional probability densities and (ii) the increasing dimension of the information state \mathbf{I}_t . In particular, the second drawback prevents the design of regular grids for \mathbf{I}_t , which already made the application of DP critical in Problems C2 and C2' for the danger of incurring the curse of dimensionality. In principle, we may repeat what we said in Sect. 6.6 about the possibility of mitigating this danger, that is, (i) approximate the optimal costs-to-go $\hat{J}_t^{\circ}(\mathbf{I}_t)$ by FSP functions and (ii) use low-discrepancy sequences to generate “non-regular” grids of discretized vectors $\mathbf{I}_t^{(l)}$. These grids should take on the structures (see (6.20))

$$\mathcal{I}'_{tL_t} \triangleq \left\{ \mathbf{I}_t^{(l)} \in \mathcal{I}_t, l = 1, \dots, L_t \right\}, \quad t = 0, 1, \dots, T - 1, \quad (8.13)$$

where \mathcal{I}_t are the sets on which the information vectors take their values. As a consequence, we should again consider the approximation and estimation errors that affect the computation of the costs $\hat{J}_t(\mathbf{I}_t)$.

However, it is clear that the increasing dimension of \mathbf{I}_t creates considerable difficulties. Indeed, we assumed that the state vector \mathbf{x}_t is unbounded, but we did not make any assumptions on the boundedness of the sets on which the random vectors

ξ_t and η_t take their values so there is no reason to believe that the vectors \mathbf{u}_t and \mathbf{y}_t , and then \mathbf{I}_t , are bounded. There would not be conceptual difficulties in introducing hypotheses that could extend what was stated at the beginning of Sect. 7.2 and defining suitable bounded sets $\bar{\mathcal{I}}_t$ (see (7.7)). Similarly, always from a conceptual point of view, we might extend the results reported in Sect. 7.3.2 to the approximate solution of DP Equation (8.12). These results refer to the computation of the expected values of the costs inside the square brackets in (8.12) by sampling the sets on which the vectors ξ_t and η_t take their values. Unfortunately, the increasing dimension of \mathbf{I}_t (and the presence of η_t) would make the discussion in Sect. 7.3.2 more difficult. Above all, the results would be of very little practical use. Therefore, we give up on approximating the “exact” DP equation (8.12) by an “approximate” DP equation as in Problems C1 and C2 (see Sects. 6.4 and 7.3, respectively). In the following sections, other types of approximation will be proposed.

8.2.2 Derivation of the Conditional Probability Densities Needed by Dynamic Programming

As we said, because of the too general hypotheses characterizing Problem C3, we gave up on finding an exact solution of DP Equation (8.12). However, it is important to remember the basic conditional probability density functions that are needed to solve (8.12). In this section, as in Problem C2, we assume that the density of ξ_t may depend on \mathbf{x}_t and \mathbf{u}_t , i.e., it has the form $p_t(\xi_t | \mathbf{x}_t, \mathbf{u}_t)$. As one can see from Recurrent Equation (8.12), three conditional densities are necessary to perform the averaging operations¹: $p_t(\xi_t | \mathbf{I}_t, \mathbf{u}_t)$, $p_t(\mathbf{x}_t | \mathbf{I}_t, \mathbf{u}_t) = p_t(\mathbf{x}_t | \mathbf{I}_t)$, and $p_{t+1}(\mathbf{y}_{t+1} | \mathbf{I}_t, \mathbf{u}_t)$.

The first density $p_t(\xi_t | \mathbf{I}_t, \mathbf{u}_t)$ can be computed by using (8.15), that is,

$$\begin{aligned} p_t(\xi_t | \mathbf{I}_t, \mathbf{u}_t) &= \int p_t(\xi_t | \mathbf{x}_t, \mathbf{I}_t, \mathbf{u}_t) p_t(\mathbf{x}_t | \mathbf{I}_t, \mathbf{u}_t) d\mathbf{x}_t \\ &= \int p_t(\xi_t | \mathbf{x}_t, \mathbf{u}_t) p_t(\mathbf{x}_t | \mathbf{I}_t) d\mathbf{x}_t. \end{aligned}$$

¹We recall three basic operations on the conditional probability density functions that are used in this section.

- The so-called chain rule:

$$p(\mathbf{a}, \mathbf{b} | \mathbf{c}) = p(\mathbf{a} | \mathbf{b}, \mathbf{c}) p(\mathbf{b} | \mathbf{c}). \quad (8.14)$$

- The integrated version of (8.14):

$$p(\mathbf{a} | \mathbf{c}) = \int p(\mathbf{a} | \mathbf{b}, \mathbf{c}) p(\mathbf{b} | \mathbf{c}) d\mathbf{b}. \quad (8.15)$$

- The Bayes formula:

$$p(\mathbf{a} | \mathbf{b}, \mathbf{c}) = \frac{p(\mathbf{b} | \mathbf{a}, \mathbf{c}) p(\mathbf{a} | \mathbf{c})}{\int (\text{numerator}) d\mathbf{a}}. \quad (8.16)$$

The density $p_t(\xi_t | \mathbf{x}_t, \mathbf{u}_t)$ is one of the data in the problem, and $p_t(\mathbf{x}_t | \mathbf{I}_t)$ is the second density we have to determine. Let us see how it is possible to compute recursively such a density for $t = 1, \dots, T - 1$, starting from $p_0(\mathbf{x}_0 | \mathbf{I}_0)$. By using (8.16), we have

$$\begin{aligned} p_t(\mathbf{x}_t | \mathbf{I}_t) &= p_t(\mathbf{x}_t | \mathbf{y}_0^t, \mathbf{u}_0^{t-1}) = p_t(\mathbf{x}_t | \mathbf{y}_t, \mathbf{y}_0^{t-1}, \mathbf{u}_0^{t-1}) \\ &= \frac{p_t(\mathbf{y}_t | \mathbf{x}_t, \mathbf{y}_0^{t-1}, \mathbf{u}_0^{t-1}) p_t(\mathbf{x}_t | \mathbf{y}_0^{t-1}, \mathbf{u}_0^{t-1})}{\int (\text{numerator}) d\mathbf{x}_t}, \quad t = 1, \dots, T - 1. \end{aligned} \quad (8.17)$$

We can write

$$p_t(\mathbf{y}_t | \mathbf{x}_t, \mathbf{y}_0^{t-1}, \mathbf{u}_0^{t-1}) = p_t(\mathbf{y}_t | \mathbf{x}_t), \quad (8.18)$$

since \mathbf{y}_t depends only on \mathbf{x}_t and η_t . $p_t(\mathbf{y}_t | \mathbf{x}_t)$ can be determined from (8.1) and $p_t(\eta_t)$. From (8.15), we also have

$$p_t(\mathbf{x}_t | \mathbf{y}_0^{t-1}, \mathbf{u}_0^{t-1}) = \int p_t(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{y}_0^{t-1}, \mathbf{u}_0^{t-1}) p_{t-1}(\mathbf{x}_{t-1} | \mathbf{y}_0^{t-1}, \mathbf{u}_0^{t-1}) d\mathbf{x}_{t-1}, \quad (8.19)$$

where

$$p_t(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{y}_0^{t-1}, \mathbf{u}_0^{t-1}) = p_t(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_{t-1})$$

and

$$p_{t-1}(\mathbf{x}_{t-1} | \mathbf{y}_0^{t-1}, \mathbf{u}_0^{t-1}) = p_{t-1}(\mathbf{x}_{t-1} | \mathbf{y}_0^{t-1}, \mathbf{u}_0^{t-2}) = p_{t-1}(\mathbf{x}_{t-1} | \mathbf{I}_{t-1}).$$

$p_t(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ can be computed from State Equation (8.3) and $p_{t-1}(\xi_{t-1} | \mathbf{x}_{t-1}, \mathbf{u}_{t-1})$. By substituting (8.18) and (8.19) into (8.17), we obtain

$$p_t(\mathbf{x}_t | \mathbf{I}_t) = \frac{p_t(\mathbf{y}_t | \mathbf{x}_t) \int p_t(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_{t-1}) p_{t-1}(\mathbf{x}_{t-1} | \mathbf{I}_{t-1}) d\mathbf{x}_{t-1}}{\int (\text{numerator}) d\mathbf{x}_t}, \quad t = 1, \dots, T - 1. \quad (8.20)$$

Recursive Equation (8.20) is started from $p_0(\mathbf{x}_0 | \mathbf{I}_0)$, which can be computed by the Bayes formula

$$p_0(\mathbf{x}_0 | \mathbf{I}_0) = p_0(\mathbf{x}_0 | \mathbf{y}_0) = \frac{p_0(\mathbf{x}_0) p_0(\mathbf{y}_0 | \mathbf{x}_0)}{\int p_0(\mathbf{x}_0) p_0(\mathbf{y}_0 | \mathbf{x}_0) d\mathbf{x}_0},$$

where $p_0(\mathbf{x}_0)$ has been assumed to be known.

The denominator of (8.17) gives $p_t(\mathbf{y}_t | \mathbf{I}_{t-1}, \mathbf{u}_{t-1})$ (hence, also $p_{t+1}(\mathbf{y}_{t+1} | \mathbf{I}_t, \mathbf{u}_t)$) as, by using (8.15), from (8.17) we obtain

$$\begin{aligned} & \int p_t(\mathbf{y}_t | \mathbf{x}_t, \mathbf{y}_0^{t-1}, \mathbf{u}_0^{t-1}) p_t(\mathbf{x}_t | \mathbf{y}_0^{t-1}, \mathbf{u}_0^{t-1}) d\mathbf{x}_t = p_t(\mathbf{y}_t | \mathbf{y}_0^{t-1}, \mathbf{u}_0^{t-1}) \\ &= p_t(\mathbf{y}_t | \mathbf{y}_0^{t-1}, \mathbf{u}_0^{t-2}, \mathbf{u}_{t-1}) \\ &= p_t(\mathbf{y}_t | \mathbf{I}_{t-1}, \mathbf{u}_{t-1}), \quad t = 1, \dots, T-1. \end{aligned} \quad (8.21)$$

Note that the densities $p_{t+1}(\mathbf{y}_{t+1} | \mathbf{I}_t, \mathbf{u}_t)$ were determined as “by-products” of the computation of $p_t(\mathbf{x}_t | \mathbf{I}_t)$. Therefore, the latter conditional density functions play a key role in applying DP and hence in deriving the optimal control law. It is well known that the LQG assumptions lead directly to the optimal solution of Problem C3 in an analytical way through such conditional densities. The optimal control functions are given by $\mathbf{u}_t^\circ = \mu_t^\circ(\mathbf{I}_t) = -L_t^\circ \hat{\mathbf{x}}_t$, $t = 0, 1, \dots, T-1$, where the gain matrices L_t° are computed by solving the discrete-time Riccati equation and $\hat{\mathbf{x}}_t = \mathbf{E}(\mathbf{x}_t | \mathbf{I}_t)$ is generated by the Kalman filter [13].

8.3 Approximate Solution of Problem C3 by the ERIM

In this section, we analyze the possibility of using the ERIM to derive an approximate solution to Problem C3. As we pointed out, there is a strict parallelism between Problem C2 and Problem C3 and, therefore, between the solving techniques by which we try to face the two problems. This results from the definition of the perfectly measurable information state \mathbf{I}_t that enabled us to devise similar computational tools to obtain the optimal control functions $\mu_0^\circ(x_0), \mu_1^\circ(x_1), \dots, \mu_{T-1}^\circ(x_{T-1})$ and $\mu_0^\circ(\mathbf{I}_0), \mu_1^\circ(\mathbf{I}_1), \dots, \mu_{T-1}^\circ(\mathbf{I}_{T-1})$. If we exploit the aforesaid parallelism, the application of the ERIM by constraining the closed-loop functions (8.6) to take one among different possible structures of FSP functions comes quite naturally, that is,

$$\mathbf{u}_t = \tilde{\mu}_t(\mathbf{I}_t, \mathbf{w}_{tn_t}), \quad t = 0, 1, \dots, T-1. \quad (8.22)$$

The analogy between Control Functions (8.22) and (7.58) is evident. The dependence of the structures of the functions (8.22) on time is now due not only to the possible time-varying regularity properties of the optimal control functions to be approximated but also to the increasing dimension of \mathbf{I}_t over time. The former dependence is taken into account by the parameter vector \mathbf{w}_{tn_t} , the latter by the subscript t in $\tilde{\mu}_t$ (see the Remark on Notations 2.1). Since, in Problem C3, \mathbf{x}_0 is a random variable, the problem is more analogous to Problem C2'_n than to Problem C2_n. Indeed, Problem C2'_n also regards the initial state $\mathbf{x}_0 \in X_0$ as a random vector.

Unfortunately, the increasing dimension of the information vector makes the analogy between Problems C2 and C3 merely formal. Indeed, from the computational point of view, things are very different. We saw that both approximate dynamic pro-

gramming (ADP) and the ERIM are effective numerical tools to solve Problem C2. However, in solving Problem C3, we noted that ADP involves great difficulties owing to (i) the practical impossibility of computing the conditional probability density functions $p_t(\mathbf{x}_t | \mathbf{I}_t)$ and $p_{t+1}(\mathbf{y}_{t+1} | \mathbf{I}_t, \mathbf{u}_t)$ and (ii) the fact that the dimension of \mathbf{I}_t may become too large. In the next section, we shall point out that the first drawback is not present if the ERIM is used, whereas the second drawback causes the same difficulties to both the ERIM and ADP. The general concepts for the various approximate solutions to Problem C3 by the ERIM in this and in the following sections inspired by [19, 20].

8.3.1 Approximation of Problem C3 by the Nonlinear Programming Problems $C3_n$

According to the approach that we already used to derive approximate solutions to Problems C2, C2', and the other optimal control problems stated in Chap. 7, the ERIM replaces Control Functions (8.6) with FSP Control Functions (8.22). Then, by repeatedly using State Equation (8.3) and Observation Equation (8.1), we eliminate the vectors \mathbf{x}_t , \mathbf{u}_t , \mathbf{y}_t , and \mathbf{I}_t in Cost (8.5). This cost turns out to be reduced to a function of the primitive random variables $\mathbf{x}_0, \xi_0, \xi_1, \dots, \xi_{T-1}, \eta_0, \eta_1, \dots, \eta_{T-1}$ and of the parameter vectors $\mathbf{w}_{tn_0}, \mathbf{w}_{tn_1}, \dots, \mathbf{w}_{tn_{T-1}}$.

Let

$$\boldsymbol{\xi} \triangleq \text{col}(\xi_0, \xi_1, \dots, \xi_{T-1}) = \xi_0^{T-1}, \quad (8.23)$$

$$\boldsymbol{\eta} \triangleq \text{col}(\eta_0, \eta_1, \dots, \eta_{T-1}) = \eta_0^{T-1}, \quad (8.24)$$

$$\mathbf{w}_n \triangleq \text{col}(\mathbf{w}_{0n_0}, \mathbf{w}_{1n_1}, \dots, \mathbf{w}_{T-1,n_{T-1}}), \quad (8.25)$$

where the subscript n denotes the global model complexity of the chain of FSP functions. The integer n enables us to enumerate the elements of the infinite nested sequences of sets of admissible FSP functions described in Sect. 2.2. It follows that the cost takes on the form

$$\hat{J}(\mathbf{w}_n) \triangleq \underset{\mathbf{x}_0, \boldsymbol{\xi}, \boldsymbol{\eta}}{\mathbb{E}} J(\mathbf{w}_n, \mathbf{x}_0, \boldsymbol{\xi}, \boldsymbol{\eta}). \quad (8.26)$$

Then, the IDO Problem C3 is approximated by a sequence of nonlinear programming Problems P_n . More specifically, for any increasing value of n , we state these Problems P_n .

Problem C3_n. *Find the vector \mathbf{w}_n° that minimizes Expected Cost (8.26) under the constraint $\mathbf{w}_n \in W_n$.* \triangleleft

As for Problem C2_n, the model complexity n must be chosen experimentally by trial-and-error procedures. Problem C3_n has to be solved under the constraint $\mathbf{w}_n \in W_n$. The set W_n of admissible vectors \mathbf{w}_n is determined by transferring Con-

straints (8.4) (and other possible constraints) into the space of \mathbf{w}_n . State Equation Constraints (8.3) and Measurement Channel Constraints (8.1) are not active any more. Indeed, they were used to eliminate all the variables present in Problem C3 except for the random variables \mathbf{x}_0 , $\boldsymbol{\xi}$, $\boldsymbol{\eta}$, and the parameter vector \mathbf{w}_n . Again, as we pointed out in Remark 7.4 and in the discussion following Procedure 7.1, we emphasize that W_n essentially has a “formal” significance since its computation is a very hard task. Constraints (8.4) will still be interpreted in a “soft way” by introducing, for example, suitable penalty functions. Problem $C3_n$ is then an unconstrained NLP problem. A possible way to solve it will be presented in the next section.

8.3.2 Solution of Problem $C3_n$

Coming back to what we said in Sect. 7.6.1, we also note that Problem $C3_n$ (like Problem $C2_n$) can be solved by batch or stochastic gradient algorithms. Batch algorithms imply the sampling of the sets on which the random vectors \mathbf{x}_0 , $\boldsymbol{\xi}$, and $\boldsymbol{\eta}$ take their values. Since we assumed that Problem $C3_n$ is an unconstrained NLP problem, one can approximate a descent method belonging to the wide class of gradient-based algorithms of the following type (compare with (5.30) and (7.78)):

$$\mathbf{w}_n(k+1) = \mathbf{w}_n(k) - \alpha_k S_k \nabla_{\mathbf{w}_n} \underset{\mathbf{x}_0, \boldsymbol{\xi}, \boldsymbol{\eta}}{\text{E}} J[\mathbf{w}_n(k), \mathbf{x}_0, \boldsymbol{\xi}, \boldsymbol{\eta}], \quad k = 0, 1, \dots \quad (8.27)$$

Of course, all the conditions ensuring that the cost J and its expected value $\text{E}(J)$ are functions of class \mathcal{C}^1 with respect to \mathbf{w}_n must be verified (see Theorem 5.2 and extend suitably Assumption 7.1). As in Algorithm (7.78), Approximation (8.27) is required by the practical impossibility of computing the expected value $\text{E}(J)$ in an analytical way. However, according to (5.32) and (5.33), we approximate $\text{E}(J)$ as follows, thus obtaining (see (7.79))

$$\begin{aligned} \mathbf{w}_n(k+1) &= \mathbf{w}_n(k) \\ &- \alpha_k S_k \frac{1}{L_1 L_2 L_3} \sum_{l_1=1}^{L_1} \sum_{l_2=1}^{L_2} \sum_{l_3=1}^{L_3} \nabla_{\mathbf{w}_n} J[\mathbf{w}_n(k), \mathbf{x}_0^{(l_1)}, \boldsymbol{\xi}^{(l_2)}, \boldsymbol{\eta}^{(l_3)}], \quad k = 0, 1, \dots \end{aligned} \quad (8.28)$$

(as in (7.79), we assumed that the sample sets $\{\mathbf{x}_0^{(l_1)}\}_{l_1=1}^{L_1}$, $\{\boldsymbol{\xi}_0^{(l_2)}\}_{l_2=1}^{L_2}$, and $\{\boldsymbol{\eta}_0^{(l_3)}\}_{l_3=1}^{L_3}$ are not changed at any iteration k).

Instead of applying Batch Algorithm (8.28), we can resort to stochastic approximation. Indeed, in the numerical examples presented in this and in the following chapters, the stochastic gradient algorithm provided good results. Therefore, in what follows, we shall concentrate on it. The algorithm takes on the form (see (7.80))

$$\begin{aligned} \mathbf{w}_n(k+1) &= \mathbf{w}_n(k) - \alpha_k \nabla_{\mathbf{w}_n} J[\mathbf{w}_n(k), \mathbf{x}_0(k), \boldsymbol{\xi}(k), \boldsymbol{\eta}(k)], \\ &\quad k = 0, 1, \dots \end{aligned} \quad (8.29)$$

The sequences $\mathbf{x}_0(k)$, $\boldsymbol{\xi}(k)$, and $\boldsymbol{\eta}(k)$, $k = 0, 1, \dots$, are generated randomly according to the probability density functions of \mathbf{x}_0 , $\boldsymbol{\xi}$, and $\boldsymbol{\eta}$, respectively. As regards the decreasing stepsize α_k , the convergence properties of (8.29), and other comments on this algorithm, we refer to what we said about (7.80) in Sect. 7.6.1.

At each iteration step of (8.29), we have now to derive the components of the gradient in (8.29), i.e., the partial derivatives (see (7.82))

$$\frac{\partial J}{\partial w_{tn_t}^i}, \quad t = 0, 1, \dots, T-1, \quad i = 1, 2, \dots, \mathcal{N}(n_t), \quad (8.30)$$

where $w_{tn_t}^i$ is the i th component of the vector \mathbf{w}_{tn_t} and $\mathcal{N}(n_t)$ is its dimension, that is, the number of parameters to be optimized in each FSP function (8.22). We recall that in Sect. 7.6.2 we considered the cost-to-go J_t . Obviously, we have $\frac{\partial J}{\partial w_{tn_t}^i} = \frac{\partial J_t}{\partial w_{tn_t}^i}$.

The same holds for the partial derivatives of J and J_t with respect to the other variables. Instead of considering general FSP functions for the solution of Problems C2_n and C2'_n as in Sect. 7.6, in the next section we shall specialize the computation of the partial derivatives necessary to apply the stochastic gradient algorithm (i.e., the derivatives (8.30)) to the case of MHL networks. This choice is justified by the following facts: (i) these networks are used in the examples reported later in this chapter; (ii) the algebra involved is rather simple thanks to the possibility of using the back-propagation procedure; and (iii) most of the conceptual results can be extended to other types of FSP functions in a straightforward way.

8.3.3 Specialization of Stochastic Gradient Algorithms When MHL Networks are Used

In order to express the partial derivatives (8.30), we shall follow a computational scheme similar to the one used in Sect. 7.6.3 to compute the derivatives (7.101). We begin by establishing a connection among the vectors \mathbf{u}_t and \mathbf{I}_t and the input/output vectors of the MHL networks

$$\mathbf{u}_t = \tilde{\mu}_t(\mathbf{I}_t, \mathbf{w}_{tn_t}), \quad t = 0, 1, \dots, T-1. \quad (8.31)$$

Of course, Functions (8.31) and (8.22), which are expressed as FSP functions, are formally similar. We can write (compare with (7.99) and (7.100))

$$\mathbf{u}_t = \mathbf{y}^t(L), \quad t = 0, 1, \dots, T-1,$$

for the output vectors of the networks $\tilde{\mu}_t$, and

$$\mathbf{I}_t = \mathbf{y}^t(0), \quad t = 0, 1, \dots, T-1, \quad (8.32)$$

for the input vectors of these networks.

We now repeat the same computational steps that led to the partial derivatives (7.101) through the application of the backpropagation procedure. More specifically, by applying the backpropagation rule to the networks $\tilde{\mu}_t$, the partial derivatives (8.31) can be written as follows (we drop the index k and use the notation introduced in Sect. 7.6.3; to simplify the notation, here and in the following, we assume L to be time-invariant):

$$\frac{\partial J}{\partial w_{pq}^t(s)} = y_p^t(s-1)\delta_q^t(s), \quad (8.33)$$

where

$$\delta_q^t(s) \triangleq \frac{\partial J}{\partial z_q^t(s)}, \quad t = 1, \dots, T-1, \quad s = 1, \dots, L, \quad q = 1, \dots, n_s. \quad (8.34)$$

The components $w_{tn_t}^i$ of the vectors \mathbf{w}_{tn_t} (see (8.25)) are replaced with the components $w_{pq}^t(s)$ of the same vectors (we dropped the subscripts n_t). $\delta_q^t(s)$ can be computed recursively by the equations

$$\delta_q^t(s) = h'[z_q^t(s)] \sum_{k=1}^{n_{s+1}} \delta_k^t(s+1) w_{qk}^t(s+1), \quad s = 1, \dots, L-1, \quad (8.35a)$$

$$\delta_q^t(L) = h'[z_q^t(L)] \frac{\partial J}{\partial y_q^t(L)}. \quad (8.35b)$$

For simplicity, we omit the computation of the partial derivatives with respect to the bias weights $w_{0q}(s)$. To initialize the backward procedure, we have then to compute the partial derivatives

$$\frac{\partial J}{\partial \mathbf{y}^t(L)} \triangleq \text{col} \left(\frac{\partial J}{\partial y_q^t(L)}, q = 1, \dots, m \right). \quad (8.36)$$

We obtain

$$\begin{aligned} \frac{\partial J}{\partial \mathbf{y}^t(L)} &= \frac{\partial J}{\partial \mathbf{u}_t} = \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t)}{\partial \mathbf{u}_t} + \frac{\partial}{\partial \mathbf{u}_t} \left[\sum_{k=t+1}^{t+T-1} h_k(\mathbf{x}_k, \mathbf{u}_k, \boldsymbol{\xi}_k) + h_T(\mathbf{x}_T) \right], \\ &t = 0, 1, \dots, T-1. \end{aligned} \quad (8.37)$$

Note that the second term in the right-hand side of (8.37) represents the derivative of the cost-to-go J_{t+1} . This cost is influenced by \mathbf{u}_t through the state \mathbf{x}_{t+1} , generated by the state equation and the neural networks $\tilde{\mu}_j$ ($j = t+1, \dots, T-1$), which have \mathbf{u}_t as their input (\mathbf{u}_t is present in the information vector \mathbf{I}_j).

Let us now establish a correspondence between $\mathbf{I}_t = \mathbf{y}^t(0)$ and the subvectors of $\mathbf{y}^t(0)$ given by the measures on the state and the controls that are the inputs to the network $\tilde{\boldsymbol{\mu}}_t$. Using the notation introduced in Sect. 7.6, we have

$$\mathbf{I}_t = \mathbf{y}^t(0) = \text{col} [\mathbf{y}_{y_j}^t(0), j = 0, 1, \dots, t, \mathbf{y}_{u_j}^t(0), j = 0, 1, \dots, t - 1],$$

where $\mathbf{y}_{y_j}^t(0) = \mathbf{y}_j$, $j = 0, 1, \dots, t$, and $\mathbf{y}_{u_j}^t(0) = \mathbf{u}_j$, $j = 0, 1, \dots, t - 1$.

On the basis of the foregoing, the partial derivatives (8.37) can be written as follows:

$$\frac{\partial J}{\partial \mathbf{y}^t(L)} = \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{u}_t} + \lambda_{t+1}^\top \frac{\partial f_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{u}_t} + \sum_{j=t+1}^{T-1} \frac{\partial J}{\partial \mathbf{y}_{u_j}^j(0)}, \\ t = 0, 1, \dots, T - 2, \quad (8.38)$$

$$\frac{\partial J}{\partial \mathbf{y}^{T-1}(L)} = \frac{\partial h_{T-1}(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}, \xi_{T-1})}{\partial \mathbf{u}_{T-1}} + \lambda_T^\top \frac{\partial f_{T-1}(\mathbf{x}_{T-1}, \mathbf{u}_{T-1}, \xi_{T-1})}{\partial \mathbf{u}_{T-1}}. \quad (8.39)$$

The second term in the right-hand side of (8.38) expresses the influence of \mathbf{u}_t on J through \mathbf{x}_{t+1} (see (7.88)). As in (7.86), we define

$$\lambda_t^\top \triangleq \frac{\partial J}{\partial \mathbf{x}_t}, \quad t = 0, 1, \dots, T. \quad (8.40)$$

To compute the vectors λ_t , note that \mathbf{x}_t directly influences h_t and the remaining part of the cost J (see (8.37)) through \mathbf{x}_{t+1} and \mathbf{y}_t . Then, we can write

$$\lambda_t^\top = \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{x}_t} + \frac{\partial J}{\partial \mathbf{x}_{t+1}} \frac{\partial f_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{x}_t} + \frac{\partial J}{\partial \mathbf{y}_t} \frac{\partial g_t(\mathbf{x}_t, \eta_t)}{\partial \mathbf{x}_t}. \quad (8.41)$$

We have now to determine $\frac{\partial J}{\partial \mathbf{y}_t}$. Note that the vector \mathbf{y}_t exerts its influence on the networks $\tilde{\boldsymbol{\mu}}_t$ in the same way as described for \mathbf{u}_t in deriving (8.38). Then, as it is $\mathbf{y}_{y_t}^j(0) = \mathbf{y}_t$, $j = t, \dots, T - 1$, we can write

$$\lambda_t^\top = \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{x}_t} + \lambda_{t+1}^\top \frac{\partial f_t(\mathbf{x}_t, \mathbf{u}_t, \xi_t)}{\partial \mathbf{x}_t} \\ + \sum_{j=t}^{T-1} \frac{\partial J}{\partial \mathbf{y}_{y_t}^j(0)} \frac{\partial g_t(\mathbf{x}_t, \eta_t)}{\partial \mathbf{x}_t}, \quad t = 0, 1, \dots, T - 1, \quad (8.42a)$$

$$\lambda_T^\top = \frac{\partial h_T(\mathbf{x}_T)}{\partial \mathbf{x}_T}. \quad (8.42b)$$

Finally, we have to compute $\frac{\partial J}{\partial \mathbf{y}_{u_t}^j(0)}$ and $\frac{\partial J}{\partial \mathbf{y}_{y_t}^j(0)}$ (i.e., the partial derivatives of the cost J with respect to the input vectors to the MHL networks $\tilde{\mu}_j$), which are present in (8.38) and (8.42a). Let us determine $\frac{\partial J}{\partial \mathbf{y}_{u_t}^j(0)}$. By using (7.98) and (8.34), we can express the p th component of $\frac{\partial J}{\partial \mathbf{y}_{u_t}^j(0)}$ as

$$\frac{\partial J}{\partial y_{u_t}^{jp}(0)} = \sum_{q=1}^{n_j} \frac{\partial J}{\partial z_q^j(1)} \frac{\partial z_q^j(1)}{\partial y_{u_t}^{jp}(0)} = \sum_{q=1}^{n_j} \delta_q^j(1) w_{pq}^j(1), \quad p \in P_{u_t}^j, \quad (8.43)$$

where, for notational simplicity, we assume that all the layers of the networks have the same number of neural units and $P_{u_t}^j$ is the subset of indexes p of the components $y_{u_t}^{jp}(0)$ that correspond to the components of $\mathbf{y}_{u_t}^j(0) = \mathbf{u}_t$. Then, we obtain

$$\frac{\partial J}{\partial \mathbf{y}_{u_t}^j(0)} = \text{col} \left(\sum_{q=1}^{n_j} \delta_q^j(1) w_{pq}^j(1), p \in P_{u_t}^j \right),$$

$$t = 0, 1, \dots, T-1, \quad j = 1, \dots, T-1. \quad (8.44)$$

A similar relationship can be derived to compute $\frac{\partial J}{\partial \mathbf{y}_{y_t}^j(0)}$.

Now, let us briefly describe the optimization mechanism (“learning”) of the neural control law. We summarize what we described in Procedure 7.2 in the following two “phases” that alternate up to a possible convergence:

Forward phase. At step k , the random vectors $\mathbf{x}_0(k)$, $\boldsymbol{\xi}(k)$, and $\boldsymbol{\eta}(k)$ are randomly generated according to their respective probability density functions. The control networks (specified by the weight vector $\mathbf{w}(k)$) compute the trajectories of the control, state, and information vectors.

Backward phase. The variables $\delta_q^t(s)$ and λ_t are determined by the backpropagation equations (8.34), (8.35) (initialized by (8.38) and (8.39)). The variables $\delta_q^t(s)$ enable one to compute the partial derivatives $\frac{\partial J}{\partial w_{pq}^t(s)}$ (see (8.33)), hence the gradient in (8.29). Then, the new weight vector $\mathbf{w}_n(k+1)$ is obtained by (8.29).

Remark 8.1 As pointed out in Remark 7.9, it is worth noting that (8.38) is the classical adjoint equation of T -stage optimal control theory, with the addition of one term (the third in (8.38)) to take into account the introduction of the FSP feedback control functions. Note also that this term does not depend on the particular structure of the FSP control functions, i.e., it is not specific for MHL networks. The presence of the MHL networks is revealed by (8.44), which includes the synaptic weights of the first layer of the networks. \triangleleft

Remark 8.2 As pointed out previously, in comparison with ADP, the ERIM has the remarkable advantage of not requiring that the density functions $p_t(\mathbf{x}_t | \mathbf{I}_t)$ (see (8.20)) and $p_{t+1}(\mathbf{y}_{t+1} | \mathbf{I}_t, \mathbf{u}_t)$ (see (8.21)) be calculated. Indeed, let us explicitly emphasize that the ERIM only needs the knowledge of the probability density functions of the primitive random variables. However, the ever-increasing dimension of the information vector \mathbf{I}_t in Control Functions (8.6), and then in (8.22), is a drawback that cannot be overcome either by ADP or the ERIM. \triangleleft

8.4 Approximation of Problem C3 by Limiting the Dimension of the Information State Vector and by Applying the ERIM

To avoid the drawback of the ever-increasing dimension of the information vector \mathbf{I}_t , even if in approximate form, the DM has to give up on the complete memory of all the control and measurement vectors since the initial stage $t = 0$. This implies that the DM only recalls (i.e., it stores in its memory) the controls and the measures belonging to the most recent past. The introduction of a *limited-memory* (LM) *information vector* as well as the introduction of a *memory vector*, generated by a suitable dynamic system, is the subject of this section.

8.4.1 Limited-Memory Information Vector

We use the term “limited-memory information vector” (of time-invariant dimension) to refer to the following collection of measures and controls:

$$\mathbf{I}_t^M \triangleq \text{col} (\mathbf{y}_{t-M+1}, \dots, \mathbf{y}_t, \mathbf{u}_{t-M+1}, \dots, \mathbf{u}_{t-1}), \quad t = M, \dots, T-1. \quad (8.45)$$

The integer M denotes the length of the “sliding window” of data that the DM stores in its memory. This memory stores M measurement vectors \mathbf{y}_t and $M - 1$ control vectors \mathbf{u}_t . Of course, at $t = 0, 1, \dots, M - 1$, the DM is able to recall the information vector \mathbf{I}_t . Then, at stage $t = M$, it begins to forget data. Concerning state estimators and controllers based on the use of limited memory, we refer the reader, for example, to the pioneering work [12].

The abrupt truncation of \mathbf{I}_t , of which we only keep the “most recent past” \mathbf{I}_t^M , may perhaps be mitigated by introducing a vector s_t by means of which the DM can maintain some form of memory of the measures and controls that were lost at the stage $t - M$. We define s_t as the *memory vector*. In order to retain a “trace of the history” of the decision process from the stage $t = 0$, it is reasonable to generate s_t by a dynamic system. A possible way to define the vector s_t and the mechanism that propagates it is described in the following paragraphs. The related control scheme

provided good numerical results; it allowed the design of a control law that performed better than the one based only on the LM information vector \mathbf{I}_t^M .

On the basis of the proposed control scheme, we assume that the DM generates the control vector by using \mathbf{I}_t till the stage t at which $\dim(\mathbf{I}_t) = \dim(\mathbf{I}_t^M)$. This occurs when

$$\dim(\mathbf{I}_t) = (t+1)p + tm = \dim(\mathbf{I}_t^M) = Mp + (M-1)m, \quad (8.46)$$

that is, at the stage $M-1$. From the stage M on, the DM exploits the LM information vector \mathbf{I}_t^M and \mathbf{s}_{t-M} . Then, the control functions take on the forms

$$\mathbf{u}_t = \mu_t(\mathbf{I}_t), \quad t = 0, 1, \dots, M-1, \quad (8.47a)$$

$$\mathbf{u}_t = \mu_t(\mathbf{I}_t^M, \mathbf{s}_{t-M}), \quad t = M, \dots, T-1. \quad (8.47b)$$

To fix ideas, we may suppose that the DM has two buffers B_1 and B_2 of capacity C_1 and C_2 , respectively, where it stores an integer number of real variables. In B_1 , the DM stores the LM information vector \mathbf{I}_t^M , whose dimension is time-invariant. $\dim(\mathbf{I}_t)$ increases up to the stage M . Then, $C_1 = Mp + (M-1)m$ (see (8.46)). In buffer B_2 , the DM stores the memory vector \mathbf{s}_t . The storing mechanism of this buffer depends on the values of C_1 and C_2 . Without loss of generality, we may assume that, for suitable values of C_1 and C_2 , \mathbf{s}_t is given by

$$\mathbf{s}_t = \text{col}(\mathbf{I}_t, \mathbf{u}_t), \quad t = 0, 1, \dots, t^*-1, \quad (8.48)$$

where t^* is the first stage at which \mathbf{s}_t reaches the time-invariant dimension of $\text{col}(\mathbf{I}_t^M, \mathbf{u}_t) = M(p+m) \leq C_2$. From t^* on, \mathbf{s}_t is generated by the dynamic model

$$\mathbf{s}_t = \varphi_t(\mathbf{I}_t^M, \mathbf{u}_t, \mathbf{s}_{t-M}), \quad t = t^*, \dots, T-M-1. \quad (8.49)$$

Denote by M_s the maximum time-invariant dimension of \mathbf{s}_t . The functions φ_t are called *memory functions*.

Clearly, we may choose that \mathbf{s}_t does not depend on \mathbf{u}_t (if we were to do so, only minor changes would be required in the following treatment). However, it is clear that the expected value of Cost (8.5), obtained by minimizing it with respect to Functions (8.47) and (8.49) cannot improve. Indeed, the functions $\mathbf{s}_t = \varphi_t(\mathbf{I}_t^M, \mathbf{s}_{t-M})$ would be a special case of Functions (8.49).

We can now state the LM approximate version of Problem C3. By “control law” we mean the sequences of both Control (8.47) and Memory Functions (8.49).

Problem C3-LM. *Find the optimal control law $\{\mu_t^\circ(\mathbf{I}_t), t = 0, 1, \dots, M-1; \mu_t^\circ(\mathbf{I}_t^M, \mathbf{s}_{t-M}), t = M, M+1, \dots, T-1; \varphi_t^\circ(\mathbf{I}_t^M, \mathbf{u}_t, \mathbf{s}_{t-M}), t = t^*, t^*+1, \dots, T-M-1\}$ that minimizes Cost Functional (8.7) subject to Constraints (8.1), (8.3), and (8.4).* \triangleleft

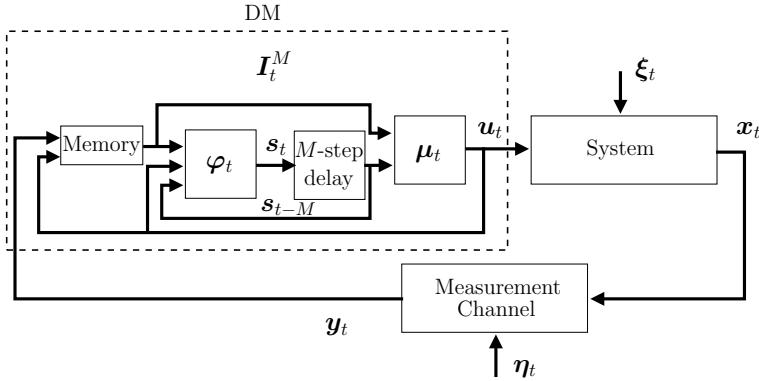


Fig. 8.2 Control scheme of a DM using the limited-memory information vectors \mathbf{I}_t^M and the memory functions φ_t

It is important to determine the first stage after which both the control and memory functions depend on vectors characterized by time-invariant dimensions. We denote this stage by t_1 . Let us consider Functions (8.47b) and (8.49). We verified that \mathbf{I}_t reaches its time-invariant dimension at the stage $t = M$ when it is replaced by the LM vector \mathbf{I}_t^M . Then, we only have to compute the first stage at which s_{t-M} reaches its maximum (and stationary) dimension M_s . For s_t , this occurs at the stage $t = t^*$. Therefore, for s_{t-M} , the same occurs M stages later, that is, at stage $t_1 = t^* + M$.

The control scheme addressed in Problem C3-LM is shown in Fig. 8.2. The figure refers to the stages $t \geq t_1$, i.e., when the control and memory functions take on the forms (8.47b) and (8.49), respectively. It shows the fact that the control scheme makes use of the pairs $(\mathbf{I}_t^M, s_{t-M})$, which might be called *approximate sufficient statistics*. This term is used in other decision contexts; see, for instance, [14] and the references cited therein. Of course, \mathbf{I}_t and $p_t(\mathbf{x}_t | \mathbf{I}_t)$ are “exact sufficient statistics.”

Remark 8.3 No separation property was enforced by introducing s_t and its dynamic generation. In short, we say that a *separation property* holds true whenever the problem of designing an optimal controller can be divided into two subproblems. One is the synthesis of an optimal (in a proper sense) estimator of the state vector on the basis of the variables coming from the measurement devices, and the other is the synthesis of an optimal control law that generates the decisions on the basis of the state estimates. As is well known, the separation property characterizes Problem C3 under LQG assumptions. However, it does not apply to Problem C3-LM. Indeed, (i) the pair $(\mathbf{I}_t^M, s_{t-M})$ cannot be at all considered as an estimate of the state \mathbf{x}_t ; and (ii) $(\mathbf{I}_t^M, s_{t-M})$ is an input not only to the memory function φ_t but also to the control function μ_t . \triangleleft

It should be noted that Problem C3-LM is related to the basic Problem PM. The total number of functions to be minimized is given by the number T of Control Functions (8.47) plus the number $T - M - t^*$ of Memory Functions (8.49).

8.4.2 Approximate Solution of Problem C3-LM by the ERIM

Once Problem C3 has been approximated by Problem C3-LM, we can apply the ERIM to the latter through the usual procedure. Thanks to the replacement of the information vectors \mathbf{I}_t with \mathbf{I}_t^M and s_{t-M} , we have to deal with functions that again depend on a time-invariant number of variables. Of course, we have “to parametrize” two sequences of functions (i.e., (8.47) and (8.49)) instead of a single one (i.e., (7.58)) but this operation only involves computational, not conceptual difficulties.

Let us constrain Functions (8.47) and (8.49) to take on the structures of FSP functions. Then we have

$$\mathbf{u}_t = \tilde{\mu}_t(\mathbf{I}_t, \mathbf{w}_{tn_t}), \quad t = 0, 1, \dots, M-1 \quad (8.50a)$$

$$\mathbf{u}_t = \tilde{\mu}_t(\mathbf{I}_t^M, s_{t-M}, \mathbf{w}_{tn_t}), \quad t = M, \dots, T-1 \quad (8.50b)$$

$$\mathbf{s}_t = \text{col}(\mathbf{I}_t, \mathbf{u}_t), \quad t = 0, 1, \dots, t^*-1, \quad (8.51a)$$

$$\mathbf{s}_t = \tilde{\varphi}_t(\mathbf{I}_t^M, \mathbf{u}_t, s_{t-M}, \tilde{\mathbf{w}}_{t\tilde{n}_t}), \quad t = t^*, \dots, T-M-1, \quad (8.51b)$$

where \mathbf{w}_{tn_t} and $\tilde{\mathbf{w}}_{t\tilde{n}_t}$ are the parameter vectors to be optimized. We added the subscript t in the FSP functions (8.50) and (8.51b). We recall that the dimensions of both \mathbf{I}_t^M and s_{t-M} become time-invariant at the stage $t_1 = t^* + M$. Accordingly, for $t \geq t_1$, the structures of FSP Functions (8.50b) and (8.51b) reach their steady-state structures. Thus, for $t \geq t_1$, the subscript t in the FSP functions $\tilde{\mu}_t$ and $\tilde{\varphi}_t$ is not necessary any more.

Now, by replacing Functions (8.47) and (8.49) with FSP Functions (8.50) and (8.51b) and repeatedly using State Equation (8.3) and Observation Equation (8.1), we eliminate the vectors \mathbf{x}_t , \mathbf{u}_t , \mathbf{y}_t , \mathbf{s}_t , \mathbf{I}_t , and \mathbf{I}_t^M in Cost (8.5). After these operations, Cost (8.7) does not depend on the control functions any more but becomes a function of the variables left out from the elimination. These variables are the parameter vectors \mathbf{w}_{tn_t} ($t = 0, 1, \dots, T-1$) and $\tilde{\mathbf{w}}_{t\tilde{n}_t}$ ($t = t^*, \dots, T-M-1$) and the primitive random vectors \mathbf{x}_0 , ξ , and η . Let

$$\mathbf{w}_n \triangleq \text{col}(\mathbf{w}_{tn_t}, t = 0, 1, \dots, T-1, \tilde{\mathbf{w}}_{t\tilde{n}_t}, t = t^*, \dots, T-1-M), \quad (8.52)$$

where the subscript n denotes the global model complexity of the set of FSP Functions (8.50) and (8.51b).

Then, Cost (8.7) can be written as follows:

$$\hat{J}_n(\mathbf{w}_n) \triangleq \underset{\mathbf{x}_0, \xi, \eta}{\mathbb{E}} J(\mathbf{w}_n, \mathbf{x}_0, \xi, \eta), \quad (8.53)$$

which is formally identical to Cost (8.26).

Once again, we approximate the functional optimization Problem C3-LM by the following sequence of NLP problems, similar to Problem P_n .

Problem C3_n-LM. Find the vector \mathbf{w}_n^* that minimizes Expected Cost (8.53) under the constraint $\mathbf{w}_n \in W_n$. \triangleleft

The remarks that we made on Problem C3_n about the constraint $\mathbf{w}_n \in W_n$ can be repeated for Problem C3_n-LM. Here again, the state equation and the measurement channel constraints are no longer needed whereas Constraints (8.4) are transferred in the belonging of \mathbf{w}_n to a suitable set W_n . Actually, in the next section, they will be interpreted in a “soft way”: for example, by the introduction of suitable penalty functions.

8.4.3 Solution of Problem C3_n-LM by Stochastic Approximation and Specialization to MHL Networks

As in Sect. 8.3.2, we solve Problem C3_n-LM by applying the stochastic gradient algorithm. This means that, here again, we replace (8.27) with the algorithm

$$\mathbf{w}_n(k+1) = \mathbf{w}_n(k) - \alpha_k \nabla_{\mathbf{w}_n} J[\mathbf{w}_n(k), \mathbf{x}_0(k), \boldsymbol{\xi}(k), \boldsymbol{\eta}(k)], \quad k = 0, 1, \dots. \quad (8.54)$$

Algorithms (8.29) and (8.54) are formally identical. Of course, Costs (8.53) and (8.26) are different.

At each iteration step k of (8.29), we derive the components of the gradient in (8.54), that is,

$$\frac{\partial J}{\partial w_{tn_t}^i}, \quad t = 0, 1, \dots, T-1, i = 1, \dots, \mathcal{N}_1(n_t), \quad (8.55)$$

$$\frac{\partial J}{\partial \tilde{w}_{t\tilde{n}_t}^i}, \quad t = t^*, \dots, T-M-1, i = 1, \dots, \mathcal{N}_2(\tilde{n}_t), \quad (8.56)$$

where $w_{tn_t}^i$ is the i th component of the vector \mathbf{w}_{tn_t} and $\mathcal{N}_1(n_t)$ is the number of parameters to be optimized in each MHL Network (8.50). Analogously, $\tilde{w}_{t\tilde{n}_t}^i$ is the i th component of the vector $\tilde{\mathbf{w}}_{t\tilde{n}_t}$ (of dimension $\mathcal{N}_2(\tilde{n}_t)$) to be optimized in each MHL Network (8.51b). We note that two chains of MHL networks have to be considered: the chain of networks (8.50), generating the control vectors \mathbf{u}_t , and the chain of networks (8.51b), generating the memory vectors \mathbf{s}_t ; see also Fig. 8.3, where the control scheme of Fig. 8.2 is shown in its “unfolded” form (compare it with Fig. 7.2a).

We express Partial Derivatives (8.55) and (8.56) by establishing a connection among the vectors \mathbf{u}_t , \mathbf{s}_t , and \mathbf{I}_t^M and the input/output vectors of the MHL networks $\tilde{\mu}_t$ and $\tilde{\varphi}_t$ (8.50) and (8.51b). We have

$$\mathbf{u}_t = \mathbf{y}^t(L), \quad t = 0, 1, \dots, T-1, \quad (8.57a)$$

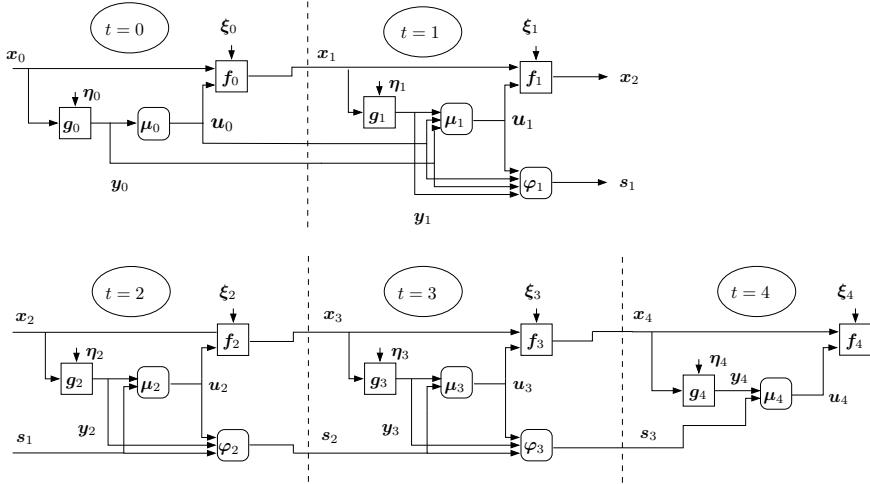


Fig. 8.3 The limited-memory control scheme of Fig. 8.2 is shown in its unfolded form. The control and output variables are assumed to be scalar, thus $m = p = 1$. We set $T = 5$, $M = 1$, and $M_s = 2$; then $t^* = 1$, and $t_I = 2$

$$\mathbf{I}_t = \mathbf{y}^t(0), \quad t = 0, 1, \dots, M-1, \quad (8.57b)$$

$$\text{col}(\mathbf{I}_t^M, \mathbf{s}_{t-M}) = \mathbf{y}^t(0), \quad t = M, \dots, T-1, \quad (8.57c)$$

for the networks $\tilde{\mu}_t$, and

$$s_t = \tilde{\mathbf{y}}^t(\tilde{L}), \quad t = t^*, \dots, T-M-1, \quad (8.58a)$$

$$\text{col}(\mathbf{I}_t^M, \mathbf{u}_t, \mathbf{s}_{t-M}) = \tilde{\mathbf{y}}^t(0), \quad t = t^*, \dots, T-M-1, \quad (8.58b)$$

for the FSP Functions $\tilde{\varphi}_t$.

We now apply the backpropagation procedure to express Partial Derivatives (8.55) and (8.56). As regards (8.55), we can use Relationships (8.33)–(8.35). In order to compute (8.56), the application of the Backpropagation rule to the memory networks $\tilde{\varphi}_t$ yields

$$\frac{\partial J}{\partial \tilde{w}_{pq}^t(s)} = \tilde{y}_p^t(s-1) \tilde{\delta}_q^t(s), \quad (8.59)$$

where

$$\tilde{\delta}_q^t(s) \triangleq \frac{\partial J}{\partial \tilde{z}_q^t(s)}, \quad t = t^*, \dots, T-M-1, \quad s = 1, \dots, \tilde{L}, \quad q = 1, \dots, \tilde{n}_s. \quad (8.60)$$

Here again, we omit the computation of the partial derivatives with respect to the bias weights $\tilde{w}_{0q}^t(s)$.

As before, $\tilde{\delta}_q^t(s)$ can be computed recursively by the equations

$$\tilde{\delta}_q^t(s) = h'[\tilde{z}_q^t(s)] \sum_{k=1}^{\tilde{n}_{s+1}} \tilde{\delta}_k^t(s+1) \tilde{w}_{qk}^t(s+1), \quad s = 1, \dots, \tilde{L}-1, \quad (8.61a)$$

$$\tilde{\delta}_q^t(\tilde{L}) = h'[\tilde{z}_q^t(\tilde{L})] \frac{\partial J}{\partial \tilde{y}_q^t(\tilde{L})}. \quad (8.61b)$$

Then, to initialize the backpropagation phase, the following partial derivatives must be computed:

$$\frac{\partial J}{\partial \mathbf{y}^t(L)} \triangleq \text{col} \left(\frac{\partial J}{\partial y_q^t(L)}, q = 1, \dots, m \right), \quad (8.62)$$

$$\frac{\partial J}{\partial \tilde{\mathbf{y}}^t(\tilde{L})} \triangleq \text{col} \left(\frac{\partial J}{\partial \tilde{y}_q^t(\tilde{L})}, q = 1, \dots, M_s \right).$$

Derivatives (8.62), which are the same as in (8.36), are computed by considering (8.37). We again address the second term in (8.37), which is the derivative of the cost-to-go J_{t+1} with respect to \mathbf{u}_t . This cost is influenced by \mathbf{u}_t through: (i) the state \mathbf{x}_{t+1} , (ii) the network $\tilde{\varphi}_t$, which has \mathbf{u}_t as its input (see (8.51b)), and (iii) the networks $\tilde{\mu}_j$ and $\tilde{\varphi}_j$ ($j = t+1, \dots, t+M-1$) which have \mathbf{u}_t as their input (\mathbf{u}_t is present in the LM information vectors \mathbf{I}_j^M , $j = t+1, \dots, t+M-1$ as can be seen in (8.50b) and (8.51b)). It may be helpful to refer to Fig. 8.4, which details a part of Fig. 8.3.

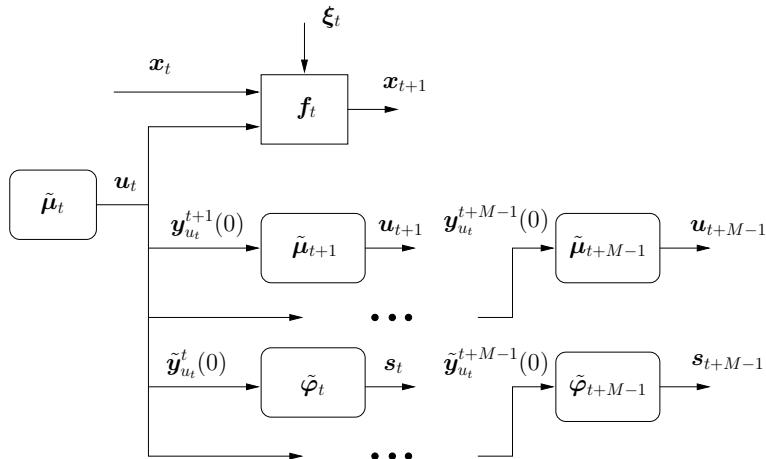


Fig. 8.4 The control vector \mathbf{u}_t influences the cost-to-go through the state equation and two chains of networks

As in Sect. 8.3.3, we establish a correspondence between the input vector to the network $\tilde{\mu}_t$, specified by the vector on the left-hand side of (8.57c) (i.e., $I_t^M \triangleq \text{col}(\mathbf{y}_{t-M+1}^t, \mathbf{u}_{t-M+1}^{t-1})$ and s_{t-M}), and the components of the input vector to this network $\tilde{\mu}_t$, specified by the vector on the right-hand side of (8.57c), i.e., $\mathbf{y}^t(0)$. A similar correspondence has to be established for the networks $\tilde{\varphi}_t$. Toward this end, we define the components of $\mathbf{y}^t(0)$ and $\tilde{\mathbf{y}}^t(0)$ as follows:

1. From (8.57c), we define $\mathbf{y}_{y_j}^t(0)$, $j = t - M + 1, \dots, t$, $\mathbf{y}_{u_j}^t(0)$, $j = t - M + 1, \dots, t - 1$, and $\mathbf{y}_{s_{t-M}}^t(0)$ as the subvectors of $\mathbf{y}^t(0)$ that correspond to the vectors \mathbf{y}_j , \mathbf{u}_j , and s_{t-M} , respectively.
2. From (8.58b), we define $\tilde{\mathbf{y}}_{y_j}^t(0)$, $j = t - M + 1, \dots, t$, $\tilde{\mathbf{y}}_{u_j}^t(0)$, $j = t - M + 1, \dots, t - 1$, and $\tilde{\mathbf{y}}_{s_{t-M}}^t(0)$ as the subvectors of $\tilde{\mathbf{y}}^t(0)$ that correspond to the vectors \mathbf{y}_j , \mathbf{u}_j , and s_{t-M} , respectively.

Then, for $t^* \leq t \leq T - M - 1$, Partial Derivatives (8.37) can be rewritten as follows (compare with (8.38)):

$$\begin{aligned} \frac{\partial J}{\partial \mathbf{y}^t(L)} &= \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t)}{\partial \mathbf{u}_t} + \boldsymbol{\lambda}_t^\top \frac{\partial f_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t)}{\partial \mathbf{u}_t} \\ &\quad + \frac{\partial J}{\partial \tilde{\mathbf{y}}_{u_t}^t(0)} + \sum_{j=t+1}^{t+M-1} \left(\frac{\partial J}{\partial \mathbf{y}_{u_t}^j(0)} + \frac{\partial J}{\partial \tilde{\mathbf{y}}_{u_t}^j(0)} \right). \end{aligned} \quad (8.63)$$

Equation (8.63) holds, with some minor modifications, also for the other values of t . In particular, the partial derivatives $\frac{\partial J}{\partial \mathbf{y}_{u_t}^j(0)}$ are present in the summation in (8.63) provided that (see (8.50b))

$$M \leq t \leq T - 1.$$

The partial derivatives $\frac{\partial J}{\partial \tilde{\mathbf{y}}_{u_t}^j(0)}$ are present in the right-hand side of (8.63) provided that (see (8.51b))

$$t^* \leq t \leq T - M - 1.$$

$\boldsymbol{\lambda}_t$ is defined as in (8.40) and is computed via (8.41). Concluding, some simplifications in (8.63) hold for the other values of $t = 0, 1, \dots, T - 1$.

The partial derivatives $\frac{\partial J}{\partial \tilde{\mathbf{y}}^t(\tilde{L})}$ can be computed by considering that $\tilde{\mathbf{y}}^t(\tilde{L}) = s_t$ influences the cost J at stage $t + M$ by acting on both the network $\tilde{\mu}_{t+M}$ (note that $s_t = \mathbf{y}_{s_t}^{t+M}(0)$) and the network $\tilde{\varphi}_{t+M}$ (note that $s_t = \tilde{\mathbf{y}}_{s_t}^{t+M}(0)$). Then, we have

$$\frac{\partial J}{\partial \tilde{\mathbf{y}}^t(\tilde{L})} = \frac{\partial J}{\partial \mathbf{y}_{s_t}^{t+M}(0)} + \frac{\partial J}{\partial \tilde{\mathbf{y}}_{s_t}^{t+M}(0)}, \quad t = t^*, \dots, T - M - 1. \quad (8.64)$$

To compute the partial derivative $\frac{\partial J}{\partial \mathbf{y}_t}$ (compare with (8.41)), note that \mathbf{y}_t influences the networks $\tilde{\boldsymbol{\mu}}_t$ in the same way as described for \mathbf{u}_t in deriving (8.63). Now, it is $\mathbf{y}_{y_t}^j(0) = \tilde{\mathbf{y}}_{y_t}^j(0) = \mathbf{y}_t$, $j = t, \dots, t + M - 1$. Therefore, we obtain

$$\begin{aligned}\boldsymbol{\lambda}_t^\top &= \frac{\partial h_t(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t)}{\partial \mathbf{x}_t} + \boldsymbol{\lambda}_{t+1}^\top \frac{\partial f(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t))}{\partial \mathbf{x}_t} \\ &\quad + \left[\sum_{j=t}^{t+M-1} \left(\frac{\partial J}{\partial \mathbf{y}_{y_t}^j(0)} + \frac{\partial J}{\partial \tilde{\mathbf{y}}_{y_t}^j(0)} \right) \right] \frac{\partial g(\mathbf{x}_t, \boldsymbol{\eta}_t)}{\partial \mathbf{x}_t},\end{aligned}\quad (8.65a)$$

$$\boldsymbol{\lambda}_T^\top = \frac{\partial h_T(\mathbf{x}_T)}{\partial \mathbf{x}_T}. \quad (8.65b)$$

As pointed out for (8.63), some terms on the right-hand side of (8.65a) may vanish for some values of $t = 0, 1, \dots, T - 1$. Again, we do not enter into such details to avoid complicating the various relationships. Moreover, the changes one has to make are quite simple.

We have now to determine the partial derivatives $\frac{\partial J}{\partial \mathbf{y}_{u_t}^j(0)}$, $\frac{\partial J}{\partial \mathbf{y}_{y_t}^j(0)}$, $\frac{\partial J}{\partial \mathbf{y}_{s_{t-M}}^t(0)}$, $\frac{\partial J}{\partial \tilde{\mathbf{y}}_{u_t}^j(0)}$, $\frac{\partial J}{\partial \tilde{\mathbf{y}}_{y_t}^j(0)}$, and $\frac{\partial J}{\partial \tilde{\mathbf{y}}_{s_{t-M}}^t(0)}$. These derivatives can be computed in the same way by which we calculated $\frac{\partial J}{\partial \mathbf{y}_{u_t}^j(0)}$ in Sect. 8.4.3. Then, we refer to (8.44).

An observation can be made similar to Remark 8.1. Here again, we note that (8.65) is the adjoint equation of T -stage optimal control theory with the addition of the third term, which takes into account the introduction of the limited-memory neural control law.

The mechanism to derive the neural optimal control law is similar to the one described in Procedure 7.2. Two phases alternate up to a possible convergence. As regards the *forward phase*, we can repeat what we said at the end of Sect. 8.3.3. Of course, we must take into consideration the introduction of the chain of memory networks $\tilde{\boldsymbol{\varphi}}_t$. The *backward phase* is performed as follows. The variables $\delta_q^t(s)$ and $\tilde{\delta}_q^t(s)$ are determined by Backpropagation Equations (8.35) and (8.61), respectively. They are initialized by (8.63) and (8.64), respectively. The vectors $\boldsymbol{\lambda}_t$ are computed by (8.65). $\delta_q^t(s)$ and $\tilde{\delta}_q^t(s)$ enable one to determine the partial derivatives $\frac{\partial J}{\partial w_{pq}^t(s)}$ and $\frac{\partial J}{\partial \tilde{w}_{pq}^t(s)}$, hence the gradient in (8.54). Then, the new weight vector $\mathbf{w}_n(k + 1)$ is obtained by (8.54).

8.5 Approximate Solution of Problem C3-LM by the Certainty Equivalence Principle and the ERIM

The attempts proposed in the literature to solve Problem C3 in approximate way are numerous. Even to give a concise overview of these approximate methods is an extremely difficult task. Among the first attempts, we cite Chap. 7 (Approximations) in [6], that is enlarged and updated in [7]. More recently, just to remain among the most widespread books, [8, Chap. 6] is dedicated to approximate techniques that mitigate the various aspects of the curse of dimensionality in DP that we discussed in Chaps. 6 and 7, and in Sect. 8.2. Other difficulties are dealt with in [8]. It may happen, for example, that “we do not get to know the exact problem to be solved until shortly before the control process begins” or that “the problem data changes as the system is being controlled.” The lack of time to make calculations that cannot be executed before the instant $t = 0$ or when unexpected variations take place in the problem data while the control process is running are among the reasons why the ERIM can be of great usefulness. This was explained in Chap. 7, where Problems C2' and C2'_n were stated.

The mere mention of the suboptimal schemes described in Chap. 6 of [8] is of great interest. The first two are given by the *certainty equivalent control* and the *open-loop feedback control*. The former replaces the random variables of the problem with deterministic quantities, typically with their expected values. The latter ignores future information (for example, at the stage t in Problem C3, the future measures y_{t+1}, \dots, y_{T-1}). Another approximation scheme is the *limited lookahead control law*, in which, at time t , the control u_t is computed on the basis of a truncated number of future stages (for instance, only one). *Rollout algorithms* are other approximate decision schemes with many variants. One of these can be designed as follows. Let us remain in the case of perfect state information (see Problem C2) and consider DP Equation (7.13). Replace this equation with the equation of the one-step lookahead method and write

$$\min_{u_t \in U_t(x_t)} E_{\xi_t} \left\{ h_t(x_t, u_t, \xi_t) + \tilde{J}_{t+1}[f(x_t, u_t, \xi_t)] \right\},$$

where \tilde{J}_{t+1} is the cost-to-go function of some known suboptimal control law $\{\tilde{\mu}_0, \tilde{\mu}_1, \dots, \tilde{\mu}_{T-1}\}$ (possibly obtained heuristically) called *base policy*. Then, the rollout control law is a one-step lookahead control law with \hat{J}_t° approximated by the cost-to-go function of the base policy. For example, we may restrict the controls to few “promising” ones. *Model predictive control* (MPC) draws concepts from the certainty equivalent control, the limited lookahead control, and the rollout algorithms. Special attention is devoted to the presence of the constraints on the control and/or the state whenever the particular nature of the problem requires consideration of the constraints in a “hard way,” therefore avoiding, for example, the use of penalty functions. The method should be dealt with on an infinite-time horizon and we refer the reader to Chap. 10 for the related literature. A not too extensive survey on the vari-

ous methods and, in particular, reporting considerations on the connections between MPC and the rolling algorithms can be found in [9], whereas [11, 15, 22] survey MPC and related closed-loop stability issues.

The ERIM may be useful in some of the methods mentioned previously. This is the case of the receding-horizon control that we shall address in Chap. 10 (we consider this term as strictly related to “limited lookahead control”). In this section, we only take into consideration the approximate solution of Problem C3-LM by the certainty equivalence (CE) principle and the ERIM. Overall, we use three approximations for the solution of Problem C3. The first is the imposition of limited memory. The second is the CE principle, which is the argument of Sect. 8.5.1, where the principle is applied for the solution of Problem C3. The CE principle is specialized in Sect. 8.5.2 to solve Problem C3-LM. In the same section, the ERIM (i.e., the third approximation) is also applied.

8.5.1 Certainty Equivalence Principle

Some optimization problems, which are characterized by the presence of random variables and are hardly solvable analytically or numerically, can be made tractable by replacing the random variables with their estimates. Typically, such estimates are given by mean values. These problems are not necessarily constituted by T -stage optimal control problems like Problems C2 or C3. When we use this approximation, we resort to the CE principle. In general, however, we know that $E[\gamma(\mathbf{x})] \neq \gamma[E(\mathbf{x})]$, where \mathbf{x} is a random variable and $\gamma : \mathbb{R}^{n_a} \rightarrow \mathbb{R}^{n_b}$ is a nonlinear function. Consequently, the solutions are generally suboptimal and other approximations may be better. The application of the CE principle will be addressed in the example of Sect. 8.6.2.

Let us now apply the CE principle to Problem C3, starting from the following demanding hypothesis (see [8, Sect. 6.1]).

Assumption 8.1 An estimator is available that, at the stage t , uses the information vector \mathbf{I}_t to compute a “reasonably accurate” estimate of the state \mathbf{x}_t . We denote this estimate by

$$\hat{\mathbf{x}}_t = \varphi_t^C(\mathbf{I}_t), \quad t = 0, 1, \dots, T - 1. \quad (8.66)$$

The random vector ξ_t is replaced by a “reasonable” estimate. Typically, it should be its expected value, i.e.,

$$\hat{\xi}_t(\mathbf{x}_t, \mathbf{u}_t) = E(\xi_t | \mathbf{x}_t, \mathbf{u}_t), \quad t = 0, 1, \dots, T - 1. \quad (8.67)$$

□

We said that Assumption 8.1 is demanding because the estimate $\hat{\mathbf{x}}_t$ should be determined on the basis of the conditional probability density $p_t(\mathbf{x}_t | \mathbf{I}_t)$. For example, we may let $\hat{\mathbf{x}}_t = E(\mathbf{x}_t | \mathbf{I}_t)$. However, it is the difficulty of computing $p_t(\mathbf{x}_t | \mathbf{I}_t)$ that

really makes Problem C3 intractable. Anyway, let us consider some possible ways to determine $\hat{\mathbf{x}}_t$. Such an estimate may be computed by the *extended Kalman filter* [10, 13] and, when this can be done, by the usual Kalman filter if the state and measurement equations are linear, the random variables are mutually independent and Gaussian, but the cost is not quadratic (in the case of quadratic costs, the LQG assumptions would be verified and Problem C3 can be solved analytically). It can also be computed by other estimators in specific circumstances. Note that, in Problem C2, Assumption 8.1 is unnecessary for applying the CE principle as the value of \mathbf{x}_t is perfectly known.

If Assumption 8.1 is verified, then the CE principle enables one to solve Problem C3 in a simpler suboptimal way. Depending on the nature of the problem and the computational convenience, the CE control functions can be determined either off- or online. Let us see how.

1. Offline computation of the CE control functions

Once Assumption 8.1 has been accepted, Cost (8.5) takes on the form

$$J(\mathbf{x}_0) = \sum_{t=0}^{T-1} h_t[\mathbf{x}_t, \boldsymbol{\mu}_t(\mathbf{x}_t), \hat{\boldsymbol{\xi}}_t(\mathbf{x}_t, \mathbf{u}_t)] + h_T(\mathbf{x}_T). \quad (8.68)$$

Then, we can state the following problem.

Problem 8.1 *Find the control functions*

$$\hat{\mathbf{u}}_t = \hat{\boldsymbol{\mu}}_t(\mathbf{x}_t), \quad t = 0, 1, \dots, T - 1, \quad (8.69)$$

that minimize Cost (8.68) subject to the constraints

$$\begin{aligned} \mathbf{x}_{t+1} &= f_t[\mathbf{x}_t, \boldsymbol{\mu}_t(\mathbf{x}_t), \hat{\boldsymbol{\xi}}_t(\mathbf{x}_t, \mathbf{u}_t)], \\ \boldsymbol{\mu}_t(\mathbf{x}_t) &\in U_t, \end{aligned}$$

for $t = 0, 1, \dots, T - 1$. □

Note that, once \mathbf{x}_t and $\boldsymbol{\xi}_t$ have been replaced by their estimates, Problem 8.1 turns out to be a T -stage deterministic optimal control problem belonging to the class of Problems C1. Then, we can determine the control functions (8.69) by applying the backward phase of DP as described in Chap. 6. When the control process starts, at any stage t the DM acquires the information vector \mathbf{I}_t , computes the estimate $\hat{\mathbf{x}}_t$ by (8.66), and applies the control $\hat{\mathbf{u}}_t = \hat{\boldsymbol{\mu}}_t(\hat{\mathbf{x}}_t)$. In order to point out the dependence of the control $\hat{\mathbf{u}}_t$ on the information vector \mathbf{I}_t , we define the control functions $\hat{\mathbf{u}}_t = \hat{\boldsymbol{\mu}}_t^C(\mathbf{I}_t)$, $t = 0, 1, \dots, T - 1$. Then, the suboptimal solution to Problem C3 that we obtain by solving Problem 8.1 can be written as

$$\hat{\mathbf{u}}_t = \hat{\boldsymbol{\mu}}_t(\hat{\mathbf{x}}_t) = \hat{\boldsymbol{\mu}}_t[\varphi_t^C(\mathbf{I}_t)] = \hat{\boldsymbol{\mu}}_t^C(\mathbf{I}_t), \quad t = 0, 1, \dots, T - 1.$$

For the discretization problems deriving from the application of DP, we refer back to Chap. 6 and to the difficulties described at the end of Sect. 8.2.1.

We point out that, whereas the backward phase of DP is implemented offline in a deterministic way, the stochastic nature of the control process reveals itself online. This is highlighted by the control functions $\hat{\mathbf{u}}_t = \hat{\mu}_t^C(\mathbf{I}_t)$, where the information vectors \mathbf{I}_t are random variables that the DM acquires online.

2. Online computation of the CE control functions

The computations of the suboptimal CE control functions and their application can be performed online according to the following scheme.

Procedure 8.1

- Set $t = 0$.
- (*) According to Assumption 8.1, compute the estimate $\hat{\mathbf{x}}_t = \varphi_t^C(\mathbf{I}_t)$ by (8.66).
- On the basis of Assumption 8.1, replace the random disturbances ξ_t with their estimates $\hat{\xi}_t(\mathbf{x}_t, \mathbf{u}_t), \dots, \hat{\xi}_{T-1}(\mathbf{x}_{T-1}, \mathbf{u}_{T-1})$.
- Solve the following deterministic problem:

Problem 8.2 Find the control vectors $\hat{\mathbf{u}}_{tt}, \hat{\mathbf{u}}_{t,t+1}, \dots, \hat{\mathbf{u}}_{t,T-1}$ that minimize the cost

$$J(\mathbf{x}_t) = \sum_{k=t}^{T-1} h_k[\mathbf{x}_k, \mathbf{u}_k, \hat{\xi}_k(\mathbf{x}_k, \mathbf{u}_k)] + h_T(\mathbf{x}_T)$$

subject to the initial condition $\mathbf{x}_t = \varphi_t^C(\mathbf{I}_t)$ and the constraints

$$\begin{aligned} \mathbf{x}_{k+1} &= f_k[\mathbf{x}_t, \mathbf{u}_k, \hat{\xi}_k(\mathbf{x}_k, \mathbf{u}_k)], \\ \mathbf{u}_k &\in U_k, \end{aligned}$$

for $k = t, t + 1, \dots, T - 1$. □

- Apply the first control of the sequence, that is,

$$\hat{\mathbf{u}}_t = \hat{\mu}_t^C(\mathbf{I}_t).$$

- If $t = T - 1$, then stop, otherwise set $t \leftarrow t + 1$ and go to (*). □

A total of T Problem 8.2 must then be solved *online*.

Summing up, the CE principle approximates Problem C3 by a deterministic Problem C1. Of course, the choice between the two procedures can only be suggested by the nature of the control problem at hand. In particular, the time that the DM has to perform the online computations plays a central role.

8.5.2 Applying the CE Principle to the LM Controller

As pointed out in Remark 8.3, the LM control law is not characterized by the separation property. It can be interesting to examine how the LM control law has to be modified. We recall that enforcing the separation property means splitting the action of the controller into two independent tasks.

1. The estimation function (or the “estimator”) that provides the (accurate in some sense) estimates $\hat{\mathbf{x}}_t$ of \mathbf{x}_t taking into account the past values of the control vector but not the way in which they were generated (i.e., the control functions are ignored).
2. The control functions that provide the controls basing only on the estimates $\hat{\mathbf{x}}_t$, which have to be interpreted as if they were the exact values of \mathbf{x}_t .

It follows that the designs of the estimation and control functions can be implemented independently. Then, we modify Control Functions (8.47) and Memory Functions (8.49) so as to split the action of the control law into two independent tasks. To this end, as reported in [19], we constrain the LM control law to take on the following structure:

1. The sliding window has length $M = 1$, i.e., $\mathbf{I}_t^M = \mathbf{y}_t$.
2. The estimate $\hat{\mathbf{x}}_t$ is computed through the usual prediction/correction recursive mechanism

$$\hat{\mathbf{x}}_t = \hat{\mathbf{x}}_t^p + \varphi_{Kt}(\mathbf{y}_t - \hat{\mathbf{y}}_t^p), \quad t = 0, 1, \dots, T-1, \quad (8.70)$$

where $\hat{\mathbf{x}}_t^p$ and $\hat{\mathbf{y}}_t^p$ are one-step estimated predictions of \mathbf{x}_t and \mathbf{y}_t , respectively, and $\varphi_{Kt} : \mathbb{R}^p \rightarrow \mathbb{R}^d$, $t = 0, 1, \dots, T-1$, are unknown functions to be determined. The application of the CE principle yields

$$\hat{\mathbf{x}}_0^p = E(\mathbf{x}_0) = \boldsymbol{\alpha}, \quad (8.71a)$$

$$\hat{\mathbf{x}}_{t+1}^p = \mathbf{f}_t(\hat{\mathbf{x}}_t, \mathbf{u}_t, \hat{\boldsymbol{\xi}}_t), \quad t = 0, 1, \dots, T-1, \quad (8.71b)$$

$$\hat{\mathbf{y}}_t^p = \mathbf{g}_t(\hat{\mathbf{x}}_t^p, \hat{\boldsymbol{\eta}}_t), \quad t = 0, 1, \dots, T-1, \quad (8.72)$$

where $\hat{\boldsymbol{\xi}}_t = E(\boldsymbol{\xi}_t)$ and $\hat{\boldsymbol{\eta}}_t = E(\boldsymbol{\eta}_t)$ (we assume that $p_t(\boldsymbol{\xi}_t | \mathbf{x}_t, \mathbf{u}_t) = p_t(\boldsymbol{\xi}_t)$).

Now, we write the control and memory functions in the form specified by (8.47) and (8.49) with $M = 1$. We have

$$\mathbf{u}_0 = \boldsymbol{\mu}_0(\mathbf{y}_0), \quad (8.73a)$$

$$\mathbf{u}_t = \boldsymbol{\mu}_t(\mathbf{y}_t, \mathbf{s}_{t-1}), \quad t = 1, \dots, T-1, \quad (8.73b)$$

$$\mathbf{s}_0 = \text{col} (\mathbf{y}_0, \mathbf{u}_0)$$

$$\mathbf{s}_t = \varphi_t(\mathbf{y}_t, \mathbf{u}_t, \mathbf{s}_{t-1}), \quad t = 1, \dots, T-2. \quad (8.74)$$

By enforcing Point 2 of the definition of the separation property, we constrain \mathbf{u}_t to depend on $\hat{\mathbf{x}}_t$, i.e.,

$$\begin{aligned}\mathbf{u}_t &= \bar{\mu}_t(\hat{\mathbf{x}}_t) \\ &= \hat{\mu}_t[\hat{\mathbf{x}}_t^p + \varphi_{Kt}(\mathbf{y}_t - \hat{\mathbf{y}}_t^p)], \quad t = 0, 1, \dots, T-1,\end{aligned}\tag{8.75}$$

or, in more compact form (see (8.72)),

$$\mathbf{u}_t = \boldsymbol{\mu}_t(\mathbf{y}_t, \hat{\mathbf{x}}_t^p), \quad t = 0, 1, \dots, T-1.\tag{8.76}$$

By (8.70), (8.71), and (8.72), we have

$$\begin{aligned}\hat{\mathbf{x}}_0 &= \hat{\mathbf{x}}_0^p + \varphi_{K0}(\mathbf{y}_0 - \hat{\mathbf{y}}_0^p) = \boldsymbol{\alpha} + \varphi_{K0}[\mathbf{y}_0 - \mathbf{g}_0(\boldsymbol{\alpha}, \hat{\boldsymbol{\eta}}_0)]. \\ \hat{\mathbf{x}}_t &= \mathbf{f}_{t-1}(\hat{\mathbf{x}}_{t-1}, \mathbf{u}_{t-1}, \hat{\boldsymbol{\xi}}_{t-1}) + \varphi_{Kt}\{\mathbf{y}_t - \mathbf{g}_t[\mathbf{f}_{t-1}(\hat{\mathbf{x}}_{t-1}, \mathbf{u}_{t-1}, \hat{\boldsymbol{\xi}}_{t-1}), \hat{\boldsymbol{\eta}}_t]\}, \\ &\quad t = 1, \dots, T-1.\end{aligned}$$

The basic difference between the LM controller, constrained to verify the CE principle, and the unconstrained one can be appreciated by comparing (8.73b) with (8.75). Indeed, (8.75) shows the separation property as \mathbf{u}_t only depends on the output $\hat{\mathbf{x}}_t$ of Estimator (8.70). In (8.73) this dependence does not appear. The difference between the two controllers is pointed out in Figs. 8.5 and 8.6.

To clarify the meaning of the memory vector \mathbf{s}_t in the LM controller based on the CE principle, let us rewrite (8.71) as follows:

$$\hat{\mathbf{x}}_{t+1}^p = \mathbf{f}_t \left\{ \hat{\mathbf{x}}_t^p + \varphi_{Kt}[\mathbf{y}_t - \mathbf{g}_t(\hat{\mathbf{x}}_t^p, \hat{\boldsymbol{\eta}}_t)], \mathbf{u}_t, \hat{\boldsymbol{\xi}}_t \right\}, \quad t = 0, 1, \dots, T-1,\tag{8.77}$$

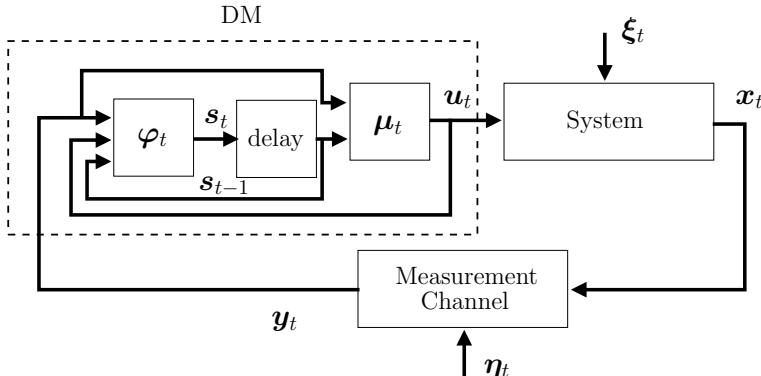


Fig. 8.5 LM controller with $M = 1$ derived from the general LM control scheme shown in Fig. 8.2. The controller is not provided with the separation property since the control function μ_t has the measure y_t as input

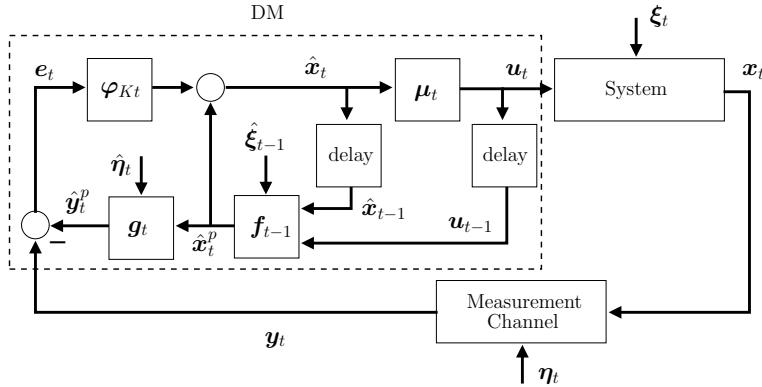


Fig. 8.6 LM controller with $M = 1$. The prediction/correction structure $\hat{x}_t = \hat{x}_t^p + \varphi_{Kt}(y_t - \hat{y}_t^p)$ and the certainty equivalence property is enforced (the control function μ_t has the estimate \hat{x}_t as unique input)

or, in more compact form,

$$\hat{x}_{t+1}^p = \hat{\varphi}_t^p(y_t, u_t, \hat{x}_t^p), \quad t = 0, 1, \dots, T-1. \quad (8.78)$$

Assume now that $\dim(\mathbf{x}_t) = \dim(\mathbf{s}_t)$. Then, by comparing (8.73b) with (8.76), and (8.74) with (8.78), we choose to set $\mathbf{s}_t = \hat{\mathbf{x}}_{t+1}^p$, thus interpreting the memory vector \mathbf{s}_{t-1} as an estimate of the one-step prediction (8.71). From what has already been said, the control law (8.75)–(8.78) cannot behave better than (8.73)–(8.74). However, the separation property may lead to a simpler design of the control law. Moreover, the availability of $\hat{\mathbf{x}}_t$ means the availability of an important information.

Of course, stating that $\hat{\mathbf{x}}_t$ constitutes an estimate of \mathbf{x}_t cannot simply result from the fact that we constrained the control law (8.73)–(8.74) to take on the structure (8.75)–(8.77). In order to be allowed to assert this, we have to design the functions $\varphi_{K0}, \varphi_{K1}, \dots, \varphi_{K,T-1}$, in the sense specified in the “estimation task” of the separation property. This computational task is described below.

1. Design of the control functions

As we said at Point 2 of the description of the separation property, the control functions can be determined as if $\hat{\mathbf{x}}_t$ was the exact value of \mathbf{x}_t . Then, the optimal control law can be derived under the assumption that the exact value of \mathbf{x}_t is always available. It follows that the control functions can be obtained by using the methods for the solution of Problem C2.

2. Design of the estimation functions

We may determine the accuracy of the state estimators by resorting to a standard least-squares criterion. To this end, we define the estimation costs [18]

$$J_0^E \triangleq |\hat{\mathbf{x}}_0 - \boldsymbol{\alpha}|_M^2 + |\mathbf{y}_0 - \mathbf{g}_0(\hat{\mathbf{x}}_0, \hat{\boldsymbol{\eta}}_0)|_{V_0}^2, \quad (8.79a)$$

$$\begin{aligned} J_t^E &\triangleq |\hat{\mathbf{x}}_0 - \boldsymbol{\alpha}|_M^2 + \sum_{i=0}^t |\mathbf{y}_i - \mathbf{g}_i(\hat{\mathbf{x}}_i, \hat{\boldsymbol{\eta}}_i)|_{V_i}^2 \\ &+ \sum_{i=1}^t |\hat{\mathbf{x}}_i - \mathbf{f}_{i-1}(\hat{\mathbf{x}}_{i-1}, \mathbf{u}_{i-1}, \hat{\boldsymbol{\xi}}_{i-1})|_{T_i}^2, \quad t = 1, \dots, T-1, \end{aligned} \quad (8.79b)$$

where M, T_i, V_i are symmetric positive-definite matrices. When applying the ERIM, in (8.79) one can use loss functions different from the quadratic ones.

Optimal state estimators can be obtained by solving the following problems at any stage $t = 0, 1, \dots, T-1$ (see also Sect. 1.5.4).

Problem 8.3 Find the optimal estimators $\hat{\mathbf{x}}_{ti}^\circ = \varphi_{ti}^\circ(\mathbf{I}_t)$, $i = 0, 1, \dots, t$, that minimize Costs (8.79). \triangleleft

At time t , only the estimates $\hat{\mathbf{x}}_{tt}^\circ = \varphi_{tt}^\circ(\mathbf{I}_t)$ have to be used. Clearly, the general assumptions under which Problem C3 (hence Problem 8.3) were stated prevent us from solving the IDO Problem 8.3 analytically. By constraining the estimation functions (8.66) to take on the structure (8.70), let us see whether it is possible to obtain more easily computable estimators, even if suboptimal with respect to the ones derived by solving Problem 8.3. Let us define $\mathbf{e}_t \triangleq \mathbf{y}_t - \hat{\mathbf{y}}_t^p$ and rewrite Costs (8.79) by replacing $\hat{\mathbf{x}}_t$ with $\hat{\mathbf{x}}_t^p + \varphi_{Kt}(\mathbf{e}_t)$:

$$\bar{J}_0^E = |\varphi_{K0}(\mathbf{e}_0)|_M^2 + |\mathbf{y}_0 - \mathbf{g}_0[\boldsymbol{\alpha} + \varphi_{K0}(\mathbf{e}_0), \hat{\boldsymbol{\eta}}_0]|_{V_0}^2 \quad (8.80a)$$

$$\begin{aligned} \bar{J}_t^E &= |\varphi_{K0}(\mathbf{e}_0)|_M^2 + \sum_{i=0}^t |\mathbf{y}_i - \mathbf{g}_i[\hat{\mathbf{x}}_i^p + \varphi_{Ki}(\mathbf{e}_i), \hat{\boldsymbol{\eta}}_i]|_{V_i}^2 \\ &+ \sum_{i=1}^t |\hat{\mathbf{x}}_i^p + \varphi_{Ki}(\mathbf{e}_i) - \mathbf{f}_{i-1}[\hat{\mathbf{x}}_{i-1}^p + \varphi_{K,i-1}(\mathbf{e}_{i-1}), \mathbf{u}_{i-1}, \hat{\boldsymbol{\xi}}_{i-1}]|_{T_i}^2, \\ &t = 1, \dots, T-1. \end{aligned} \quad (8.80b)$$

We can note that the function φ_{Kt}° , which minimizes each cost \bar{J}_t^E , depends on \mathbf{I}_t and the functions $\varphi_{K0}, \varphi_{K1}, \dots, \varphi_{K,t-1}$ through $\hat{\mathbf{x}}_0^p, \dots, \hat{\mathbf{x}}_t^p$. Then, once these functions have been fixed, at time t , the minimization of \bar{J}_t^E with respect to φ_{Kt} can be limited to the determination of a single function instead of all functions φ_{ti} , $i = 0, 1, \dots, t$. However, as in Problem 8.3, we have to solve IDO problems. Because of the generality of the assumptions, these problems cannot be solved in an analytical way. Anyway, the advantage of having to determine a smaller number of estimation functions leads to the following recursive minimization.

Procedure 8.2

- Set $t = 0$.
- Find the optimal value $\varphi_{K0}^\circ(\mathbf{e}_0)$ that minimizes (8.80a).

- Set $t = 1$.

(*) Find the optimal value $\varphi_{Kt}^\circ(\mathbf{e}_t)$ that minimizes

$$\begin{aligned}\mathcal{J}_t^E &= \|\mathbf{y}_t - \mathbf{g}_t[\hat{\mathbf{x}}_t^{po} + \varphi_{Kt}(\mathbf{e}_t), \hat{\boldsymbol{\eta}}_t]\|_{V_t}^2 \\ &\quad + \|\hat{\mathbf{x}}_t^{po} + \varphi_{Kt}(\mathbf{e}_t) - \mathbf{f}_{t-1}[\hat{\mathbf{x}}_{t-1}^{po} + \varphi_{K,t-1}^\circ(\mathbf{e}_{t-1}), \mathbf{u}_{t-1}, \hat{\boldsymbol{\xi}}_{t-1}]\|_{I_t}^2.\end{aligned}\quad (8.81)$$

- If $t = T - 1$, then stop, otherwise set $t \leftarrow t + 1$ and go to (*). \triangleleft

The minimizations in Procedure 8.2 have only a theoretical meaning. Indeed, in order to determine the optimal functions $\varphi_{Kt}^\circ(\mathbf{e}_t)$, we should solve IDO problems. As we said previously, owing to the generality of the assumptions, this is an impossible task. Then, we may resort to the ERIM and constrain the functions $\varphi_{Kt}(\mathbf{e}_t)$ to take on the structure of FSP functions, that is,

$$\tilde{\varphi}_K(\mathbf{e}_t, \mathbf{w}_{tn_t}) = \tilde{\varphi}_K(\mathbf{y}_t - \hat{\mathbf{y}}_t^p, \mathbf{w}_{tn_t}), \quad t = 0, 1, \dots, T - 1. \quad (8.82)$$

By substituting (8.82) into the costs (8.80) and by repeatedly using (8.1), (8.3), and (8.77), we obtain the functions

$$\begin{aligned}\tilde{\mathcal{J}}_0^E(\mathbf{w}_{0n_0}, \mathbf{x}_0, \boldsymbol{\eta}_0) \\ \tilde{\mathcal{J}}_t^E(\mathbf{w}_{0n_0}, \dots, \mathbf{w}_{tn_t}, \mathbf{x}_0, \boldsymbol{\xi}_0^{t-1}, \boldsymbol{\eta}_0^t), \quad t = 1, \dots, T - 1.\end{aligned}$$

Now, Procedure 8.2 can be modified as follows.

Procedure 8.3

- Set $t = 0$.
- Find the optimal vector $\mathbf{w}_{0n_0}^\circ$ that minimizes the expected cost

$$\underset{\mathbf{x}_0, \boldsymbol{\eta}_0}{\text{E}} \tilde{\mathcal{J}}_0^E(\mathbf{w}_{0n_0}, \mathbf{x}_0, \boldsymbol{\eta}_0). \quad (8.83)$$

- Set $t = 1$.

(*) Find the optimal vector $\mathbf{w}_{tn_t}^\circ$ that minimizes the expected cost

$$\underset{\mathbf{x}_0, \boldsymbol{\xi}_0^{t-1}, \boldsymbol{\eta}_0^t}{\text{E}} \tilde{\mathcal{J}}_t^E(\mathbf{w}_{0n_0}, \dots, \mathbf{w}_{tn_t}, \mathbf{x}_0, \boldsymbol{\xi}_0^{t-1}, \boldsymbol{\eta}_0^t). \quad (8.84)$$

- If $t = T - 1$, then stop, otherwise set $t = t + 1$ and go to (*). \triangleleft

As usual, a stochastic gradient algorithm can be used to minimize the expected costs (8.83) and (8.84). We refer the reader to [18] for the computation of the gradient

of Cost Functions (8.83) and (8.84) with respect to the weight parameters in the case of MHL networks. Substitution of (8.82) in (8.70) (with the optimal vectors $\mathbf{w}_{tn_t}^\circ$), gives the approximate estimator

$$\begin{aligned}\hat{\mathbf{x}}_t &= \hat{\mathbf{x}}_t^p + \tilde{\varphi}_K(\mathbf{y}_t - \hat{\mathbf{y}}_t^p, \mathbf{w}_{tn_t}^\circ), \quad t = 0, 1, \dots, T-1, \\ \hat{\mathbf{x}}_0^p &= \boldsymbol{\alpha}.\end{aligned}\quad (8.85)$$

Remark 8.4 Note that Approximate Estimator (8.85) can be used not only in the CE control functions but also as an autonomous device that is not part of a controller. Up to this point, in Chaps. 7 and 8, we described approximate solutions to Problems C2 and C3 by proposing the ERIM as an effective alternative to ADP. We repeatedly stated that the ERIM is a computational tool able to solve general IDO problems in which ADP cannot be applied. The nonlinear state estimation is just one of these problems. Another important problem is the team optimal control problem that will be addressed in the next chapter. As regards the design of approximate state estimators based on the use of FSP functions and in particular on neural networks, we refer the reader to, for instance, [1–5, 18]. \triangleleft

8.6 Numerical Results

In this section, we shall present two examples to show the effectiveness of the ERIM in solving approximately Problems C3-LM. The first example aims at evaluating the ability of optimal LM neural control laws to approximate an optimal classical control law with an unlimited memory. This can be accomplished because the example involves an LQG optimal control problem, which can be solved in closed form. The second example addresses the optimal control of a freeway traffic system already treated in the literature. The example is noteworthy for its engineering importance, the high nonlinearity of the dynamic system, and the large number of state components.

8.6.1 Comparison Between an LQG Optimal Control Law and Limited-Memory Optimal Control Laws

In the present example (see [19]), we compare the performances of some control laws given by (8.50) and (8.51) with the optimal one. In order to make this comparison, we need to compute such an optimal control law. This can be obtained in an analytical form by addressing an LQG optimal control problem. Consider the state equation

$$\mathbf{x}_{t+1} = \begin{bmatrix} 0.65 & -0.19 \\ 0 & 0.83 \end{bmatrix} \mathbf{x}_t + \begin{bmatrix} 7 \\ 7 \end{bmatrix} u_t + \boldsymbol{\xi}_t, \quad t = 0, 1, \dots, T-1,$$

where $T = 10$, $p_0(\mathbf{x}_0) = \mathcal{N}(\mathbf{0}, \Sigma)$ with $\Sigma = \text{diag}(0.01, 0.01)$, and $p_t(\xi_t) = \mathcal{N}(\mathbf{0}, Q)$ with $Q = \text{diag}(0.01, 0.01)$. We assume

$$y_t = [1 \ 1] \mathbf{x}_t + \eta_t, \quad t = 0, 1, \dots, T-1,$$

where $p_t(\eta_t) = \mathcal{N}(0, 0.01)$.

As in Case 2 of Sect. 7.8.1, the state \mathbf{x}_t is required to reach any possible vector \mathbf{x}_T^* belonging to the final set

$$X_T = \{\mathbf{x}_T^* \in \mathbb{R}^2 : 1 \leq x_T^{*1} \leq 3.5, -1.5 \leq x_T^{*2} \leq 1\}.$$

\mathbf{x}_T^* is interpreted as a random vector uniformly distributed on X_T . All the random vectors are mutually independent. Then, Cost Function (8.5) is given by

$$\sum_{t=0}^{T-1} u_t^2 + 40 |\mathbf{x}_T^* - \mathbf{x}_T|^2.$$

Since \mathbf{x}_T^* becomes known to the DM as soon as the decision process begins, we suppose that at stage $t = 0$ the DM has not the time to compute the optimal control law to reach the specific vector \mathbf{x}_T^* it learned. Therefore, such an optimal control law must be computed a priori. By applying the ERIM, we have to optimize the control and memory functions (8.50) and (8.51) which must be slightly modified to take into account the presence of \mathbf{x}_T^* , that is,

$$u_t = \tilde{\mu}_t(\mathbf{I}_t, \mathbf{x}_T^*, \mathbf{w}_{tn_t}), \quad t = 0, 1, \dots, M-1, \quad (8.86a)$$

$$u_t = \tilde{\mu}_t(\mathbf{I}_t^M, s_{t-M}, \mathbf{x}_T^*, \mathbf{w}_{tn_t}), \quad t = M, \dots, T-1, \quad (8.86b)$$

$$s_t = \text{col}(\mathbf{I}_t, \mathbf{u}_t), \quad t = 0, 1, \dots, t^*-1, \quad (8.87a)$$

$$s_t = \tilde{\varphi}_t(\mathbf{I}_t^M, \mathbf{u}_t, s_{t-M}, \mathbf{x}_T^*, \tilde{\mathbf{w}}_{tn_t}), \quad t = t^*, \dots, T-M-1. \quad (8.87b)$$

In accordance with what was observed in Remark 8.3, note that the absence of a separation property in the solution of Problem C3-LM does not authorize us to omit \mathbf{x}_T^* in the memory functions (8.87b).

The optimal parameter vector \mathbf{w}_n° is then obtained by minimizing the expected cost (see (8.53))

$$\hat{J}(\mathbf{w}_n) \triangleq \underset{\mathbf{x}_0, \mathbf{x}_T^*, \xi, \eta}{\mathbb{E}} J(\mathbf{w}_n, \mathbf{x}_0, \mathbf{x}_T^*, \xi, \eta).$$

As usual, the global model complexity n is determined experimentally. The optimal control law is given by

$$u_t^\circ = -L_t^\circ \hat{\mathbf{x}}_t + K_t^\circ \mathbf{x}_T^*, \quad t = 0, 1, \dots, T-1,$$

where L_t° and K_t° are obtained by solving a discrete-time Riccati equation and $\hat{\mathbf{x}}_t \triangleq \mathbb{E}(\mathbf{x}_t | \mathbf{I}_t)$ is the Kalman state estimate.

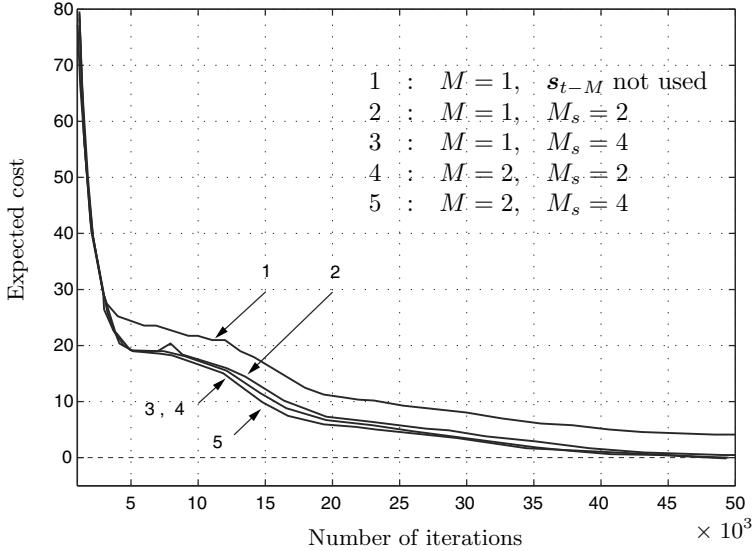


Fig. 8.7 Convergence behaviors of the expected costs of the limited-memory neural control laws.
©1996 IEEE. Reprinted, with permission, from [19]

Functions (8.86) and (8.87b) are implemented by OHL networks with time-invariant numbers of neural units $n = 70$ and $\tilde{n} = 60$, respectively. We set $c_1 = 1$ and $c_2 = 10^6$ in Stochastic Gradient Algorithm (8.29). We considered four different limited-memory control laws, corresponding to $M = 1, 2$ and $M_s = 2, 4$. We also considered a neural control law of the form $u_t = \tilde{\mu}_t^1(\mathbf{I}_t^M, \mathbf{x}_T^*, \mathbf{w}_t)$ (which does not make use of the memory vector s_{t-M}) with $M = 1$.

In Fig. 8.7, we present the behaviors of the expected costs during the learning process and we compare these expected costs with the optimal expected one given by the dashed line. The stochastic gradient algorithm converged after about $5 \cdot 10^4$ iterations. It is interesting to note that the expected costs corresponding to the control laws (8.86) and (8.87) exhibit almost the same good convergence characteristics to the optimal expected cost (equal to ~ 0.192), even for small values of the parameters M and M_s . The convergence behavior of the expected cost corresponding to $\tilde{\mu}_t^1(\mathbf{I}_t^M, \mathbf{x}_T^*, \mathbf{w}_t)$ is much worse, thus revealing the usefulness of the memory vector s_{t-M} .

8.6.2 Freeway Traffic Optimal Control

In this section, a problem of freeway traffic optimal control is faced. A severe traffic congestion must be cleared within a fixed and rather short time. This example (see [19]) is motivated by its engineering importance, by the fact that it deals with a

high-order strongly nonlinear dynamic system, and by the possibility of comparing the approach based on the FSP functions with a technique presented in the literature. We refer to the macroscopic model first proposed by Payne in [21] (see also [17] for a comprehensive description of the model). Macroscopic models represent the behavior of the entire traffic stream of vehicles in terms of aggregate variables such as density and mean speed. These models use continuous variables traditionally describing fluid mechanics. Microscopic models on the other hand represent the behavior of each single vehicle in the traffic stream. Payne's model is based on the discretization (in both space and time) of the equation of the matter conservation, from which the following equations are derived:

$$\begin{aligned} v_{i,t+1} = & v_{it} + \frac{\delta_T}{\tau} \left\{ V_f b_{it} \left[1 - (\rho_{it}/\rho_{\max})^{m(3-2b_{it})} \right]^l - v_{it} \right\} \\ & + \frac{\delta_T}{\Delta_i} v_{it} (v_{i-1,t} - v_{it}) + \frac{\nu \delta_T (\rho_{i+1,t} - \rho_{it})}{\tau \Delta_i (\rho_{it} + \chi)} \\ & - \delta_{\text{on}} \frac{\delta_T}{\Delta_i} v_{it} \frac{r_{it}}{\rho_{it} + \chi}, \quad t = 0, 1, \dots, T-1, \quad i = 1, 2, \dots, D, \end{aligned} \quad (8.88)$$

$$\begin{aligned} \rho_{i,t+1} = & \rho_{it} + \frac{\delta_T}{\Delta_i} [\alpha (1 - \gamma_i) \rho_{i-1,t} v_{i-1,t} \\ & + (1 - 2\alpha + \gamma_i \alpha - \gamma_i) \rho_{it} v_{it} - (1 - \alpha) \rho_{i+1,t} v_{i+1,t} + r_{it}], \\ & t = 0, 1, \dots, T-1, \quad i = 1, 2, \dots, D, \end{aligned} \quad (8.89)$$

where i denotes one of the $D = 30$ sections of length $\Delta_i = 1 \text{ km}$ in which the freeway is divided (see Figs. 8.8 and 8.9), $\delta_T = 15 \text{ s}$ is the sample time interval, $T = 60$, v_{it} is the mean traffic speed on section i at time $t\delta_T$, ρ_{it} is the traffic density, and r_{it} is the on-ramp traffic volume (actually, r_{it} is different from zero only for the sections that contain on/off-ramps). The control horizon lasts 15 min ; as shown later, this time is sufficient to clear severe congestions. Equations (8.88) and (8.89) should be modified (in a simple way) for both the first and the last sections of the considered freeway stretch.

To complete the model, we have to describe the dynamics of the queues on the on-ramps. To this end, we define the variable l_{it} as the number of vehicles in queues on the on-ramp of section $i \in \mathcal{I}^r$ (the indexes of the sections with on/off-ramps constitute the set \mathcal{I}^r), and we define d_{it} as the stochastic demand flow for access to the same ramp. Clearly, these quantities are related by the equation

$$l_{i,t+1} = l_{it} + \delta_T (d_{it} - r_{it}), \quad i = 0, 1, \dots, T-1, \quad i \in \mathcal{I}^r. \quad (8.90)$$

Figure 8.9 shows a freeway section containing on/off-ramps.

The on-ramp traffic volume r_{it} is monitored by traffic lights. b_{it} denotes speed limits set by variable message signs ($b_{it} = 1$ means that there is no speed limit).

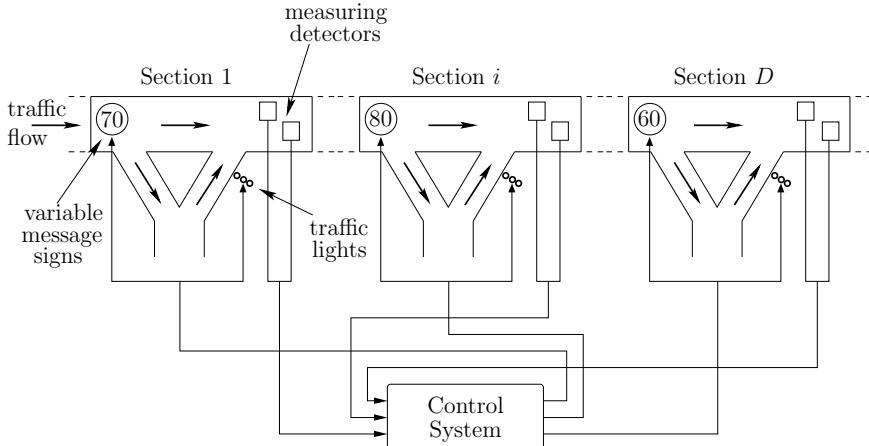


Fig. 8.8 Control system of the freeway traffic

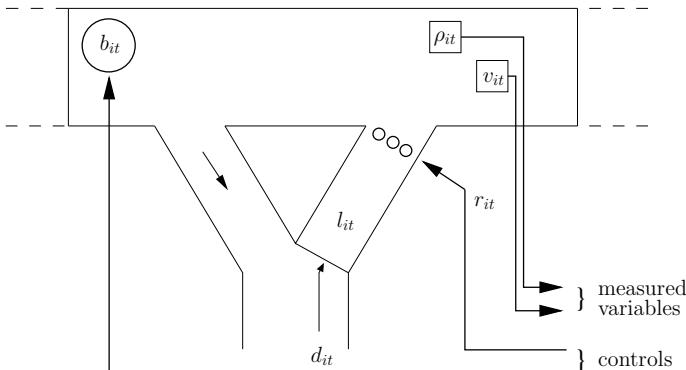


Fig. 8.9 A freeway section i containing on/off-ramps

Then, r_{it} and b_{it} play the role of control variables whereas v_{it} , ρ_{it} , and l_{it} are state variables. We let

$$\mathbf{x}_t \triangleq \text{col} (v_{it}, i = 1, \dots, D, \rho_{it}, i = 1, \dots, D, l_{it}, i \in \mathcal{I}^r),$$

$$\mathbf{u}_t \triangleq \text{col} (r_{it}, i \in \mathcal{I}^r, b_{it}, i = 1, \dots, D),$$

$$\mathbf{d}_t \triangleq \text{col} (d_{it}, i \in \mathcal{I}^r).$$

We assume the demands d_{it} to be mutually independent and uniformly distributed over the subinterval \mathcal{D} of the interval $[0, r_{i \max}]$, where $\mathcal{D} \triangleq [r_{i \max}/2 - \delta, r_{i \max}/2 + \delta]$, with $\delta = 500 \text{ veh/h}$. The value of δ is chosen to produce significant perturbations on the demands without violating the constraints on the on-ramps queues, whatever the queue contents l_{it} may be.

To sum up, the model of the freeway can be described by a discrete-time nonlinear state equation of the form

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t, \mathbf{d}_t), \quad t = 0, 1, \dots, T-1. \quad (8.91)$$

The function \mathbf{f} results from the aggregation of Equations (8.88), (8.89), and (8.90). The various state and control components are subject to the following constraints:

$$0 \leq \rho_{it} \leq \rho_{i\max}, \quad t = 0, 1, \dots, T-1, i = 1, 2, \dots, D, \quad (8.92)$$

$$0 \leq v_{it} \leq v_{i\max}, \quad t = 0, 1, \dots, T-1, i = 1, 2, \dots, D, \quad (8.93)$$

$$0 \leq l_{it} \leq l_{i\max}, \quad t = 0, \dots, T-1, i \in \mathcal{I}_r, \quad (8.94)$$

$$0.7 \leq b_{it} \leq 1, \quad t = 0, 1, \dots, T-1, i = 1, 2, \dots, D, \quad (8.95)$$

$$r_{i\min, t} \leq r_{it} \leq r_{i\max, t}, \quad t = 0, 1, \dots, T-1, i \in \mathcal{I}_r, \quad (8.96)$$

where

$$r_{i\min, t} = \max \left\{ r_{i\min}, d_{it} - \frac{1}{\delta_T} (l_{i\max} - l_{it}) \right\},$$

$$r_{i\max, t} = \min \left\{ r_{i\max}, d_{it} + \frac{1}{\delta_T} l_{it} \right\}.$$

$r_{i\min}$ and $r_{i\max}$ are fixed parameters dependent on the road characteristics.

The model describing the dynamic behavior of the freeway traffic is characterized by the following parameters: $\alpha = 0.8$, $V_f = 123 \text{ km/h}$, $l = 4$, $m = 1.4$, $\tau = 0.01 \text{ h}$, $\nu = 21.6 \text{ km}^2/\text{h}$, $\chi = 20 \text{ veh/km}$, $\delta_{\text{on}} = 0.1$; $\rho_{i\max} = \rho_{\max} = 200 \text{ veh/km}$ and $v_{i\max} = 200 \text{ km/h}$, for $i = 1, \dots, D$; $l_{i\max} = 200 \text{ veh}$, $r_{i\min} = 0 \text{ veh/h}$, and $r_{i\max} = 20000 \text{ veh/h}$, for each on-ramp; $\gamma_i = 0.1$ for each off-ramp (the values of the above scalars were taken from Table 3.1 in [17]). On-ramps and off-ramps are present in sections 3, 9, 15, 21, and 27 of the freeway (these numbers constitute the set \mathcal{I}'). Then, there are five control variables r_{it} , $i \in \mathcal{I}'$. There are five other control variables b_{it} for speed limits, and each acts on six subsequent sections. The variable message signs are placed in sections 1, 7, 13, 19, and 25. It follows that $\dim(\mathbf{x}_i) = 65$ and $\dim(\mathbf{u}_i) = 10$, hence the problem is characterized by a very high-dimensional setting.

All the state components are assumed to be measurable by means of detectors affected by noise. The measurement channel is given by

$$\mathbf{y}_t = \mathbf{x}_t + \boldsymbol{\eta}_t, \quad t = 0, 1, \dots, T-1. \quad (8.97)$$

Each component of $\boldsymbol{\eta}_t$ is uniformly distributed over an interval centered at the mean value of the corresponding state component. Such a mean value is determined on the basis of its temporal profile considered in the noise-free measurement Case 1 described below. The length of each interval is 40% of the range defined by the constraints on the corresponding state component. The initial state \mathbf{x}_0 is assumed

to be uniformly distributed over the set X_0 of states describing all possible initial congestion situations. $\mathbf{x}_0, \mathbf{d}_0, \dots, \mathbf{d}_{T-1}, \boldsymbol{\eta}_0, \dots, \boldsymbol{\eta}_{T-1}$ are assumed to be mutually independent. The components of each vector \mathbf{d}_t and $\boldsymbol{\eta}_t$ are assumed to be mutually independent.

As in Problem P3 stated in [17], the process cost is given by the total time spent on the freeway and in the on-ramps, that is

$$J = \delta_T \sum_{t=0}^{T-1} \left(\sum_{i=1}^D \rho_{it} \Delta_i + \sum_{i \in \mathcal{I}^r} l_{it} \right). \quad (8.98)$$

Since we use the Stochastic Gradient Algorithm (8.29) and, as said after the statement of Problem C3_n, we interpret the constraints in a “soft way,” we resort to a penalty functions approach. As regards State Constraints (8.92), (8.93), and (8.94), we add terms of type (7.93)

$$K_\rho \{ [\max(0, -\rho_{it})]^2 + [\max(0, \rho_{it} - \rho_{i \max})]^2 \},$$

$$K_v \{ [\max(0, -v_{it})]^2 + [\max(0, v_{it} - v_{i \max})]^2 \},$$

and

$$K_l \{ [\max(0, -l_{it})]^2 + [\max(0, l_{it} - l_{i \max})]^2 \},$$

to the cost (8.98), respectively, where K_ρ , K_v , and K_l are positive scalars. Similar terms are added for the control constraints.

In conclusion, we are in the presence of a stochastic optimal control problem that, owing to the presence of a noisy measurement channel on the state, has the characteristics of Problem C3. As we choose to limit the dimension of the information state \mathbf{I}_t and to apply the ERIM, Problem C3 is approximated by a Problem C3_n-LM. As usual, the global model complexity n was determined experimentally. Let us formally state the example of Problem C3_n-LM.

Problem 8.4 *Find the optimal control and memory functions of the forms (8.50) and (8.51) that clear any possible traffic congestion described by the state $\mathbf{x}_0 \in X_0$ by minimizing the expected value of Cost (8.98).* \triangleleft

In order to evaluate the effectiveness of the ERIM, we first compare it with the control scheme proposed in [16] to solve the traffic control problem described previously by assuming that the state vector is measured without noises. This context, in which (8.97) is simply given by $\mathbf{y}_t = \mathbf{x}_t$, is called Case 1. Then, in Case 2, we address Problem 8.4 (see [19] for details).

Case 1. Noise-free measurement channel

Though we consider a situation in which the state vector is perfectly measurable, we do not solve a problem of the C2 type, i.e., an optimal control problem where the initial state \mathbf{x}_0 takes on a fixed value $\hat{\mathbf{x}}$, as we want to remain in the context of Problem 8.4.

This means that we have to state a problem of type C2', in which \mathbf{x}_0 can take any value in the set X_0 of all possible initial states. As we apply the ERIM and not ADP, Problem C2' has to be approximated by Problem C2'_n. Then, we search for a closed-loop optimal control law that: (i) regards \mathbf{x}_0 as a random vector uniformly distributed on X_0 ; and (ii) minimizes the mean value of Cost (8.98) averaged with respect to \mathbf{x}_0 and, obviously, with respect to the disturbance vectors $\mathbf{d}_0, \mathbf{d}_1, \dots, \mathbf{d}_{T-1}$. The optimal control functions must then be determined in the class of FSP functions (7.58) by minimizing the expected cost (7.75), where $\xi \triangleq \text{col}(\mathbf{d}_0, \mathbf{d}_1, \dots, \mathbf{d}_{T-1})$. The FSP optimal control functions take on the forms (7.76), that is,

$$\tilde{\mathbf{u}}_t^\circ = \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t}^\circ), \quad t = 0, 1, \dots, T-1. \quad (8.99)$$

Of course, Cost (7.75) is given by the expected value of Cost (8.98). As explained in the introduction of Chap. 7, the use of Control Law (8.99) is justified if the DM does not have the time to compute online the optimal control vectors for any traffic congestion $\mathbf{x}_0 \in X_0$ that may occur. This happens if such a computing time is not “short” as compared with the sampling time interval. Then, Control Law (8.99) has to be determined offline.

Let us compare the ERIM with the control scheme proposed in [16]. Such a scheme consists of a receding-horizon optimization that is performed *online* periodically. This means that when the controlled plant is in the state \mathbf{x}_t at the stage t , an optimal control problem is solved over a number T' of stages by using an NLP technique, thus deriving a sequence of optimal controls $\mathbf{u}_t^\circ, \dots, \mathbf{u}_{t+T'-1}^\circ$. Then, the first control \mathbf{u}_t° of this sequence is applied. The dynamic system moves to the state \mathbf{x}_{t+1} and the online procedure is repeated. As the control vector \mathbf{u}_t° depends on the current state \mathbf{x}_t , the control law takes on a feedback structure. The possibility of using NLP techniques derives from the fact that the stochastic demand flows d_{it} are replaced by deterministic predicted values, which are supposed to be known. This means that the CE principle is adopted. Returning to the beginning of Sect. 8.5, we can say that *certainty equivalent* and *open-loop feedback controls* (CEOLF control) are applied in a receding-horizon context. Clearly, the CEOLF control proposed in [16] can be implemented provided that the controller’s computing system is sufficiently fast as compared with the freeway system dynamics. We used the penalty functions previously defined in both the CEOLF control scheme and in the ERIM so as to make the two methods comparable. In the CEOLF control we set $T' = 5$. In applying the ERIM, we used the stochastic gradient algorithm (8.29).

The FSP control functions (8.99) are implemented by MHL networks containing two hidden layers composed of 30 and 15 units, respectively. We set $c_1 = 1$, $c_2 = 10^8$, and $\beta = 0.9$. An initial state describing a severe congestion on section 11 is considered. Figure 8.10a, b show the profiles of the variables ρ_{it} and of v_{it} under the action of Control Functions (8.99). When the CEOLF controller is acting, the “surfaces” of these variables (as functions of stage t and section i) are nearly the same, so we do not specify them. The neural controller gives an increase of the cost of about 0.4 %. Similar behaviors can be observed for different traffic congestions $\mathbf{x}_0 \in X_0$.

Case 2. Noisy measurement channel

Let us go back to Problem 8.4, in which all the measures on the state components are affected by noises (see (8.97)). Owing to the rather large number of stages, it is worth resorting to an LM control law, that is, to the functions (8.50) and (8.51). Then, we assign a length $M = 1$ to the “sliding window” of data, which constitute the LM information vector \mathbf{I}_t^M . We also set $M_s = p + m = 65 + 10 = 75$ to define the steady-state dimension of the memory vector s_t . It follows that the control laws (8.50) and (8.51) are given by

$$\begin{aligned}\mathbf{u}_0 &= \tilde{\mu}_0(\mathbf{y}_0, \mathbf{w}_{0n_0}), \\ \mathbf{u}_t &= \tilde{\mu}_t(\mathbf{y}_t, s_{t-1}, \mathbf{w}_{tn_t}), \quad t = 1, 2, \dots, T - 1,\end{aligned}$$

and

$$\begin{aligned}s_0 &= \text{col}(\mathbf{y}_0, \mathbf{u}_0), \\ s_t &= \tilde{\varphi}_t(\mathbf{y}_t, \mathbf{u}_t, s_{t-1}, \tilde{\mathbf{w}}_{t\bar{n}_t}), \quad t = 1, 2, \dots, T - 2.\end{aligned}$$

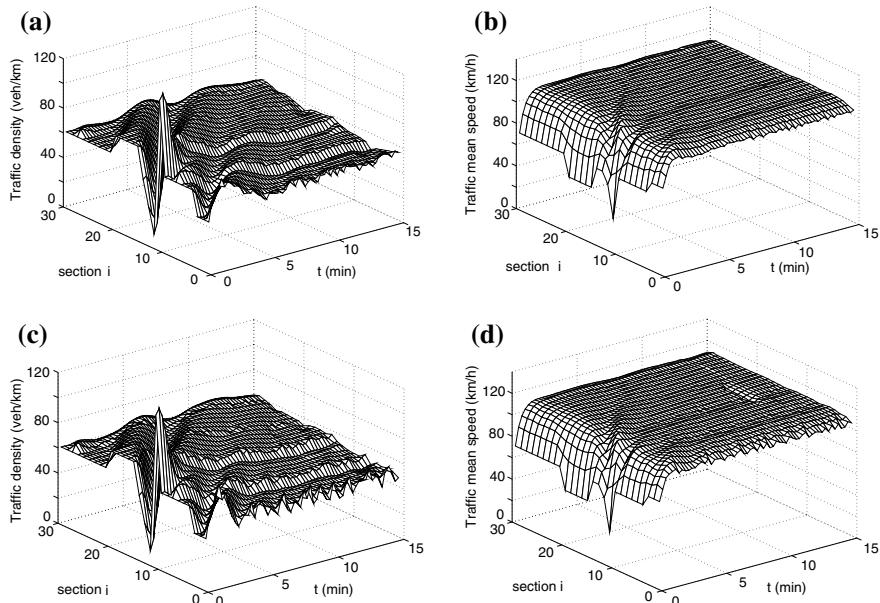


Fig. 8.10 Profiles of the traffic density ρ_{it} and of the mean speed v_{it} in clearing a severe congestion on section 11 under the action of the optimal MHL control law. *Case 1.* **a** and **b** random disturbances on the dynamic system and perfect measurements on the state vector. *Case 2.* **c** and **d** Random disturbances on the dynamic system and noisy measurements on the state vector. ©1996 IEEE. Reprinted, with permission, from [19]

The functions $\tilde{\mu}_t$ are implemented by MHL networks containing two hidden layers, with 30 units in the first layer and 15 units in the second one. The functions $\tilde{\varphi}_t$ are implemented by MHL networks containing two hidden layers, with 30 units in the first layer and 20 units in the second one. Figure 8.10c, d show the profiles of ρ_{it} and v_{it} for the same sequence of random variables d_{it} as in the example shown in Fig. 8.10a, b. Some interesting conclusions can be drawn from Case 1 and Case 2. In both cases, it can be seen that the optimal neural control laws drive the system state to a neighborhood of the traffic configuration (or state vector) characterized by a mean traffic speed that is quite close to the so-called free speed V_f (in our case, $V_f = 123 \text{ km/h}$). V_f is a parameter specific for the freeway and is given by the mean speed corresponding to very low values of traffic density [17]. In a sense, such a result may be interpreted as a stabilizing property of the experimented neural control laws. This property was verified for a variety of initial conditions. The algorithms to optimize the neural control laws converged after about $3 \cdot 10^5$ iterations in Case 1 and after about $4 \cdot 10^5$ iterations in Case 2.

References

1. Alessandri A, Baglietto M, Battistelli G (2008) Moving-horizon state estimation for nonlinear discrete-time systems: new stability results and approximation schemes. *Automatica* 44:1753–1765
2. Alessandri A, Baglietto M, Battistelli G, Gaggero M (2011) Moving-horizon state estimation for nonlinear systems using neural networks. *IEEE Trans. Neural Netw* 22:768–780
3. Alessandri A, Baglietto M, Parisini T, Zoppoli R (1999) A neural state estimator with bounded errors for nonlinear systems. *IEEE Trans Autom Control* 44:2028–2042
4. Alessandri A, Parisini T, Zoppoli R (1997) Neural approximations for nonlinear finite-memory state estimation. *Int J Control* 67:275–302
5. Alessandri A, Parisini T, Zoppoli R (2001) Sliding-window neural state estimation in a power plant heater line. *Int J Adapt Control Signal Process* 15:815–836
6. Aoki M (1967) Optimization of stochastic systems. Academic Press
7. Aoki M (1989) Optimization of stochastic systems. Academic Press, 2nd edn
8. Bertsekas DP (2005) Dynamic programming and optimal control, vol 1. Athena Scientific
9. Bertsekas DP (2005) Dynamic programming and suboptimal control: a survey from ADP to MPC. *Eur J Control* 11:310–334
10. Chui CK, Chen G (2017) Kalman filtering. Springer, 5th edn
11. Garcia CE, Prett DM, Morari M (1989) Model predictive control: theory and practice—a survey. *Automatica* 25:335–348
12. Jazwinski AH (1968) Limited memory optimal filters. *IEEE Trans Autom Control* 13:558–563
13. Jazwinski AH (1970) Stochastic processes and filtering theory. Academic Press
14. Lemon A, Lall S (2016) Approximate sufficient statistics for team decision problems. In: 2016 AAAI spring symposium series. Challenges and opportunities in multiagent learning for the real world
15. Mayne DQ, Rawlings JB, Rao CV, Scokaert POM (2000) Constrained model predictive control: stability and optimality. *Automatica* 36:789–814
16. Messner A, Papageorgiou M (1992) Motorway network control via nonlinear optimization. In: Proceedings of the 1st meeting of the EURO working group on urban traffic and transportation, pp 1–24

17. Papageorgiou M (1983) Applications of automatic control concepts to traffic flow modeling and control. Lecture Notes in Control and Information Sciences. Springer
18. Parisini T, Zoppoli R (1994) Neural networks for nonlinear state estimation. *Int J Robust and Nonlinear Control* 4:231–248
19. Parisini T, Zoppoli R (1996) Neural approximations for multistage optimal control of nonlinear stochastic systems. *IEEE Trans Autom Control* 41:889–895
20. Parisini T, Zoppoli R (1998) Neural approximations for infinite-horizon optimal control of nonlinear stochastic systems. *IEEE Trans Neural Netw* 9:1388–1408
21. Payne HJ (1971) Models of freeway traffic and control. *Simul Council Proc* 1:51–61
22. Rawlings JB, Mayne DQ, Diehl MM (2018) Model predictive control: theory, computation, and design. Nob Hill Publishing

Chapter 9

Team Optimal Control Problems



In economics and engineering, situations frequently occur in which a process is influenced by several *decision-makers* (DMs). Different degrees of cooperation and of distribution of the available information are possible among the decision-makers. The decentralization of the information can be due to several reasons. For example, there may be physical constraints on the communication links: this happens when different DMs can access local variables and an instantaneous exchange of measured data among the DMs is impossible.

Obviously, the performance of a unique central DM, making decisions on the basis of all the information acquired by the measurement devices, is usually better than that of a set of DMs each provided with local information. In general, it is very probable that some cost should be paid to obtain an ideal (or at least a quasi-ideal) centralization of information. A trade-off must be sought between lower performance owing to the decentralization and possible gains in system robustness and/or economic saving.

In this chapter, we consider the case where various DMs share different information but they take decisions aimed at accomplishing a common goal, i.e., minimizing the same cost functional. Such an organization can be mathematically described within the framework of Marschak and Radner's *team theory* [22, 23, 29] (see also [18]). An example of a team optimal control problem was given in Sect. 1.5.5.

All the a priori information is assumed to be shared by the DMs. This information is given by the cost functional, the probability densities of the random variables, the mathematical models of the DMs' observation channels and, in the dynamic case, of the system controlled in common. Under these hypotheses, the computation of the optimal control laws of the DMs can be performed either in a decentralized way by any DM or by a single, centralized unit.

Typical examples of team organizations can be encountered in communication and computer networks extending over large geographical areas where several routing nodes cooperate on the overall performance, in large-scale traffic systems when

a metropolitan area is divided into relatively independent sectors, in large-scale freeway systems, in production plants in which mobile robots moving semifinished products have to be coordinated, in geographically distributed systems for the production of electrical energy, etc.

After the pioneering results by Marschak and Radner, in the 70s of last century major improvements were obtained by Ho and Chu [14], who extended closed-form solutions from *static* to *dynamic* teams. In the former, the DMs make decisions independently of one another. In the latter, the decisions of the DMs influence the information of other DMs. All the results available about closed-form optimal solutions, however, require quite strong assumptions about the *information structure of the team*, i.e., the way each DM's information set is influenced by the stochastic environment and by the decisions made by the other DMs. The main results are based on the fact that the *LQG hypotheses* hold and on the concept of a *partially nested information structure*. A general approach to the solution of a team optimal decision problem, however, has not yet been presented in the literature.

In this chapter, we give up on seeking after optimal solutions to a general team optimal control problem and, once again, we resort to the extended Ritz method (ERIM) by constraining the control functions to take on the form of fixed-structure parametrized (FSP) functions with “free” parameters to be optimized.

Basic concepts of team theory are presented in Sect. 9.1. In Sect. 9.2, we focus on team problems with known optimal solutions, i.e., LQG static teams and LQG dynamic teams with partially nested information structures. The optimal decentralized control problem (Problem DC) is stated in Sect. 9.3, where its approximate solution by the ERIM is addressed, too. Then, the ERIM is tested on two case studies. The first, presented in Sect. 9.4, deals with the well-known *Witsenhausen counterexample*, whose optimal solution has not yet been found (but it has been demonstrated that it exists). In Sect. 9.5, we shall consider *dynamic routing* in communication networks. A nonlinear discrete-time dynamic model is given for a *store-and-forward packet-switching network* in which the routing nodes play the role of cooperating DMs of a team. The resulting problem does not verify either the LQG hypotheses or the partially nestedness assumption on the information structure.

9.1 Basic Concepts of Team Theory

Going back to what we said in Sect. 1.5.5, let us consider a set $\{DM_1, \dots, DM_M\}$ of M DMs (or agents). Each of them, on the basis of its own *information vector* $\mathbf{I}_i \in \mathbb{R}^{q_i}$, must make its decisions $\mathbf{u}_i \in \mathbb{R}^{m_i}$. The information vector \mathbf{I}_i of DM_i includes all the information useful for making decisions, that is, information on the decisions made by the other agents and on a vector $\mathbf{z} \in \mathbb{R}^r$ that describes a stochastic environment, i.e., the uncertainties in the external world that are not influenced by any DM. The random vector \mathbf{z} is characterized by a probability density function $p(\mathbf{z})$. Then, each information vector can be described by the function

$$\mathbf{I}_i = \mathbf{g}_i(\mathbf{u}_1, \dots, \mathbf{u}_{i-1}, \mathbf{u}_{i+1}, \dots, \mathbf{u}_M, \mathbf{z}), \quad i = 1, \dots, M. \quad (9.1)$$

The functions \mathbf{g}_i and the connections among them play a central role in team theory. The following definition is useful.

Definition 9.1 The set of functions $\mathbf{g} \triangleq \{\mathbf{g}_1, \dots, \mathbf{g}_M\}$ is called the “information structure” of the team. \triangleleft

In Sect. 9.1.1, the team optimal control problem is stated. Team information structures are detailed in Sect. 9.1.2, where we introduce the important concept of a partially nested information structure. In Sect. 9.1.3, the “person-by-person optimality” is stated, which is a necessary condition for being an optimal solution of a team optimal control problem.

9.1.1 Statement of the Team Optimal Control Problem

A *causality condition* in the teams has to be verified. This means that, if the control action \mathbf{u}_i of DM_i affects the information vector \mathbf{I}_j of DM_j , then \mathbf{u}_j does not affect \mathbf{I}_i . By “affecting” we mean that the decisions of a DM modify the information vector of another. We shall further explain this concept later. The following definition distinguishes two basically different behaviors in a team.

Definition 9.2 A team is said to be “static” if the functions \mathbf{g}_i , $i = 1, \dots, M$, are independent of the DMs’ control actions, i.e.,

$$\mathbf{I}_i = \mathbf{g}_i(\mathbf{z}).$$

A team which is not static is said to be “dynamic.” \triangleleft

The following definition drastically simplifies how to obtain DMs’ optimal decision functions in a closed form.

Definition 9.3 An information structure is said to be “linear” if the information vectors of each agent DM_i are given by a linear combination of the stochastic vector \mathbf{z} and of the control vectors \mathbf{u}_j of the other agents DM_j , that is,

$$\mathbf{I}_i = H_i \mathbf{z} + \sum_{j \neq i} D_{ij} \mathbf{u}_j, \quad i = 1, \dots, M, \quad (9.2)$$

where H_i and D_{ij} are matrices of appropriate dimensions. \triangleleft

In a linear information structure, the causality condition imposes the following constraint on (9.2):

$$D_{ij} \neq 0 \implies D_{ji} = 0, \quad \forall i, j, i \neq j. \quad (9.3)$$

On the basis of what we previously said, the decision or control function of each decision-maker DM_i takes on the form

$$\boldsymbol{u}_i = \boldsymbol{\mu}_i(\boldsymbol{I}_i), \quad i = 1, \dots, M. \quad (9.4)$$

We assume $\boldsymbol{\mu}_i \in \mathcal{S}_i$, where \mathcal{S}_i is the set of all admissible control functions for the decision-maker DM_i . The decision or control law of the team is defined as

$$\boldsymbol{\mu} \triangleq \text{col}(\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_M). \quad (9.5)$$

The agents DM_1, \dots, DM_M cooperate on the minimization of the mean value of the common cost function

$$J = J(\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_M, \boldsymbol{z}). \quad (9.6)$$

Hence, we have a *team optimal control problem*. Connections with game theory can be found in [3].

As we said previously, we assume that all the possible a priori information (i.e., $p(\boldsymbol{z}), \boldsymbol{g}, J, \mathcal{S}_1, \dots, \mathcal{S}_M$) is known to all the DMs. The following problem can be stated.

Problem T. *Find the optimal control functions $\boldsymbol{\mu}_1^\circ(\boldsymbol{I}_1), \dots, \boldsymbol{\mu}_M^\circ(\boldsymbol{I}_M)$ that minimize the expected value of Cost (9.6), i.e.,*

$$\underset{\boldsymbol{z}}{\mathbb{E}} J[\boldsymbol{\mu}_1(\boldsymbol{I}_1), \dots, \boldsymbol{\mu}_M(\boldsymbol{I}_M), \boldsymbol{z}], \quad (9.7)$$

for all $\boldsymbol{\mu}_i \in \mathcal{S}_i$. \triangleleft

As pointed out in Sect. 1.5.5 for static teams (see Problem 1.8), Problem T takes on the form of Problem PM.

Remark 9.1 As the DMs share all the a priori information, each of them is able to solve Problem T. Thus, it can determine not only its own optimal control function but also those of the other DMs. From the *computational point of view*, Problem T can then be solved in both a *centralized* and a *decentralized* way. At run time, each DM has only access to its own information vector and, on the basis of it, it decides its control action by using its control function. Then, from an *online decisional point of view*, the overall team control law is decentralized. \triangleleft

9.1.2 Partially Nested Information Structures

We follow the framework presented by Ho and Chu [14, 16] to give some definitions that are necessary to characterize the properties of a team information structure. Some concepts are taken from [38] to deal with nonlinear information structures.



Fig. 9.1 Precedence diagram of a static team

Definition 9.4 Given a realization of the random vector z , DM_i is said to “affect” DM_j for this particular value of z (and we write $i R^z j$) if there exist two control vectors, $\mathbf{u}^a \triangleq \text{col}(\mathbf{u}_1, \dots, \mathbf{u}_i^a, \dots, \mathbf{u}_M)$ and $\mathbf{u}^b \triangleq \text{col}(\mathbf{u}_1, \dots, \mathbf{u}_i^b, \dots, \mathbf{u}_M)$, such that

$$\mathbf{g}_j(\mathbf{u}^a, z) \neq \mathbf{g}_j(\mathbf{u}^b, z).$$

△

Definition 9.4 refers to a situation in which DM_i may affect DM_j for certain values of z but DM_j may affect DM_i for other values of z . By “affect” we mean that a DM can modify the information vector of another by its decisions.

We want to address teams in which the concept of “affecting” is independent of z . To this end, we give the following definition.

Definition 9.5 We say that DM_i is a “direct precedent” of DM_j (and we write $i R j$) if $i R^z j$ is verified for any value of z without the possibility that there may be values of z for which $j R^z i$. △

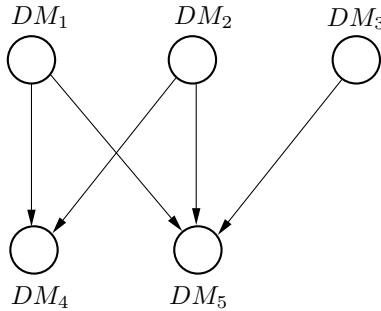
The abovementioned definition enables us to extend the causality condition that was introduced and formally stated in (9.3) from linear information structures to nonlinear ones. As we are only interested in teams that verify the causality condition, we fix a unidirectional “flow of affecting information” inside the team. Therefore, it is necessary to establish an order among the DMs. The following definition aims at this objective.

Definition 9.6 We say that DM_i is a “precedent” of DM_j (and we draw an arc from DM_i to DM_j , i.e., $DM_i \rightarrow DM_j$) if a) $i R j$ or b) there exist distinct $r, s, \dots, t \in \{1, \dots, M\}$ such that $i R r, r R s, \dots, t R j$ and there do not exist $r', s', \dots, t' \in \{1, \dots, M\}$ such that $j R r', r' R s', \dots, t' R i$. △

The precedence relations can be graphically depicted by the so-called *precedence diagram*. The DMs are represented as nodes of a graph and a direct solid link is drawn from DM_i to DM_j if $i R j$. If $DM_i \rightarrow DM_j$ through other agents, the precedence diagram is depicted by a path of directed links joining DM_i with DM_j through the nodes associated with these agents. For a static team, the precedence diagram consists only of isolated nodes, as shown in Fig. 9.1. An example of precedence diagram of a dynamic team is shown in Fig. 9.2.

Having clarified that the concept of “precedence” means that a decision-maker DM_i can modify the information vector I_j of another DM_j , it is now important to establish under which conditions a given DM can incorporate the information vector of another one. Then, we give the following definition.

Fig. 9.2 Precedence diagram of a dynamic team



Definition 9.7 The DM_i 's information vector \mathbf{I}_i is said to be “included” in the DM_j 's information vector \mathbf{I}_j (we write $DM_i \succ DM_j$ and we draw a dashed arc from DM_i to DM_j in the precedence diagram), if there exists a function f_{ji}^μ (possibly dependent on μ) such that, for any z ,

$$\mathbf{I}_i = f_{ji}^\mu(\mathbf{I}_j).$$

△

Therefore, like the precedence relation, the inclusion relation can also be represented graphically. The dashed arcs constitute a *memory-communication graph*. This term is motivated as follows. If in the team there is a player acting at different subsequent stages, this player gives rise to different DMs. Each DM has a perfect memory, i.e., it remembers the information vectors of the DMs that acted before. We agree that, since the DM_j includes the information vector of DM_{j-1} , it is not necessary to draw the dashed arcs from DM_{j-2} (and the other previous ones) to DM_j . This situation is depicted in Fig. 9.3 and corresponds to State Equations (8.3) and Observation Channel (8.1). Moreover, if in the team there are different players, the information vector of one of them may be “communicated” to another player.

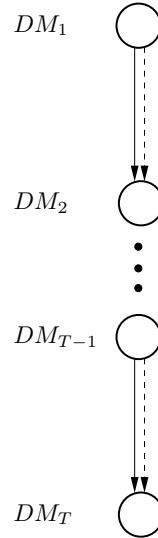
The following definition is useful to describe the information structure of a team graphically, as it connects the concepts of “precedence” and “memory-communication.”

Definition 9.8 The precedence diagram and the memory-communication diagram constitute the “information structure diagram or graph” of the team.

△

As will be clear in the following sections, the information structure of a team organization is very important to establish if a team problem can admit an easily computable optimal solution. A fundamental consideration is whether or not the information structure of a team is *partially nested*. We say that a team is provided with a *partially nested information structure* if each DM can reconstruct the information vectors of the DMs that affected its own information vector. This situation occurs

Fig. 9.3 Information structure of the team of DMs corresponding to the controller of Problem C3.
 ©1972 IEEE. Adapted, with permission, from [14]



when the precedence diagram and the memory-communication diagram coincide or the latter contains additional arcs. Let us formalize this concept.

Definition 9.9 We say that an information structure is “partially nested” if

$$DM_i \rightarrow DM_j \implies DM_i \succ DM_j.$$

△

9.1.3 Person-by-Person Optimality

Since the determination of the team optimal control functions (9.4) may be a difficult (and sometimes impossible) task, a simplified version of Problem T is addressed here. Consider Expected Cost (9.7) and let

$$\hat{J}[\mu_1(\mathbf{I}_1), \dots, \mu_M(\mathbf{I}_M)] \triangleq \underset{z}{\mathbb{E}} J[\mu_1(\mathbf{I}_1), \dots, \mu_M(\mathbf{I}_M), z].$$

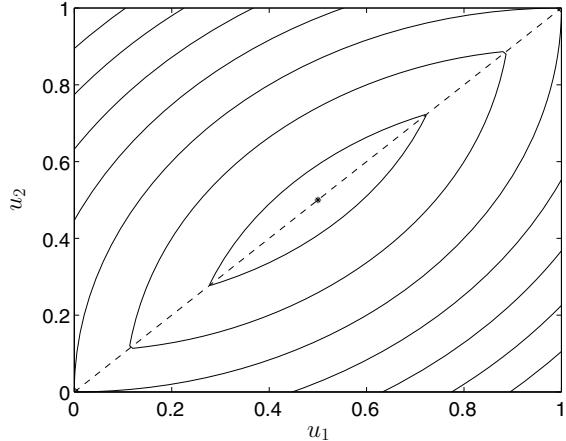
Then, the following problem is stated.

Problem PBP. For $i = 1, \dots, M$, suppose that the control functions $\mu_1^*, \dots, \mu_{i-1}^*, \mu_{i+1}^*, \dots, \mu_M^*$ have been fixed. Find the optimal control function $\mu_i^* \in \mathcal{S}_i$ that minimizes

$$\hat{J}[\mu_1^*, \dots, \mu_{i-1}^*, \mu_i, \mu_{i+1}^*, \dots, \mu_M^*].$$

△

Fig. 9.4 Isocost curves for Example 9.1



The functions μ_1^*, \dots, μ_M^* are called *person-by-person optimal control functions* [29]. Problem PBP obviously establishes a necessary but not sufficient condition for being an optimal solution to Problem T.

The following theorem [29] points out the importance of the concept of person-by-person optimality.

Theorem 9.1 Suppose that the vectors $\mathbf{u}_1, \dots, \mathbf{u}_M$ can take their values from $\mathbb{R}^{m_1}, \dots, \mathbb{R}^{m_M}$, and that, for any \mathbf{z} , Cost Function (9.6) is convex and differentiable with respect to $\mathbf{u}_1, \dots, \mathbf{u}_M$. Then, any person-by-person optimal law is also an optimal law. Furthermore, if Cost Function (9.6) is strictly convex for any \mathbf{z} , then the team optimal law, if any, is unique. \square

The differentiability of the cost function (9.6) plays an essential role in the proof of Theorem 9.1, as shown by the following example [29].

Example 9.1 Let us consider a team composed of two DMs. The cost function is independent of \mathbf{z} and is given by

$$J(\mu_1, \mu_2) = \max[u_1^2 + (u_2 - 1)^2, (u_1 - 1)^2 + u_2^2].$$

In Fig. 9.4, isocost curves are depicted. J is strictly convex but not differentiable. It is easy to show that any pair $u_1 = u_2$ corresponds to a person-by-person optimal solution, but only $u_1 = u_2 = 1/2$ is the optimal control law. \triangleleft

When the information structure of a team is linear, the cost function is quadratic, and the stochastic vector \mathbf{z} has a Gaussian probability density function (i.e., when the LQG hypotheses hold; see Sect. 9.2.1), Theorem 9.1 is an operational tool to find an optimal solution.

9.2 Team Problems with Known Optimal Solution

In this section, we address some of the few team optimal control problems whose optimal solution is known. In Sect. 9.2.1, we show the optimal solution derived by Radner [29] for LQG static teams. The reduction of LQG dynamic teams to static ones, under the assumption of *partially nested information structures*, is considered in Sect. 9.2.2. In Sect. 9.2.3, we describe the concept of sequential partitioning of a team, which is useful for the next section. The control process given by a dynamic system, on which several controllers exert their actions – each provided with different measures on the system state – is examined in Sect. 9.3. This team organization is commonly called *optimal decentralized control* and is taken into consideration under different assumptions on the exchange of information among the controllers.

9.2.1 LQG Static Teams

Let us now consider a static LQG problem. This means that Information Structure (9.2) takes on the form:

$$\mathbf{I}_i = H_i z, \quad i = 1, \dots, M, \quad (9.8)$$

where H_i , $i = 1, \dots, M$, are $q_i \times r$ full rank matrices (with $r \geq q_i$). Let $p(z) = \mathcal{N}(\hat{z}, Z)$ with $Z = Z^\top > 0$. Assume the cost (9.6) to be quadratic, that is,

$$J = \mathbf{u}^\top Q \mathbf{u} + 2\mathbf{u}^\top S z, \quad (9.9)$$

where $Q = Q^\top > 0$ and S are matrices of appropriate dimensions. Problem PBP becomes

$$\begin{aligned} \min_{\mathbf{u}_i} \hat{J}_i(\mathbf{u}_i, \mathbf{I}_i) &= \min_{\mathbf{u}_i} \mathbb{E} \left[\sum_{r, j \neq i} \boldsymbol{\mu}_r^{*\top} Q_{rj} \boldsymbol{\mu}_j^* + 2 \sum_{j \neq i} \mathbf{u}_i^\top Q_{ij} \boldsymbol{\mu}_j^* \right. \\ &\quad \left. + \mathbf{u}_i^\top Q_{ii} \mathbf{u}_i + 2 \sum_{j \neq i} \boldsymbol{\mu}_j^{*\prime} S_j z + 2 \mathbf{u}_i^\top S_i z \mid \mathbf{I}_i \right], \quad i = 1, \dots, M, \end{aligned}$$

where Q_{ij} and S_i are blocks of appropriate dimensions of Q and S . As we are dealing with a static team, the decision \mathbf{u}_i cannot influence the control laws $\boldsymbol{\mu}_j$, $j \neq i$. The following necessary conditions are obtained by setting to zero the partial derivatives of the cost \hat{J} with respect to the components of \mathbf{u}_i :

$$Q_{ii} \boldsymbol{\mu}_i^*(\mathbf{I}_i) + \sum_{j \neq i} Q_{ij} \mathbb{E} [\boldsymbol{\mu}_j^*(\mathbf{I}_j) \mid \mathbf{I}_i] + S_i \mathbb{E} (z \mid \mathbf{I}_i) = 0, \quad \forall i, \forall \mathbf{I}_i. \quad (9.10)$$

Note that the hypotheses of Theorem 9.1 apply to Cost (9.9). Any person-by-person optimal control law is then optimal. Moreover, if the cost is strictly convex for any \mathbf{z} , then the optimal control law is unique. Since any solution of the necessary conditions (9.10) is (by uniqueness) the optimal solution, we consider control laws of the form

$$\mathbf{u}_i = \boldsymbol{\mu}_i(\mathbf{I}_i) = A_i \mathbf{I}_i + \mathbf{b}_i, \quad i = 1, \dots, M, \quad (9.11)$$

where A_i and \mathbf{b}_i are unknown matrices and vectors of dimensions $m_i \times q_i$ and m_i , respectively. Substituting (9.11) into (9.10) gives

$$Q_{ii} (A_i \mathbf{I}_i + \mathbf{b}_i) + \sum_{j \neq i} Q_{ij} E(A_j H_j \mathbf{z} + \mathbf{b}_j | \mathbf{I}_i) + S_i E(\mathbf{z} | \mathbf{I}_i) = 0$$

and hence

$$Q_{ii} (A_i \mathbf{I}_i + \mathbf{b}_i) + \left(\sum_{j \neq i} Q_{ij} A_j H_j + S_i \right) E(\mathbf{z} | \mathbf{I}_i) + \sum_{j \neq i} Q_{ij} b_j = 0, \\ i = 1, \dots, M, \forall \mathbf{I}_i. \quad (9.12)$$

From known properties of Gaussian random variables, we have

$$E(\mathbf{z} | \mathbf{I}_i) = \hat{\mathbf{z}} + K_i(\mathbf{I}_i - H_i \hat{\mathbf{z}}) \\ = K_i \mathbf{I}_i + (I - K_i H_i) \hat{\mathbf{z}}, \quad i = 1, \dots, M, \quad (9.13)$$

where $K_i = Z H_i^\top (H_i Z H_i^\top)^{-1}$ is invertible on the basis of the previous hypotheses. By substituting (9.13) into (9.12), we get

$$\left[Q_{ii} A_i + \left(\sum_{j \neq i} Q_{ij} A_j H_j + S_i \right) K_i \right] \mathbf{I}_i \\ + \sum_{j=1}^N Q_{ij} \mathbf{b}_j + \left(\sum_{j \neq i} Q_{ij} A_j H_j + S_i \right) (I - K_i H_i) \hat{\mathbf{z}} = \mathbf{0}, \\ i = 1, \dots, M. \quad (9.14)$$

As Eq. (9.14) must be true for any possible value of \mathbf{I}_i , we have

$$\sum_{j=1}^M Q_{ij} A_j H_j Z H_i' = -S_i Z H_i^\top, \quad i = 1, \dots, M, \quad (9.15)$$

and

$$\sum_{j=1}^M Q_{ij} \mathbf{b}_j = -\left(\sum_{j \neq i} Q_{ij} A_j H_j + S_i \right) (I - K_i H_i) \hat{\mathbf{z}}, \quad i = 1, \dots, M. \quad (9.16)$$

It can be proved [29] that the coefficients that multiply the matrices A_i in Algebraic System (9.15) form a symmetric positive-definite matrix. As we also have $Q = Q^\top > 0$, (9.15) and (9.16) admit a solution. Then, Control Law (9.11) does exist. The following theorem [29] summarizes the previous results.

Theorem 9.2 *A team optimal problem with Cost Function (9.9) and Information Structure (9.8) has a unique optimal solution of the form (9.11), where the matrices A_i and the vectors b_i can be found by solving Algebraic Systems (9.15) and (9.16).* \square

In next section, we shall show that, under suitable assumptions, the results of Theorem 9.2 can be generalized to the case of dynamic teams.

9.2.2 LQG Dynamic Teams with Partially Nested Information Structure

As in the previous section, we consider team organizations with quadratic cost functions, a stochastic environment described by a random vector z with probability density $p(z) = N(\hat{z}, Z)$ with $Z = Z^\top > 0$, and the linear but dynamic, information structure (compare with (9.2))

$$\mathbf{I}_i = H_i z + \sum_{DM_j \rightarrow DM_i} D_{ji} \mathbf{u}_j, \quad i = 1, \dots, M, \quad (9.17)$$

where H_i and D_{ji} are matrices with appropriate dimensions. Let us also suppose that the dynamic team is characterized by a partially nested information structure. Under this hypothesis, the terms related to the preceding DMs (i.e., the terms in Summations (9.17)) are “redundant” and so can be eliminated. All this is proved in [14] and the following result holds true.

Theorem 9.3 *In a dynamic team with a linear and partially nested information structure, the dynamic information structure (9.17) is equivalent to an information structure in static form*

$$\hat{\mathbf{I}}_i = H_i z, \quad i = 1, \dots, M, \quad (9.18)$$

for any fixed set of control laws. \square

Remark 9.2 Of course, the possibility of also reducing nonlinear partially nested information structures to static ones is an important property. In [15], it is shown that the linearity of the information structure in Theorem 9.3 is a sufficient but not necessary condition for this possibility. Under suitable conditions, when it is possible to reconstruct $\hat{\mathbf{I}}_i$ from \mathbf{I}_i and vice versa, the result of Theorem 9.3 can be generalized to the case of nonlinear information structures. \triangleleft

Under LQG assumptions, the following theorem [14] extends the results of Theorem 9.2 from static to dynamic teams.

Theorem 9.4 *In a dynamic team with a partially nested information structure, whose matrices H_i and Q satisfy the same properties as in the static case, the optimal control law exists and is unique and linear with respect to the information vectors \mathbf{I}_i .* \square

9.2.3 Sequential Partitions

In this section, we consider the possibility of dividing a team optimal problem into simpler independent subproblems. Following the definitions given in [38], denote the team by $\mathcal{G} \triangleq \{DM_1, \dots, DM_M\}$ and decompose the set \mathcal{G} into the subsets $\mathcal{G}_1, \dots, \mathcal{G}_K$ so that the DMs belonging to \mathcal{G}_1 precede those belonging to \mathcal{G}_2 , the DMs belonging to \mathcal{G}_2 precede those belonging to \mathcal{G}_3 , and so on. The following definitions formalize and extend this decomposition to a more general case.

Definition 9.10 $\{\mathcal{G}_1, \dots, \mathcal{G}_K\}$ is a “partition” of the set \mathcal{G} if the following conditions are satisfied:

$$\begin{aligned} \bigcup_{i=1}^K \mathcal{G}_i &= \mathcal{G}, \\ \mathcal{G}_i \cap \mathcal{G}_j &= \emptyset, \quad i, j = 1, \dots, K, \quad i \neq j. \end{aligned}$$

\triangleleft

Definition 9.11 Given two subsets, $\mathcal{G}_i, \mathcal{G}_j \subset \mathcal{G}$, $\mathcal{G}_i \cap \mathcal{G}_j = \emptyset$, we say that \mathcal{G}_j does not precede \mathcal{G}_i (and we write $\mathcal{G}_j \not\rightarrow \mathcal{G}_i$), if DM'_j is not a precedent of DM'_i for any $DM'_i \in \mathcal{G}_i$ and any $DM'_j \in \mathcal{G}_j$. \triangleleft

Also taking into account the memory-communication relationships, we can give the following further definition.

Definition 9.12 A partition $\{\mathcal{G}_1, \dots, \mathcal{G}_K\}$ of \mathcal{G} is a “sequential partition” if:

- (a) $\mathcal{G}_j \not\rightarrow \mathcal{G}_i$, $j = i + 1, \dots, K$,
- (b) $\mathcal{G}_i \succ \mathcal{G}_{i+1}$, $i = 1, \dots, K - 1$,

where $\mathcal{G}_i \succ \mathcal{G}_{i+1}$ means that each DM belonging to \mathcal{G}_{i+1} can reconstruct the information vector of each DM belonging to \mathcal{G}_i . \triangleleft

Suppose that Cost Function (9.6) takes on the additive structure

$$J = \sum_{i=1}^K w_i(\mu_{\mathcal{G}_1}, \dots, \mu_{\mathcal{G}_i}, z), \quad (9.19)$$

where $\mu_{\mathcal{G}_i} \triangleq \{\mu_j : DM_j \in \mathcal{G}_i\}$. We refer to [38] for determining the team optimal control law $\{\mu_{\mathcal{G}_1}^o, \dots, \mu_{\mathcal{G}_K}^o\}$ that minimizes the expected value of Cost (9.19). Under general assumptions, determining this optimal control law is quite a hard task. The concept of sequential partition is, however, a useful premise for addressing, in the next section, optimal decentralized control problems of practical interest. Indeed, Theorem 2 and Corollary 1 of [38] show the applicability of dynamic programming (DP) to teams with a sequential partition.

9.3 The Optimal Decentralized Control Problem and its Reduction to a Nonlinear Programming Problem

9.3.1 Statement of Problem DC

After the description of general issues on team theory presented so far, let us consider optimal control problems characterized by the presence of several controllers C_1, \dots, C_R acting on the same dynamic system, that is,

$$\mathbf{x}_{t+1} = f_t(\mathbf{x}_t, \mathbf{u}_t^1, \dots, \mathbf{u}_t^R, \xi_t), \quad t = 0, 1, \dots, T-1, \quad (9.20)$$

where \mathbf{u}_t^j is the control generated by C_j . Each controller is provided with a perfect memory and observes the state vector by the measurement channel

$$\mathbf{y}_t^j = g_t^j(\mathbf{x}_t, \eta_t^j), \quad j = 1, \dots, R, \quad t = 0, 1, \dots, T-1. \quad (9.21)$$

The random vectors $\mathbf{x}_0, \xi_t, \eta_t^j, t = 0, 1, \dots, T-1, j = 1, \dots, R$, are mutually independent. We assume that, at the stage t , the controllers exchange the measures \mathbf{y}_{t-k}^j and the controls \mathbf{u}_{t-k}^j among them with k steps of delay. This scheme of information exchange is usually called a *k-step delay sharing information structure*. Note, however, that a variety of possible exchanges of information about measures and/or controls may occur. All this gives rise to what is commonly referred to as *optimal decentralized control*. For a survey on the subject, see, for instance, [21]. Therefore, the information vector of the controller C_j is given by

$$\begin{aligned} \mathbf{I}_0^j &= \text{col}(\mathbf{y}_0^j), \quad j = 1, \dots, R, \\ \mathbf{I}_t^j &= \text{col}(\mathbf{y}_0^j, \dots, \mathbf{y}_t^j, \mathbf{u}_0^j, \dots, \mathbf{u}_{t-1}^j), \quad 1 \leq t < k, \quad j = 1, \dots, R, \\ \mathbf{I}_t^j &= \text{col}(\tilde{\mathbf{I}}_{t-k}, \mathbf{y}_{t-k+1}^j, \dots, \mathbf{y}_t^j, \mathbf{u}_{t-k+1}^j, \dots, \mathbf{u}_{t-1}^j), \quad t \geq k, \quad j = 1, \dots, R, \end{aligned}$$

where

$$\begin{aligned}\tilde{\mathbf{I}}_t &= \text{col}(\mathbf{y}_\tau^m, \mathbf{u}_\tau^m, \tau = 0, 1, \dots, t, m = 1, \dots, R) \\ &= \text{col}(\tilde{\mathbf{I}}_{t-1}, \mathbf{y}_t^m, \mathbf{u}_t^m, m = 1, \dots, R), \quad t \geq 0\end{aligned}$$

(with $\tilde{\mathbf{I}}_{-1}$ the empty vector) is the information vector shared among the R controllers after k steps of delay. The control functions of C_1, \dots, C_R take on the form

$$\mathbf{u}_t^j = \mu_t^j(\mathbf{I}_t^j), \quad t = 0, 1, \dots, T-1, j = 1, \dots, R. \quad (9.22)$$

The cost function is given by

$$J = \sum_{t=0}^{T-1} h_t(\mathbf{x}_t, \mathbf{u}_t^1, \dots, \mathbf{u}_t^R, \xi_t) + h_T(\mathbf{x}_T). \quad (9.23)$$

Then, the optimal decentralized control problem can be stated as follows.

Problem DC. *Find the optimal control functions*

$$\mathbf{u}_t^{j\circ} = \mu_t^{j\circ}(\mathbf{I}_t^j), \quad t = 0, 1, \dots, T-1, j = 1, \dots, R, \quad (9.24)$$

that minimize the expected value of the cost J . \triangleleft

We consider for simplicity (and without loss of generality) the case of two controllers C_1 and C_2 . The control structure of the decentralized control scheme is shown in Fig. 9.5. C_1 and C_2 (for $t = 0, 1, \dots, T-1$) give rise to a team of $2T$ decision-makers $DM_1^1, \dots, DM_T^1, DM_1^2, \dots, DM_T^2$.

Fig. 9.5 Decentralized control with k -step delay sharing information structure. The dashed arrows represent the exchange of information between the controllers C_1 and C_2

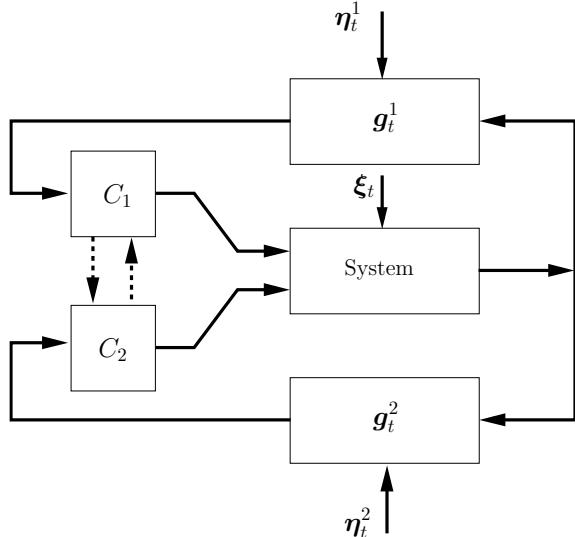
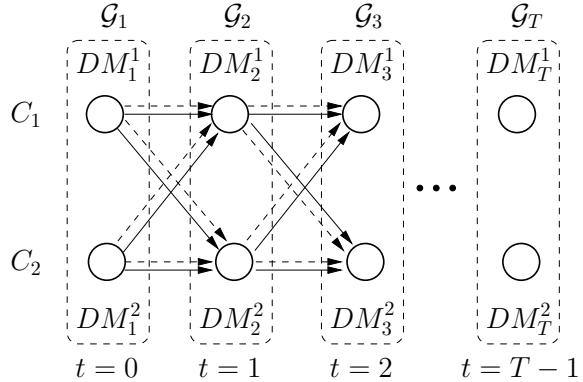


Fig. 9.6 Decentralized control with one-step-delay sharing information structure. ©1978 IEEE. Adapted, with permission, from [38]



We now consider two examples. The former is characterized by a sequential partition while the latter has none.

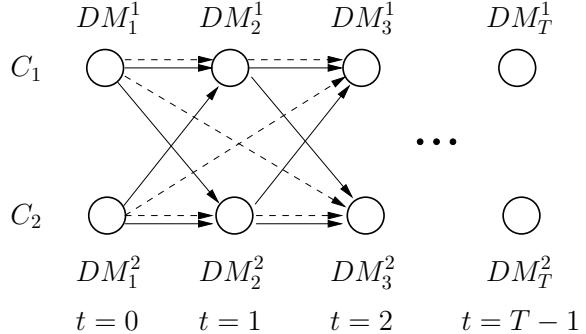
Example 9.2 Decentralized control with one-step-delay sharing information structure.

The information structure diagram of the team is depicted in Fig. 9.6 for the case of two controllers \$C_1\$ and \$C_2\$. From this diagram, it is clear that the team \$\mathcal{G}\$ can be decomposed in a *sequential partition* \$\mathcal{G} = \{\mathcal{G}_1, \dots, \mathcal{G}_T\}\$, where the subsets \$\mathcal{G}_1, \dots, \mathcal{G}_T\$ are shown in Fig. 9.6. Let \$z \triangleq \text{col}(\mathbf{x}_0, \boldsymbol{\xi}_t, \boldsymbol{\eta}_t^1, \boldsymbol{\eta}_t^2, t = 0, 1, \dots, T-1)\$. Note that Cost (9.23) has the structure of Cost (9.19). Thus, on the basis of what said in the previous section, Problem DC can be solved by applying Corollary 1 of [38]. It is easy to show that the solution of Problem DC can also be derived by applying DP (see, for instance, [33, 37]). This requires the computation of the probability density \$p_t(\mathbf{x}_t | \tilde{\mathbf{I}}_{t-1})\$, which can be calculated as the density \$p_t(\mathbf{x}_t | \mathbf{I}_t)\$ in Problem C3. Under general assumptions, the computational drawbacks are similar. Finally, if LQG assumptions are verified, Problem DC can be solved in a closed form [33, 37]. \triangleleft

Example 9.3 Decentralized control with two-step-delay sharing information structure.

If the decentralized control is characterized by an exchange of information among the controllers with more than a one-step delay, generally there is no sequential partition. To show this, consider a delay of two steps. In Fig. 9.7, both the precedence diagram among DMs and the memory-communication diagram are depicted. It can be seen from the figure that many partitions verify Condition (a) of Definition 9.12 but none of them verifies Condition (b). As a consequence, it does not seem reasonable to think that DP is generally able to solve decentralized control problems with two-step delay or more than two-step-delay sharing information structures. \triangleleft

Fig. 9.7 Decentralized control with two-step-delay sharing information structure. ©1978 IEEE. Adapted, with permission, from [38]



Remark 9.3 We conclude this section by recalling some terms commonly used in the literature on optimal decentralized control (see, for example, [21]):

1. If only one controller is present and has perfect recall (see the information structure diagram of Fig. 9.3), Problem DC is simply given by Problem C3. The corresponding information structure is called *classical*.
2. Decentralized control systems with controllers composed of sets of DMs characterized by a *partially nested* information structure are also called decentralized control systems with *quasi-classical* information structure. Apart from the case in which the LQG assumptions are verified and a dynamic team can be converted to a static one (see Sect. 9.2.2), solving optimal decentralized control problems with this type of information structure still remains a largely open issue.
3. Information structures that are neither classical nor quasi-classical are called *nonclassical*. \triangleleft

9.3.2 Approximation of Problem DC by the Nonlinear Programming Problems DC_n

It is easy to show that Problem DC can be solved in an approximate way by following the same procedure that we used for Problems C2 and C3. We just have to constrain the control functions (9.22) to take on the form of FSP functions, that is,

$$\mathbf{u}_t^j = \tilde{\mu}_t^j(\mathbf{I}_t^j, \mathbf{w}_{tn_t^j}^j), \quad t = 0, 1, \dots, T-1, \quad j = 1, \dots, R. \quad (9.25)$$

By using repeatedly State Equation (9.20), we eliminate the vectors \$\mathbf{x}_t\$, \$\mathbf{u}_t^j\$, \$\mathbf{y}_t^j\$, and \$\mathbf{I}_t^j\$ in Cost (9.23). This cost can be expressed as a function of the vector

$$\mathbf{z} \triangleq \text{col}(\mathbf{x}_0, \boldsymbol{\xi}_t, \boldsymbol{\eta}_t^j, \quad t = 0, 1, \dots, T-1, \quad j = 1, \dots, R)$$

of all the primitive random variables, and of the vector

$$\mathbf{w}_n \triangleq \text{col} (\mathbf{w}_{tn_t}^j, t = 0, 1, \dots, T - 1, j = 1, \dots, R),$$

where n is the *global model complexity* of the R chains of FSP functions (9.25). Thus, the cost becomes

$$J(\mathbf{w}_n, \mathbf{z}). \quad (9.26)$$

To sum up, the following problem has to be solved (compare with Problems C2 $_n$ and C3 $_n$).

Problem DC $_n$. *Find the vector \mathbf{w}_n° that minimizes the expected cost*

$$\hat{J}(\mathbf{w}_n) = \underset{\mathbf{z}}{\mathbb{E}} J(\mathbf{w}_n, \mathbf{z}).$$

△

Therefore, similar to what said in Chaps. 7 and 8 (for increasing values of n), the infinite-dimensional optimization (IDO) Problem DC can be approximated to any degree of accuracy by finite-dimensional optimization (FDO) or nonlinear programming (NLP) problems.

As we said in Sect. 5.3.2 and repeated for Problems C2 $_n$ and C3 $_n$, provided that suitable assumptions on differentiability are verified, one can solve Problem DC $_n$ by some gradient-based batch algorithm or by stochastic gradient algorithms. We shall give more details on the solution procedure in the next two sections, when we address Problem W, related to the Witsenhausen counterexample, and Problem ROUT, related to dynamic routing in traffic networks. In both problems, we shall apply the stochastic gradient algorithm.

Remark 9.4 We point out that the ERIM is a particularly efficient tool to address team optimal control problems. In practice, optimal solutions can be determined in closed-form only in a few cases, like in LQG static teams and in LQG dynamic teams with partially nested information structure. Dynamic programming can be applied under special assumptions as in the optimal decentralized control with one-step sharing information structure. In this problem, however, DP is subject to the computational difficulties that we encountered in considering Problem C3 while the ERIM can be applied without additional difficulties also in nonclassical optimal decentralized control problems. △

9.4 The Witsenhausen Counterexample

In this section, we address an approximate solution to the famous (and still unsolved) Witsenhausen counterexample [35], which exhibits the essential difficulties arising when a partially nested information structure does not hold, even if the LQG assumptions are verified. In Sect. 9.4.1, we state the problem and present the suboptimal nonlinear solution proposed by Witsenhausen, which attains a lower cost than the

optimal linear one. The approximate solution obtained by the ERIM is described in Sect. 9.4.2, and numerical results are shown in Sect. 9.4.3.

9.4.1 Statement of the Problem

Let us consider two decision-makers, DM_1 and DM_2 , which generate their controls u_1 and u_2 by the functions $u_1 = \mu_1(y_1)$ and $u_2 = \mu_2(y_2)$, respectively ($\mu_1, \mu_2 : \mathbb{R} \rightarrow \mathbb{R}$). DM_1 and DM_2 have different information on the stochastic environment, i.e. (see (9.1))

$$\begin{aligned} I_1 &= y_1 = g_1(u_2, z) = x, \\ I_2 &= y_2 = g_2(u_1, z) = u_1 + \eta, \end{aligned} \quad (9.27)$$

where $z \triangleq \text{col}(x, \eta)$, $p(x) = \mathcal{N}(0, \sigma_x^2)$ and $p(\eta) = \mathcal{N}(0, 1)$ are two independent Gaussian random variables. Then, we can state the following problem.

Problem W. Let $k \in \mathbb{R}$. Find the optimal control functions $\mu_1^\circ(y_1)$ and $\mu_2^\circ(y_2)$ that minimize the cost functional

$$\hat{J}_W(\mu_1, \mu_2) \triangleq \underset{x, \eta}{\mathbf{E}} [J_W(\mu_1, \mu_2)], \quad (9.28)$$

where

$$J_W(\mu_1, \mu_2) = k^2[x - \mu_1(y_1)]^2 + [\mu_1(y_1) - \mu_2(y_2)]^2.$$

□

The control scheme and the information structure diagram of the team are shown in Fig. 9.8a, b, respectively. In the information structure diagram, there is only the precedence diagram but not the memory-communication one. Therefore, the information structure is not partially nested.

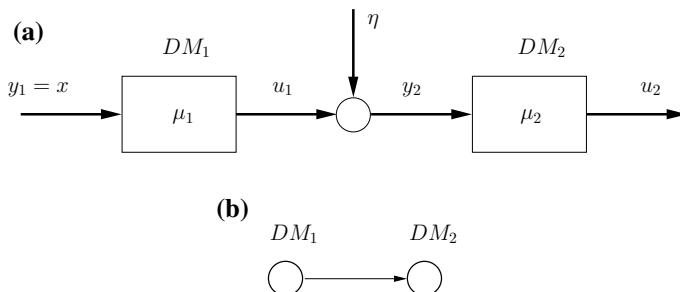


Fig. 9.8 a Control scheme of Problem W. b The information structure of the team is not partially nested. $DM_1 \rightarrow DM_2$ but $DM_1 \not\simeq DM_2$ (the dashed arc does not exist)

Even if Problem W aims to highlight the difficulties of a very simple nonclassical information structure, an interesting engineering interpretation of the problem can be drawn. The scheme in Fig. 9.8a may be considered as a communication system where a transmitter (DM_1) has to send a message to a receiver (DM_2) through a noisy channel. Alternatively, the scheme may refer to a two-stage optimal control problem without a perfect recall of the measures. Indeed, the controller at the second stage (represented by DM_2) does not recall the value of x that it measured at the first stage (i.e., when it was represented by DM_1) because of the action of the noise η .

Witsenhausen demonstrated that the best linear solution to Problem W is not guaranteed to be the optimal one. He did so by using the functions

$$\mu_1(y_1) = \sigma_x \operatorname{sgn}(y_1), \quad (9.29a)$$

$$\mu_2(y_2) = \sigma_x \tanh(\sigma_x y_2), \quad (9.29b)$$

which, for $k^2\sigma_x^2 = 1$ and $k \rightarrow 0$, give a lower cost than the one of the best solution attained in the class of linear control functions. For Control Functions (9.29), \hat{J}_W approaches the value $2(1 - \sqrt{2/\pi}) \simeq 0.404$. The optimal solution (which in [35] was proved to exist for any $k^2 > 0$) was not found, however.

No progress in deriving an optimal solution was made by “discretizing” Problem W [13], that is, by assigning discrete values to the random variables x and η . The corresponding probability densities are then converted into probability mass functions. Also the control variables u_1 and u_2 are constrained to only take on discrete values. The difficulty of finding an optimal solution was explained in [26] by demonstrating that the discretized Problem W is NP-complete.

In [35], the following findings are reported. Assume that $E[\mu_1^2(y_1)] < \infty$. Then, for a certain $\mu_1(y_1)$, let us denote by $\mu_2^*(y_2 | \mu_1)$ the control function $\mu_2(y_2)$ minimizing $\hat{J}_W(\mu_1, \mu_2)$. It can be demonstrated that such a function is given by

$$\mu_2^*(y_2 | \mu_1) = \frac{\int \mu_1(y_1) \varphi[y_2 - \mu_1(y_1)] \varphi(y_1) dy_1}{\int \varphi[y_2 - \mu_1(y_1)] \varphi(y_1) dy_1} = E[\mu_1(y_1) | y_2], \quad (9.30)$$

where $\varphi(z) \triangleq \frac{1}{\sqrt{2\pi}} e^{-z^2/2}$.

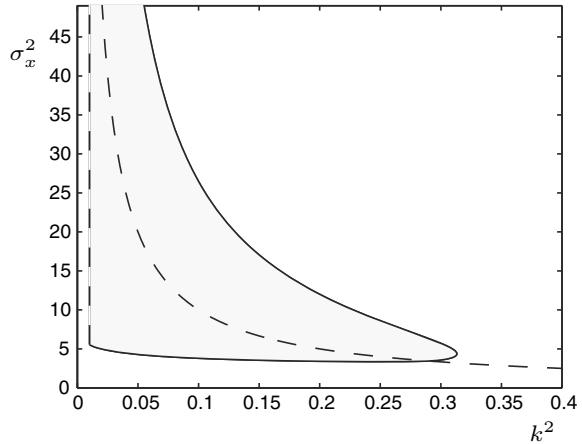
In [6], it is shown that, by replacing σ_x with $\varepsilon = \sqrt{2/\pi}\sigma_x$, Control Functions (9.29) become

$$\mu_1(y_1) = \varepsilon \operatorname{sgn}(y_1), \quad (9.31a)$$

$$\mu_2^*(y_2 | \mu_1) = 2 \tanh(\varepsilon y_2), \quad (9.31b)$$

and give an asymptotic cost $\hat{J}_W = 1 - (2/\pi) \simeq 0.363$. This cost is lower than the cost $\hat{J}_W \simeq 0.404$ derived in [35] for Functions (9.29).

Fig. 9.9 In the gray area, the optimized Witsenhausen solutions outperform the best linear solutions. ©2001 IEEE. Reprinted, with permission, from [5]



In the sequel, we shall consider control functions of the form (9.31), where ε has to be considered as a “free scalar.” This scalar must be determined so as to minimize Cost (9.28), that is,

$$\hat{J}_W[\varepsilon \operatorname{sgn}(y_1), \varepsilon \tanh(\varepsilon y_2)],$$

for given values of k and σ_x . The control functions corresponding to the scalar ε° that minimizes \hat{J}_W are called *optimized Witsenhausen* (OW) control functions.

We point out that the OW solutions outperform the best linear solutions only in a limited portion of the plane of k^2 and σ_x^2 . In Fig. 9.9, this portion is given by the gray area, which was determined numerically. Not surprisingly, this area encompasses (for not too large values of k) the curve $\sigma_x^2 k^2 = 1$ (which corresponds to the case considered by Witsenhausen), indicated in the figure by the dashed line. Of course, the gray area does not contain the semiaxis $k^2 = 0$. Indeed, if $k^2 = 0$, then any pair of control functions $\mu_1(y_1) = \mu_2(y_2) = \text{constant}$ reduces the cost \hat{J}_W to zero.

9.4.2 Application of the ERIM for the Approximate Solution of Problem W

As usual, the application of the ERIM for solving Problem W approximately requires that the DMs’ control functions $u_1 = \mu_1(y_1)$ and $u_2 = \mu_2(y_2)$ are constrained to take on the form of FSP functions $u_1 = \tilde{\mu}_1(y_1, \mathbf{w}_{1n_1})$ and $u_2 = \tilde{\mu}_2(y_2, \mathbf{w}_{2n_2})$ by which the optimal control functions μ_1° and μ_2° have to be approximated, respectively.

Substitution of the FSP functions $\tilde{\mu}_1$ and $\tilde{\mu}_2$ into the cost $J_W(\mu_1, \mu_2)$ yields the cost function

$$J_W(\mathbf{w}_n, x, \eta),$$

where $\mathbf{w}_n \triangleq \text{col}(\mathbf{w}_{1n_1}, \mathbf{w}_{2n_2})$. Thus, instead of the original IDO Problem W, for increasing values of the global model complexity n , we have to solve NLP problems. Let us state them as follows.

Problem W_n . *Find the optimal vector \mathbf{w}_n° that minimizes the cost function*

$$\hat{J}_W(\mathbf{w}_n) = \underset{x, \eta}{\mathbb{E}} [J_W(\mathbf{w}_n, x, \eta)], \quad (9.32)$$

where

$$\begin{aligned} J_W(\mathbf{w}_n, x, \eta) = & k^2[x - \tilde{\mu}_1(y_1, \mathbf{w}_{1n_1})]^2 \\ & + [\tilde{\mu}_1(y_1, \mathbf{w}_{1n_1}) - \tilde{\mu}_2(y_2, \mathbf{w}_{2n_2})]^2. \end{aligned}$$

△

Problem W_n can be solved by one of the minimization techniques for the solution of Problem P_n described in Sect. 5.3. In the numerical results described in the next section, we chose the stochastic gradient algorithm. Then (2.37) and (5.50) become

$$\begin{aligned} \mathbf{w}_n(k+1) = & \mathbf{w}_n(k) - \alpha_k \nabla_{\mathbf{w}_n} J_W[\mathbf{w}_n(k), x(k), \eta(k)], \\ k = 0, 1, \dots, \end{aligned} \quad (9.33)$$

where the sequence $\{x(k), \eta(k), k = 0, 1, \dots\}$ is generated by a random selection of the samples $x(k)$ and $\eta(k)$ according to their respective probability density functions. The stepsize is given by the usual decreasing function $\alpha_k = c_1/(c_2 + k)$, $c_1, c_2 > 0$. Two passes alternate: (1) a *forward* pass, in which the pair $x(k), \eta(k)$ is generated randomly, (2) a *backward* pass, in which the gradient in (9.33) is computed by back-propagation through $\tilde{\mu}_2(y_2, \mathbf{w}_{2n_2})$ and $\tilde{\mu}_1(y_1, \mathbf{w}_{1n_1})$ thus yielding the new parameter vector $\mathbf{w}_n(k+1)$.

As regards the choice of the FSP functions, it seems reasonable to adopt \mathcal{C} - or \mathcal{L}_2 -approximating functions because we have little a priori information about the functions μ_1° and μ_2° to be approximated. If one believes that such functions may not be continuous (as the literature on the Witsenhausen counterexample leads us to presume), \mathcal{L}_2 -approximating functions (or even other functions defined on other suitable normed spaces) turn out to be appropriate. Therefore, in the numerical experiments, we shall use sigmoidal OHL neural networks, which benefit from both the \mathcal{C} - and \mathcal{L}_2 -density properties.

9.4.3 Numerical Results Obtained by the ERIM

In order to test the ERIM on a team problem whose optimal solution is known, let us address the following problem stated in [6] (see [5]).

Problem BB (Bansal and Başar). Find the optimal functions μ_1^* and μ_2^* that minimize the cost functional

$$\hat{J}_{BB}(\mu_1, \mu_2) = \underset{x, \eta}{\mathbb{E}} \{k_0 \mu_1^2(y_1) + s_{01} \mu_1(y_1) x + [x - \mu_2(y_2)]^2\},$$

where the information structure is given by (9.27), $p(x) = \mathcal{N}(0, \sigma_x^2)$, and $p(\eta) = \mathcal{N}(0, \sigma_v^2)$. \triangleleft

The problem is proved to have the following optimal linear solution:

$$\begin{aligned}\mu_1^*(y_1) &= \lambda^* y_1, \\ \mu_2^*(y_2) &= \frac{\lambda^* \sigma_x^2}{\lambda^* \sigma_x^2 + \sigma_v^2} y_2,\end{aligned}\tag{9.34}$$

where λ^* is a root of a certain fifth-order equation. We now apply the ERIM by replacing the control functions with FSP functions. Then, a sequence of Problems BB_n has to be stated to approximate Problem BB (we omit the statement of Problem BB_n as its meaning should be clear). To perform the numerical computations presented in this section, sigmoidal neural OHL networks were chosen.

To evaluate the costs $\hat{J}_W(\mathbf{w}_n^\circ)$ and $\hat{J}_{BB}(\mathbf{w}_n^\circ)$, related to the approximate optimal control functions $\tilde{\mu}_1(y_1, \mathbf{w}_{1n_1}^\circ)$ and $\tilde{\mu}_2(y_2, \mathbf{w}_{2n_2}^\circ)$, their empirical estimates

$$\mathbb{E}(J) \simeq \frac{1}{L} \sum_{l=1}^L J[\mathbf{w}^\circ, x(l), \eta(l)]$$

were computed. $x(l)$ and $\eta(l)$ were generated randomly on the basis of their respective probability densities, and L is a sufficiently large number of samples (the value $L = 10^8$ was set). To simplify the notation, in the expression for \hat{J}_W° and in the following the subscript n is dropped.

Clearly, as in most other numerical examples of the book, one cannot be sure that Stochastic Gradient Algorithm (9.33) leads to a global minimum and not to a local one owing to the extremely high complexity of the surfaces of the cost functions. Then, $\text{col}(\mathbf{w}_1^\circ, \mathbf{w}_2^\circ)$ denotes a global or local minimum attained by the stochastic gradient. Moreover, the pair of functions $\tilde{\mu}_1(y_1, \mathbf{w}_1^\circ)$ and $\tilde{\mu}_2(y_2, \mathbf{w}_2^\circ)$ is simply called a *neural solution* (and not “globally or locally optimal neural solution”). In any case, even if it is possible that only local minima were computed, to judge by the fact that very low costs were derived for Problems W_n when compared with other ones obtained in the literature, the points $(\mathbf{w}_1^\circ, \mathbf{w}_2^\circ)$ are surely “good” local minima. Concerning the costs of the optimal solutions to Problem BB (\hat{J}_{BB}°), the costs of the best linear solutions to Problem W (\hat{J}_W°), and the costs of the OW solutions ($\hat{J}_{W_{opt}}^\circ$), all of them can be evaluated analytically.

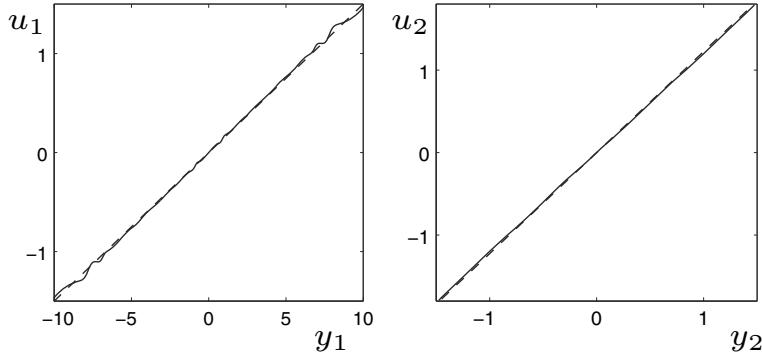


Fig. 9.10 Neural (—) and optimal (---) control functions for a Problem BB with $k_0 = 10$, $s_{01} = -1$, $\sigma_x^2 = 10$, and $\sigma_v^2 = 1$. ©2001 IEEE. Reprinted, with permission, from [5]

(a) *Problem BB: comparison of optimal and neural solutions*

In Fig. 9.10, the control functions $\tilde{\mu}_1(y_1, \mathbf{w}_1^\circ)$ and $\tilde{\mu}_2(y_2, \mathbf{w}_2^\circ)$ are compared with Control Functions (9.34). The functions $\tilde{\mu}_1$ and $\tilde{\mu}_2$ were implemented by using neural OHL networks with $n_1 = n_2 = 30$ sigmoidal units. The costs $\hat{J}_{BB}^\circ = 8.913$ and $\hat{J}_{BBE}^\circ = 8.9171$ were obtained for the parameters $k_0 = 10$, $s_{01} = -1$, $\sigma_x^2 = 10$, and $\sigma_v^2 = 1$ (\hat{J}_{BBE}° denotes the optimal cost computed by the ERIM). Convergence to \hat{J}_{BBE}° was reached after about 10^4 steps of the stochastic gradient algorithm. As can be seen, the neural control functions practically coincide with the optimal ones.

(b) *Problem W: comparison of optimized Witsenhausen solutions, best linear solutions, and neural solutions*

On the basis of the encouraging results obtained in solving approximately Problem BB, the ERIM was applied to search approximate optimal solutions to Problem W. Here the method has to compete at least with the OW solutions and with the best linear solutions. Then, neural solutions were computed in several points (k^2, σ_x^2) of the plane of k^2 and σ_x^2 . Fig. 9.11 (which is an extension of Fig. 9.9) shows a grid of points (k^2, σ_x^2) at which numerical computations and comparisons were performed. The results appear to be interesting and, in certain cases, a little surprising, as compared with those previously presented in the literature. They are discussed in the following paragraphs. The neural control functions were implemented by using OHL networks with $n_1 = 150$ and $n_2 = 30$ sigmoidal units. Convergence to the cost J_{WE}° (the optimal cost computed by the ERIM) was obtained in about $3 \cdot 10^6$ steps of the stochastic gradient algorithm.

1. The shapes of the neural control functions strongly depend on the particular points (k^2, σ_x^2) considered. This is clearly shown in Fig. 9.11, which is divided into various areas, each characterized by the particular form of DM_1 's control function $\tilde{\mu}_1(y_1, w_1^\circ)$ (note that, in a sense, $\tilde{\mu}_2(y_2, w_2^\circ)$ "follows" $\tilde{\mu}_1(y_1, w_1^\circ)$; this is explained by (9.30)). The areas are separated by dashed strips to point out that the boundary lines are not clear-cut but derived by clustering the grid points

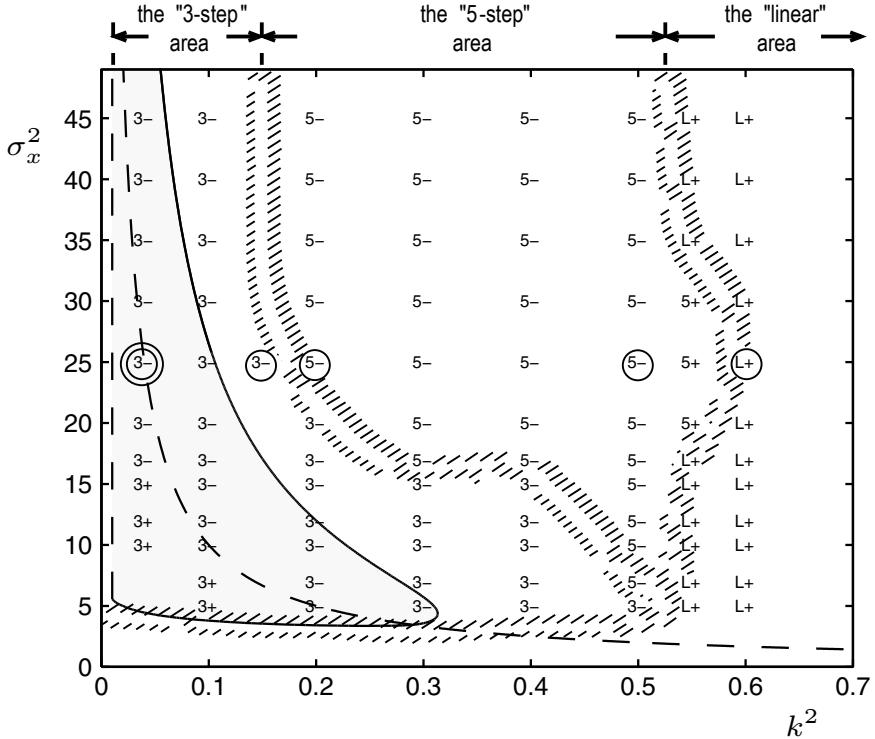


Fig. 9.11 The plane of k^2 and σ_x^2 and the areas where the neural control functions are “similar” in shape. ©2001 IEEE. Reprinted, with permission, from [5]

(k^2, σ_x^2) at which the neural control functions were computed and compared with the best linear and OW control functions. Each point is labeled by an integer or the letter “L” and by the sign “+” or “-.” The integer denotes the number of steps present in the function $\tilde{\mu}_1(y_1, w_1^\circ)$ when it has the form of a scale. The letter “L” indicates that such a function turned out to be linear. The sign “+” (“-”) means that the cost \hat{J}_{WE} is higher (lower) than the lowest cost derived by enforcing both the linear control functions (yielding the cost \hat{J}_{WL}°) and the OW control functions (yielding the cost \hat{J}_{Wopt}°).

- Moving from one area to another in Fig. 9.11, a change can be noticed in the number of steps present in the function $\tilde{\mu}_1(y_1, w_1^\circ)$. σ_x^2 was kept constant, for instance, at the value $\sigma_x^2 = 25$ and k^2 was increased from the value $k^2 = 0.04$ to the values $k^2 = 0.15, 0.2, 0.5$, and 0.6 (see the points marked with circles in Fig. 9.11). The comparisons of the neural, best linear, and OW control functions are shown for these values in Fig. 9.12. For $k^2 = 0.04$ and $k^2 = 0.15$, the neural solutions outperform the OW solutions, which, in turn, outperform the best linear

Fig. 9.12 Neural (—), optimized Witsenhausen (---), and best linear (---) control functions at the points on the plane k^2 and σ_x^2 considered in Table 9.1 and marked with circles in Fig. 9.11. ©2001 IEEE. Reprinted, with permission, from [5]

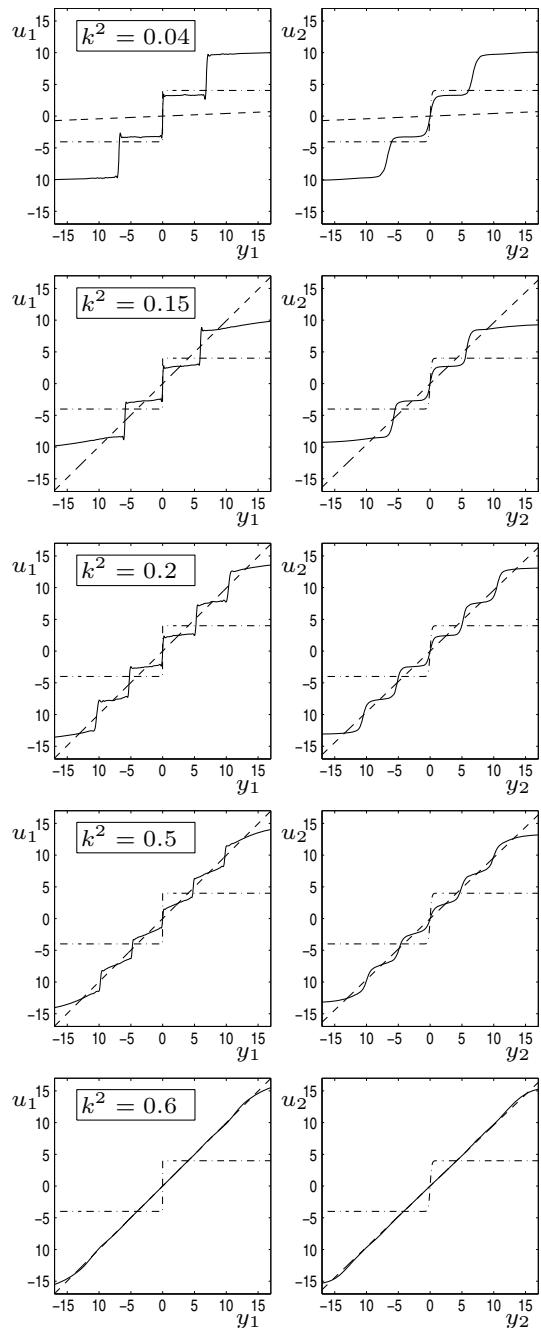


Table 9.1 Costs of the neural, optimized Witsenhausen, and best linear solutions

k^2	σ_x^2	\hat{J}_{WE}°	\hat{J}_{Wopt}°	\hat{J}_{WL}°
0.04	25	0.2473	0.3648	0.9613
0.15	25	0.57	1.3642	0.9614
0.2	25	0.68	1.8185	0.9614
0.5	25	0.95	4.5439	0.9515
0.6	25	0.97	5.4523	0.9615

ones (we are inside the gray area). The values of the costs are given in Table 9.1. It can be seen from Fig. 9.12 that $\tilde{\mu}_1(y_1, w_1^\circ)$ and $\tilde{\mu}_2(y_2, w_2^\circ)$ contain a linear component (a structure of this type was proposed in [6]).

Increasing k^2 to the value 0.2 leads inside the area where $\tilde{\mu}_1(y_1, w_1^\circ)$ has five steps. In this area, the neural solution outperforms the best linear solution, which outperforms the OW one (we are outside the gray area). For $k^2 = 0.5$, we are still in the five-step area. It can be noticed that the slope of the linear component increased, thus approaching the slope of the DM_1 's best linear control function. Also, the cost \hat{J}_{WE}° tends to get closer to the cost \hat{J}_{WL}° . Finally, when the value $k^2 = 0.6$ is reached, the neural control function turns out to be nearly linear and practically coincides with the best linear control function (it is likely that other steps in $\tilde{\mu}_1(y_1, w_1^\circ)$ would appear if the number of basis functions in the approximating networks were increased). The cost \hat{J}_{WE}° is a little higher than \tilde{J}_{WL}° , but this is probably due to the use of an insufficient number of sigmoidal units. The same occurs for some points in the gray area below the curve $k^2\sigma_x^2 = 1$.

9.4.4 Improved Numerical Results and Variants of the Witsenhausen Counterexample

In [7], ordinal optimization was applied to solve Problem W for the values $k^2 = 0.04$ and $\sigma_x^2 = 25$ (see the point marked with a double circle in Fig. 9.11). Ordinal optimization is a technique for speeding up the process of stochastic optimization via parametric simulation. It works by successive narrowings of the solutions' space that are guided by the designer's a priori knowledge of the problem and by probabilistic rules used to choose smaller and smaller subsets in which one can find the largest number of "good solutions" with the highest probability. To initialize ordinal optimization in Problem W, symmetric staircase functions $\mu_1(y_1)$ were considered on the positive semiaxis of y_1 , which was divided into n intervals $I_1 = [0, a_1], \dots, I_n = [a_{n-1}, +\infty)$, where $\text{Prob}\{y_1 \in I_i\} = 0.5/n$, $i = 1, \dots, n$. The values of $u_1 = \mu_1(y_1)$, for y_1 belonging to any interval I_i , were uniformly picked from the interval $(-3\sigma_x, 3\sigma_x)$. Then, on the basis of the approximate evaluation of the cost function, the following restrictions were applied in sequence:

(1) for each interval I_i , $u_1 = \mu_1(y_1)$ takes its values uniformly from $(-0.5\sigma_x, 2.5\sigma_x)$, (2) $\mu_1(y_1)$ is an increasing function in $(-0.5\sigma_x, 2.5\sigma_x)$, and (3) $\mu_1(y_1)$ is a two-value increasing step function in $(-0.5\sigma_x, 2.5\sigma_x)$. Finally, the best jump point of $\mu_1(y_1)$ was determined by checking several points of y_1 . The best control function turned out to be

$$\begin{aligned}\mu_1(y_1) &= 3.1686 \quad \text{for } 0 \leq y_1 < 6.41, \\ \mu_1(y_1) &= 9.0479 \quad \text{for } y_1 \geq 6.41,\end{aligned}\tag{9.35}$$

which gave the cost $\hat{J}_{DH} = 0.1901$.

For the same values $k^2 = 0.04$ and $\sigma_x^2 = 25$, the ERIM gave a three-step neural control function $\tilde{\mu}_1(y_1, w_1^\circ)$ that was quite similar to (9.35), despite the presence of a slight-slope linear function (see Fig. 9.12). However, we obtained the cost $\hat{J}_{WE}^\circ = 0.2473 > \hat{J}_{DH}$. Note that, if one were not pleased with the obtained shapes of $\tilde{\mu}_1(y_1, w_1^\circ)$ and $\tilde{\mu}_2(y_2, w_2^\circ)$ (for example, because unexpected slight oscillations are visible in the diagram of $\tilde{\mu}_1(y_1, w_1^\circ)$), one might replace $\tilde{\mu}_1(y_1, w_1^\circ)$ with a suitably piecewise linear function made up of straight segments determined by a least-squares criterion. Then, only $\tilde{\mu}_2(y_2, w_2)$ should be optimized. This was done and the cost $\hat{J}_{WE}^\circ = 0.1735 < \hat{J}_{DH}$ was obtained.

For the values $k^2 = 0.04$ and $\sigma_x^2 = 25$, other local and global minima were searched by experimenting with the ERIM using a large number of initial weight vectors $w_n(0)$. One of these initial values gave the control functions $\tilde{\mu}_1(y_1, \tilde{w}_1^\circ)$ and $\tilde{\mu}_2(y_2, \tilde{w}_2^\circ)$ shown in Fig. 9.13 and revealing four-step shapes. After regularization of $\tilde{\mu}_1(y_1, \tilde{w}_1^\circ)$ with a least-squares technique, the minimum cost turned out to be $\hat{J}_{WE}^\circ = 0.1701$. We can conclude that two local minima were found (or a local and a global one).

In [19], an improvement over the solution in [7] was obtained by exploiting a “hierarchical search” based on the application of successive “reasonable” hypotheses on the structure of the solution and refinements based on a “scattered search” method.

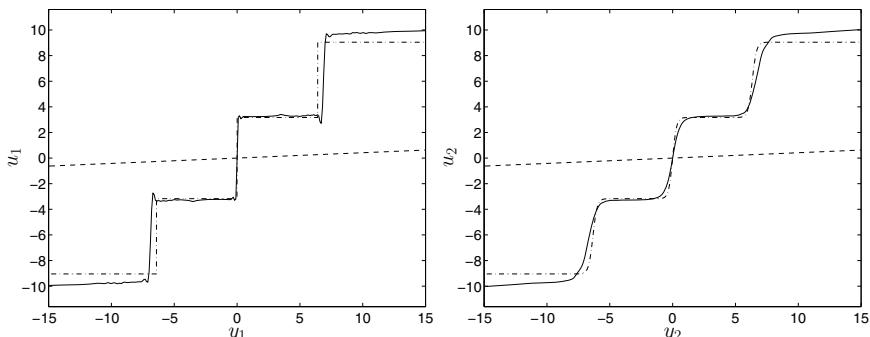


Fig. 9.13 Neural four-step control functions (-), optimized Witsenhausen (- · -), and best linear (- - -) control functions at $k^2 = 0.04$ and $\sigma_x^2 = 25$. ©2001 IEEE. Reprinted, with permission, from [5]

By constraining the function μ_1 to be symmetric and piecewise constant, the function $\mu_2^*(y_2|\mu_1)$ can be computed analytically. Further improvements over the solution were achieved by breaking each main step into smaller ones. A symmetric function with six “main steps” was obtained with no step in the origin and a cost $\hat{J}_{LLH} = 0.1673$ was attained (where “LLH” stands for the initials of the authors of [19]).

From the above, we can draw a general conclusion on the application of the ERIM. Indeed, it is important to point out that the ERIM was able to find the correct shapes of the control functions without using either a priori information about the problem or the not-utterly-simple decision-making procedures described in [7, 19].

In [20], the Witsenhausen counterexample was faced by a learning approach usually adopted in potential games and called *fading-memory joint-strategy fictitious play*. A solution similar in shape to that shown in Fig. 9.13 was derived with a cost $\hat{J}_{LMS} = 0.1671$ (where “LMS” stands for the initials of the authors of [20]).

Some variants of the Witsenhausen counterexample were proposed [2, 10, 11, 24, 30]. In particular, in [24], the presence of “side information” was considered consisting of an additional Gaussian channel between DM_1 and DM_2 . In [30], a different cost is addressed where an induced norm is minimized. It is shown that in this case, the linear strategies are uniformly optimal.

A vector version of the counterexample is considered in [11], where the scalar random variables x and η of the original formulation become vector quantities and the cost J_W is modified to account for the norms of the quantities involved and normalized with respect to the vector lengths. Lower bounds on the minimum cost are obtained in [11], while in [10, 28] lattice-based team control laws are proposed that guarantee the achievement of a cost within a constant factor with respect to the optimal one. For the scalar (classical) case, such a factor is not larger than 8.

Experimental and theoretical results on the Witsenhausen counterexample continue to appear in the literature. Making an analysis of these results is not within our purposes because we are only interested in verifying the good behavior of the ERIM in approximately solving a famous problem proved to be hard. However, it is interesting to mention the comparison on some methods, proposed in the literature, given in [25], where an approach based on deterministic annealing was applied. See also the best results to date, appeared in the literature and reported in Table I of [25].

Among the theoretical results, [32] and [31, Part II] are worth noting. Static and dynamic team problems are approximated by finite models obtained through uniform discretization of the observation and control spaces of the DMs on finite grids. It is shown that, under mild conditions, these models enable the design of control laws approximating the optimal one with arbitrary precision. As regards the Witsenhausen counterexample, in [32] and [31, Chap. 9] the authors show the possibility of constructing an ε -optimal control law for any $\varepsilon > 0$ through the solution of a simpler problem. Interestingly, [32] reduces the Witsenhausen counterexample from a dynamic team to a static one. This possibility was proved by Witsenhausen in [36] not only for his counterexample but also for a class of decentralized optimal control problems. However, the “equivalent” static team is characterized by a nonquadratic cost. Hence, the reduction was not considered practically useful.

Very briefly, the reduction from the Witsenhausen counterexample to its static version is derived from the demonstration that the precedence diagram in Fig. 9.2b can be eliminated and that the control functions to be optimized depend on the primitive independent random variables x and η , i.e., $u_1 = \bar{\mu}_1(x)$ and $u_2 = \bar{\mu}_2(\eta)$. Problem W is then reduced to the static problem WS by introducing the cost

$$\hat{J}_{WS}(\bar{\mu}_1, \bar{\mu}_2) \triangleq \underset{x, \eta}{\mathbb{E}} [J_{WS}(\bar{\mu}_1, \bar{\mu}_2)],$$

where

$$J_{WS}(\bar{\mu}_1, \bar{\mu}_2) \triangleq e^{\bar{\mu}_1(x)[2\eta - \bar{\mu}_1(x)]/2} \{[\bar{\mu}_1(x) - \bar{\mu}_2(\eta)]^2 + k^2[x - \bar{\mu}_1(x)]\}.$$

It can be shown that $\hat{J}_W(\mu_1, \mu_2) = \hat{J}_{WS}(\bar{\mu}_1, \bar{\mu}_2)$, where there is a one-to-one correspondence between the pairs (μ_1, μ_2) and $(\bar{\mu}_1, \bar{\mu}_2)$, which implies that minimizing \hat{J}_{WS} minimizes also \hat{J}_W and vice versa. The static cost \hat{J}_{WS} is nonquadratic, though.

9.4.5 Theoretical Results on the Application of the ERIM

By exploiting the reduction of Problem W to a static team optimization problem, mentioned in Sect. 9.4.4, in [9] the accuracy of suboptimal solutions obtained via the ERIM was estimated. FSP functions $\bar{\mu}_1(x, \tilde{\mathbf{w}}_{1n})$ and $\bar{\mu}_2(\eta, \tilde{\mathbf{w}}_{2n})$ given by OHL networks with the same number n of sigmoidal units were considered therein. In the following, we briefly summarize the theoretical results obtained [9].

For x and η belonging to finite symmetric intervals centered in the origin, it was proven that the optimal functions $\bar{\mu}_1^\circ(x)$ and $\bar{\mu}_2^\circ(\eta)$ can be approximated arbitrarily well by OHL sigmoidal neural networks with rate of order $1/n$ in the sup-norm on bounded intervals (hence also in the \mathcal{L}_2 -norm). The proof technique used therein to derive such estimates strongly depends on the scalar nature of the state in Problem W and cannot be applied to its vector generalization proposed in [12]. By exploiting the Maurey–Jones–Barron Theorem (see Theorem 3.3), estimates of order $1/\sqrt{n}$ were obtained in the \mathcal{L}_2 -norm. Although they are worse than the abovementioned ones of order $1/n$, they have the advantage of being potentially applicable also to the vector generalization of Problem W. Moreover, for the case in which control laws made up of OHL networks with sigmoidal computational units are used, it was estimated how far the value of the functional to be minimized in Problem W is from the minimum value.

9.5 Dynamic Routing in Traffic Networks

In this section, we apply the ERIM to the solution of a very complex IDO problem consisting in the optimal control of large-scale traffic networks (see [4]; preliminary results are reported in [1, 27]). These systems, characterized by great technological importance, include (among others) the following networks: computer networks extending over large geographical areas, store-and-forward packet-switching communication networks, freeway systems, reservoir networks in multi-reservoir systems, and queueing networks in manufacturing systems.

The aforesaid traffic networks share some characteristics. The networks can be modeled as graphs in which a set of nodes (with storing capabilities) is connected through a set of links (which may be characterized by traffic delays and transportation costs) that cannot be loaded by traffic without exceeding their finite capacities. Traffic flows can be described by continuous variables, even if the “objects” exchanged among the nodes are discrete in nature (e.g., data packets, messages, cars, work-pieces, etc.). This is justified whenever the number of objects is so large as to require macroscopic modeling. As the traffic flows entering the networks are assumed to vary over time, the nodes or, to be more specific, the DMs acting at the nodes may be requested to change the amount of traffic flow to be sent to their neighboring nodes. Then, a *dynamic routing problem* arises. The DMs are realistically assumed (i) to generate their routing decisions on the basis of local information and possibly of some data received from other nodes, typically the neighboring ones, and (ii) to cooperate in the accomplishment of a common goal, that is, the minimization of the total traffic cost. Therefore, they can be regarded as the cooperating members of team organizations.

It is worth noting that the term “decision-maker” is used here in a sense that differs slightly from that so far considered in this chapter. Indeed, when we speak of a DM generating routing decisions at a certain node i of the network at the instants $t = 0, 1, \dots, T - 1$, we actually refer to a family of T DMs, each only making decisions once at a specific instant t . According to a notation different from that used previously, in the sequel DM_i will denote the family of the decision-makers $DM_i(0), DM_i(1), \dots, DM_i(T - 1)$. Then, the notation $DM_i(t)$ has a meaning congruent with the term “decision-maker” so far adopted, whereas DM_i has a new meaning. DM_i has then the same meaning as the term “controller” that we used in other parts of the book. This, however, should not entail any risk of ambiguity.

The control of traffic flows is assumed to be exerted over a finite-time horizon. It follows that the routing problem assumes major importance in congestion situations, when the traffic accumulated at the nodes must be cleared as soon as possible. The finite-horizon routing problem however can be extended rather easily to the infinite-horizon case, and be solved approximately after being restated in a receding-horizon form. In the following sections, we focus on store-and-forward packet-switching networks.

9.5.1 Modeling the Communication Network

To describe the model of the communication network, let us consider a directed graph $\mathcal{C} = (\mathcal{N}, \mathcal{L})$, consisting of a set \mathcal{N} of N nodes and a set \mathcal{L} of oriented links. Besides the set \mathcal{N} , there is a single destination node D to which all the nodes of \mathcal{N} send their messages (see an example of this network in Fig. 9.15). The uniqueness of the destination node is assumed only for simplicity, without loss of generality (we refer to [4, 27] for the case in which there are more destination nodes). At each node $i \in \mathcal{N}$, stochastic flows of messages may be inserted from outside the communication network. The sets of nodes that are upstream and downstream neighbors of node i are denoted by $\mathcal{P}(i)$ and $\mathcal{S}(i)$, respectively. We assume that at each node $i \in \mathcal{N}$, there is a queue in which messages are stored for the destination node D . Therefore, the node D must be reachable from each node of \mathcal{N} . We assume that the rates of messages that are sent from one node to another are updated synchronously by control variables at discrete periodic instants $0, 1, \dots, T - 1$.

Up to this point, the communication network may be considered as the discrete-time version of the continuous-time model proposed by Segall in [34] and used in subsequent works. Following [17], we also take into account the various possible delays that may occur in a real communication network (besides the queueing delays). More specifically, by p_{ij} we denote the total delay in transmitting a message from the node i (i.e., the time between starting and ending the transmission of a message), in propagating it on the link joining node i to node j , and in processing the message at the node j (i.e., inserting it in the queue and performing the routing computations). We assume that p_{ij} can be rounded off to an integer, that is, to a multiple of the sample period.

Under the previous assumptions, the communication-network dynamics and the constraints on the traffic flows are expressed as

$$m_i(t+1) = m_i(t) u_{ii}(t) + \sum_{k \in \mathcal{P}(i)} m_k(t - p_{ki}) u_{ki}(t - p_{ki}) + r_i(t), \quad i \in \mathcal{N}, \quad t = 1, \dots, T - 1, \quad (9.36)$$

$$u_{ij}(t) \geq 0, \quad i \in \mathcal{N}, \quad j \in \tilde{\mathcal{S}}(i), \quad (9.37)$$

$$\sum_{j \in \tilde{\mathcal{S}}(i)} u_{ij}(t) = 1, \quad i \in \mathcal{N}, \quad (9.38)$$

$$f_{ij}(t) = m_i(t) u_{ij}(t) \leq C_{ij}, \quad i \in \mathcal{N}, \quad j \in \mathcal{S}(i), \quad (9.39)$$

where $\tilde{\mathcal{S}}(i) \triangleq \mathcal{S}(i) \cup \{i\}$, $m_i(t)$ specifies the length of the queue of messages at node i at time t , $f_{ij}(t)$ is the traffic flow routed from node i to node j , $r_i(t) \geq 0$ represents the stochastic external input flow entering the node i , C_{ij} is the capacity of link $(i, j) \in \mathcal{L}$. $u_{ij}(t)$ are the control variables representing the portion of $m_i(t)$ that is routed from node i to node $j \in \mathcal{S}(i)$, and $u_{ii}(t)$ is the portion of $m_i(t)$ that has to remain at the node i . According to these definitions, we have the equalities in (9.39).

The initial values of the queue lengths $m_i(0) \geq 0$ and the values of the traffic flows $f_{ij}(\tau) \geq 0$, $\tau = -1, \dots, -p_{ij}$, are assumed to be random variables. Note that the vector $\mathbf{x}(t) \triangleq \text{col}[\mathbf{x}_i(t), i \in \mathcal{N}]$, where $\mathbf{x}_i(t) \triangleq \text{col}[m_i(t), f_{ki}(t-\tau), k \in \mathcal{P}(i), \tau = 1, \dots, p_{ki}]$, plays the role of a state vector. Let $\mathbf{r} \triangleq \text{col}(\mathbf{r}_i, i \in \mathcal{N})$, where $\mathbf{r}_i \triangleq \text{col}[r_i(t), t = 0, 1, \dots, T-1]$. All the random variables are assumed to be mutually independent and characterized by given probability density functions.

Note that the model described previously is nonlinear. It can be shown that the nonlinear model is equivalent (in the sense specified in [27]) to the linear one introduced by Segall [34] and modified subsequently. As to the network dynamics, the linear model is given by

$$m_i(t+1) = m_i(t) + \sum_{k \in \mathcal{P}(i)} f_{ki}(t-p_{ki}) - \sum_{j \in \mathcal{S}(i)} f_{ij}(t) + r_i(t), \quad i \in \mathcal{N}. \quad (9.40)$$

The constraints of the linear model (which are linear, too) can be easily derived from the constraints of the nonlinear model. The introduction of the seemingly more complex nonlinear model is justified by the fact that it allows us to take some constraints involved by the use of the linear model implicitly into account.

The cost to be minimized is given by the total traffic in the queues

$$J \triangleq \sum_{t=1}^T \left[\sum_{i \in \mathcal{N}} \alpha_{it} m_i(t) + \sum_{(i,j) \in \mathcal{L}} \sum_{\tau=\max(0,t-p_{ij})}^{t-1} \beta_{ijt} f_{ij}(t-\tau) \right], \quad (9.41)$$

where α_{it} and β_{ijt} are positive weighting constants. Cost (9.41) may also be written in the following form, which will be handled more easily in deriving the control law approximating the optimal one:

$$J \triangleq \sum_{t=1}^T \sum_{i \in \mathcal{N}} \alpha_{it} m_i(t) + \sum_{t=0}^{T-1} \sum_{(i,j) \in \mathcal{L}} \left(\sum_{\tau=t+1}^{\min(t+p_{ij}, T)} \beta_{ij\tau} \right) f_{ij}(t). \quad (9.42)$$

The coefficients α_{it} and β_{ijt} allow Cost (9.41) or (9.42) to take into account a wide variety of practical situations. If these coefficients are set equal to 1, then the cost functions give the total time spent by the messages at the nodes and on the links of the communication network (from stage 1 up to stage T). The presence of the subscript t is useful as time-increasing weighting coefficients may accelerate the clearance of a traffic congestion in the communication network. Finally, the subscripts i, j in the coefficients β_{ijt} enable one to associate possible costs that have to be paid to convey messages through the link joining node i to node j .

9.5.2 Statement of the Dynamic Routing Problem

In modeling the communication network, we admitted that the decision-makers DM_1, \dots, DM_N generate the control variables without sharing a common information set. To be more specific, we assume that each decision-maker DM_i stores in its memory at stage t a *local information vector*, which we denote by $\mathbf{I}_i^l(t)$, made up of the state variables it measured in its node and of the control variables it generated. If the total number T of stages is large and if DM_i has a limited memory, then we have to reduce the dimension of $\mathbf{I}_i^l(t)$, and we assume that DM_i only stores in its memory the most “recent” values of local state variables and control. Then, we can define the vector

$$\mathbf{I}_i^l(t) \triangleq \text{col} [m_i(\tau), \tau = t - \bar{\tau}_i, \dots, t, u_{ij}(\tau), \tau = t - \bar{\tau}_i, \dots, t - 1, j \in \mathcal{S}(i)],$$

where $\bar{\tau}_i$ is a suitable integer related to the maximum number of variables that can be stored in DM_i 's memory (compare with the “limited memory” in Problem C3-LM).

We also assume that DM_i may receive *information messages* from the other DMs of the network (typically, but not necessarily, from its neighboring nodes). Let $\mathcal{P}_{\text{inf}}(i)$ be the set of DMs that send information messages to DM_i . We denote by $\mathbf{I}_{ki}(t)$ the information message received by DM_i at stage t from $DM_k \in \mathcal{P}_{\text{inf}}(i)$ at the stage $t - \bar{p}_{ki}$ after \bar{p}_{ki} steps of delay. The information messages may be conveyed by the links of the communication network or by dedicated channels. As information messages are much simpler than the basic ones, we assume that no capacity constraints are imposed on the dedicated channels. For the same reason, the information messages can be conveyed by the links of the communication network even if the flow of basic messages saturate them. Therefore, as the dedicated channels are not subject to capacity constraints, the information messages cannot give rise to queues.

To simplify the notation and without loss of generality, we shall assume that DM_i only retains in its memory the last information vector $\mathbf{I}_{ki}(t)$ it received from DM_k . To sum up, the overall *personal information vector* of DM_i is given by

$$\mathbf{I}_i(t) \triangleq \text{col} [\mathbf{I}_i^l(t), \mathbf{I}_{ki}(t), k \in \mathcal{P}_{\text{inf}}(i)]. \quad (9.43)$$

Then, the control functions take on the form

$$u_{ij}(t) = \mu_{ijt} [\mathbf{I}_i(t)], \quad i \in \mathcal{N}, j \in \tilde{\mathcal{S}}(i), t = 0, 1, \dots, T - 1. \quad (9.44)$$

We can now state the following particular case of Problem T.

Problem ROUT. *Find the optimal control functions $\mu_{ijt}^\circ[\mathbf{I}_i(t)]$ that minimize the expected value of Cost (9.41) or (9.42), i.e.,*

$$\underset{x(0), r}{\mathbb{E}} J.$$

△

As stated previously, each of the N decision-makers DM_i actually represents a family of T decision-makers $DM_i(0), \dots, DM_i(T - 1)$. Then, Problem ROUT is related to Problem PM with $M = N \cdot T$ (see the discussion in Sect. 2.9).

Obviously, Problem ROUT is neither LQG nor is it, in general, characterized by a partially nested information structure. Therefore, it is much more difficult than the Witsenhausen's counterexample and there can be no hope for solving it in closed form.

9.5.3 Approximate Solution of Problem ROUT by the ERIM

As Problem ROUT is far from verifying the assumptions enabling us to obtain its optimal solution in an analytical way, following the ERIM approach we constrain the control functions (9.44) to take on the form of FSP functions given by sigmoidal OHL neural networks. Then, we have

$$\begin{aligned} \tilde{u}_{ij}(t) &= \tilde{\mu}_{ij} [\mathbf{I}_i(t), \mathbf{w}_i(t)] \triangleq \sum_{k=1}^{n_i} c_{ikt}^{(j)} \varphi_k [\mathbf{I}_i(t), \tilde{\mathbf{w}}_{ik}(t)], \\ i &\in \mathcal{N}, \quad j \in \tilde{\mathcal{S}}(i), \quad t = 0, 1, \dots, T - 1, \end{aligned} \quad (9.45)$$

where

$$\mathbf{w}_i(t) \triangleq \text{col}[c_{ikt}^{(j)}, k = 1, \dots, n_i, j \in \tilde{\mathcal{S}}(i), \tilde{\mathbf{w}}_{ik}(t), k = 1, \dots, n_i].$$

The coefficients $c_{ikt}^{(j)}$ and the components of the vectors $\tilde{\mathbf{w}}_{ik}(t)$ are parameters to be determined so as to minimize the expected value of Cost (9.41) or Cost (9.42). In order to simplify the notation, we dropped the integers n_{it} (which would denote the numbers of neural units of the OHL networks) in both the vectors $\mathbf{w}_i(t)$ and the control functions \tilde{u}_{ij} . For the sake of simplicity, we assume that the integers n_{it} are time-invariant; hence it is $n_{it} = n_i$, $t = 0, 1, \dots, T - 1$.

In the following paragraphs, we shall choose OHL networks with sigmoidal basis functions of the form $\sigma(z) = 1/(1 + e^{-z})$. Then, the outputs $\tilde{u}_{ij}(t)$ of the OHL networks (9.45) are the arguments of these sigmoidal functions. This assures the nonnegativity of the variables $\bar{u}_{ij}(t)$ generated by the functions

$$\begin{aligned} \bar{u}_{ij}(t) &= \bar{\mu}_{ij} [\mathbf{I}_i(t), \mathbf{w}_i(t)] \triangleq \sigma \{ \tilde{\mu}_{ij} [\mathbf{I}_i(t), \mathbf{w}_i(t)] \}, \\ i &\in \mathcal{N}, \quad j \in \tilde{\mathcal{S}}(i), \quad t = 0, 1, \dots, T - 1. \end{aligned} \quad (9.46)$$

To further simplify the notation, we aggregate Functions (9.44), (9.45), and (9.46) into the vectorial forms

$$\mathbf{u}_i(t) = \boldsymbol{\mu}_{it} [\mathbf{I}_i(t)], \quad i \in \mathcal{N}, \quad t = 0, 1, \dots, T - 1, \quad (9.47)$$

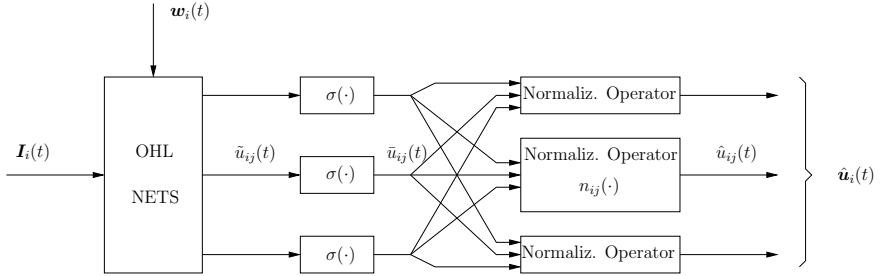


Fig. 9.14 Scheme of the neural routing function $\hat{u}_i(t) = \hat{\mu}_i[\mathbf{I}_i(t), \mathbf{w}_i(t)]$

$$\tilde{\mathbf{u}}_i(t) = \tilde{\mu}_i[\mathbf{I}_i(t), \mathbf{w}_i(t)], \quad i \in \mathcal{N}, t = 0, 1, \dots, T-1, \quad (9.48)$$

$$\bar{\mathbf{u}}_i(t) = \bar{\mu}_i[\mathbf{I}_i(t), \mathbf{w}_i(t)], \quad i \in \mathcal{N}, t = 0, 1, \dots, T-1, \quad (9.49)$$

where $\boldsymbol{\mu}_{it} \triangleq \text{col} [\mu_{ijt}, j \in \tilde{\mathcal{S}}(i)]$, and similar definitions hold for $\tilde{\boldsymbol{\mu}}_i$, $\bar{\boldsymbol{\mu}}_i$, $\mathbf{u}_i(t)$, $\tilde{\mathbf{u}}_i(t)$, and $\bar{\mathbf{u}}_i(t)$. Finally, at all the time instants $t = 0, 1, \dots, T-1$, we obtain the control variables $\hat{u}_{ij}(t)$ (generated by the decision-maker $DM_i(t)$) by introducing the following normalization operators:

$$\hat{u}_{ij}(t) = n_{ij}[\bar{\mathbf{u}}_i(t)] \triangleq \bar{u}_{ij}(t) / \sum_{k \in \tilde{\mathcal{S}}(i)} \bar{u}_{ik}(t), \quad i \in \mathcal{N}, j \in \tilde{\mathcal{S}}(i). \quad (9.50)$$

We let

$$\mathbf{n}_i[\bar{\mathbf{u}}_i(t)] \triangleq \text{col} \{n_{ij}[\bar{\mathbf{u}}_i(t)], j \in \tilde{\mathcal{S}}(i)\}, \quad i \in \mathcal{N}. \quad (9.51)$$

In the remaining part of this section, we call *neural routing functions* (NRFs) the mappings resulting from the composition of the OHL networks with the sigmoidal functions $\sigma(\cdot)$ and the normalization operators described by (9.50) and (9.51) (see the scheme shown in Fig. 9.14).

Therefore, the structures of the NRNs are given by

$$\hat{u}_{ij}(t) = \hat{\mu}_{ij}[\mathbf{I}_i(t), \mathbf{w}_i(t)] = n_{ij} \{ \bar{\mu}_i[\mathbf{I}_i(t), \mathbf{w}_i(t)] \}, \quad i \in \mathcal{N}, j \in \tilde{\mathcal{S}}(i),$$

and, in aggregated form,

$$\begin{aligned} \hat{\mathbf{u}}_i(t) &= \hat{\mu}_i[\mathbf{I}_i(t), \mathbf{w}_i(t)] = \mathbf{n}_i \{ \bar{\mu}_i[\mathbf{I}_i(t), \mathbf{w}_i(t)] \} \\ &\triangleq \text{col} \{ \hat{\mu}_{ij}[\mathbf{I}_i(t), \mathbf{w}_i(t)], j \in \tilde{\mathcal{S}}(i) \}, \quad i \in \mathcal{N}. \end{aligned} \quad (9.52)$$

The above normalization operators enable us to remove Constraints (9.38). Furthermore, as already said, the use of the sigmoidal functions $\sigma(\cdot)$ ensures the fulfillment

of the nonnegativity constraints (9.37). As to Constraints (9.39), we eliminate them by adding commonly used differentiable penalty functions of the form

$$\rho_{ij}[f_{ij}(t)] = \{\max[0, f_{ij}(t) - C_{ij}]\}^2 \quad (9.53)$$

to the cost (9.41) or (9.42). It follows that the new cost function is obtained by substituting the fixed-structure NRFs (9.52) into the cost (9.41) or (9.42) and by adding the penalty functions (9.53). If we address Cost (9.42), we have

$$\begin{aligned} J[\mathbf{w}_n, \mathbf{x}(0), \mathbf{r}] &= \sum_{t=1}^T \sum_{i \in \mathcal{N}} \alpha_{it} m_i(t) \\ &+ \sum_{t=0}^{T-1} \sum_{(i,j) \in \mathcal{L}} \left[\left(\sum_{\tau=t+1}^{\min(t+p_{ij}, T)} \beta_{ij\tau} \right) f_{ij}(t) + K_L \rho_{ij}[f_{ij}(t)] \right], \end{aligned} \quad (9.54)$$

where K_L is a positive constant and $\mathbf{w}_n \triangleq \text{col}[\mathbf{w}_i(t) : i \in \mathcal{N}, t = 0, \dots, T-1]$. The subscript n is the *global model complexity* of the set of OHL neural networks. Then, we can state the following unconstrained NLP problem approximating Problem ROUT.

Problem ROUT_n. *Find the vector \mathbf{w}_n° that minimizes the expected cost*

$$\underset{\mathbf{x}(0), \mathbf{r}}{\mathbb{E}} \{J[\mathbf{w}_n, \mathbf{x}(0), \mathbf{r}]\}.$$

△

9.5.4 Computation of the Neural Routing Functions via Stochastic Approximation

Once again, as pointed out in Sect. 5.3, we can choose between a gradient-based algorithm and the stochastic gradient algorithm. As for Problem W, we shall apply the latter method. This choice is useful because it may allow us to adapt the routing control law online when the network characteristics change over time. Then, the stochastic gradient algorithm takes on the form

$$\mathbf{w}_n(k+1) = \mathbf{w}_n(k) - \alpha(k) \nabla_{\mathbf{w}} J[\mathbf{w}_n(k), \mathbf{x}(0|k), \mathbf{r}(k)], \quad k = 0, 1, \dots, \quad (9.55)$$

where $\mathbf{x}(0|k)$ denotes the initial state $\mathbf{x}(0)$ generated at the iteration k . The stepsize is given by $\alpha(k) = c_1/(c_2 + k)$. A “momentum” term $\beta[\mathbf{w}(k) - \mathbf{w}(k-1)]$ is added to (9.55).

Let us now derive the vectors $(\partial J / \partial \mathbf{w}_i(t))^\top$, $i \in \mathcal{N}$, $t = 0, 1, \dots, T-1$, that compose the gradient $\nabla_{\mathbf{w}} J[\mathbf{w}(k), \mathbf{x}(0|k), \mathbf{r}(k)]$ (we omit the subscript n). These

vectors can be computed by means of the backpropagation procedure that is initialized by the partial derivatives (see Fig. 9.14)

$$\frac{\partial J}{\partial \bar{u}_{ij}(t)}, \quad i \in \mathcal{N}, j \in \tilde{\mathcal{S}}(i), t = 0, 1, \dots, T-1. \quad (9.56)$$

To limit the notational burden, in computing the partial derivatives (9.56) (and in choosing the example presented in Sect. 9.5.5), we address a particular scheme of variables measured by the DMs and of information messages exchanged among them. The scheme dealt with is characterized by the following assumptions: a) the local information vector of each decision-maker $DM_i(t)$ is only given by the length of its queue measured at stage t , while the past values of such a length are forgotten (then $\bar{\tau}_i = 0, \forall i$, and $\mathbf{I}_i^l(t) = m_i(t)$), b) the information messages received by each decision-maker $DM_i(t)$ are only the lengths of the queues belonging to its downstream neighbors (i.e., $\mathcal{P}_{\text{inf}}(i) = \mathcal{S}(i)$); these lengths are sent to it with one step of delay (then $\mathbf{I}_{ji}(t) = \text{col}[m_j(t-1), j \in \mathcal{S}(i)]$). It follows that the NRFs (9.52) are given by

$$\hat{\mathbf{u}}_i(t) = \hat{\mu}_i \left\{ \text{col} [m_i(t), m_j(t-1), j \in \mathcal{S}(i)], \mathbf{w}_i(t) \right\}, \quad i \in \mathcal{N}, t = 0, 1, \dots, T-1. \quad (9.57)$$

Clearly, more complex schemes for measuring and storing variables and for exchanging information messages among DMs can be addressed without any additional conceptual difficulty. To further simplify the notation, we shall now drop the iteration step k of Algorithm (9.55).

By taking into account the presence of Normalization Operators (9.50), we obtain

$$\frac{\partial J}{\partial \bar{u}_{ij}(t)} = \sum_{k \in \tilde{\mathcal{S}}(i)} \frac{\partial J}{\partial \hat{u}_{ik}(t)} \frac{\partial \hat{u}_{ik}(t)}{\partial \bar{u}_{ij}(t)}. \quad (9.58)$$

A simple algebra yields

$$\frac{\partial \hat{u}_{ik}(t)}{\partial \bar{u}_{ij}(t)} = \begin{cases} \sum_{\substack{l \in \tilde{\mathcal{S}}(i) \\ l \neq k}} \bar{u}_{il}(t) / \left[\sum_{l \in \tilde{\mathcal{S}}(i)} \bar{u}_{il}(t) \right]^2, & \text{if } k = j, \\ -\bar{u}_{ik}(t) / \left[\sum_{l \in \tilde{\mathcal{S}}(i)} \bar{u}_{il}(t) \right]^2, & \text{if } k \neq j. \end{cases}$$

Let us denote by $\bar{y}_{ii}(t)$ the input variable to the NRF (i.e., the component of the information vector $\mathbf{I}_i(t)$) corresponding to $m_i(t)$. Similarly, we define as $\bar{y}_{ij}(t)$ the input variable corresponding to the state variable $m_j(t-1)$. Let us also define the following variables, which play a basic role in the development of the proposed

updating algorithm,

$$\lambda_{it} \triangleq \frac{\partial J}{\partial m_i(t)}, \quad i \in \mathcal{N}, t = 1, \dots, T. \quad (9.59)$$

Then, the partial derivatives $\frac{\partial J}{\partial \hat{u}_{ij}(t)}$, $j \in \tilde{\mathcal{S}}(i)$, which are used (see (9.58) and (9.46)) to initialize the backpropagation procedure for the OHL network $\tilde{\mu}_i$ (see (9.45) and (9.48)), can be computed by using the results given in the following procedure. Its derivation, which involves simple but tedious computations, is reported in [4].

Backpropagation Initialization Procedure.

The BP procedure for the OHL neural networks $\tilde{\mu}_i[\mathbf{I}_i(t), \mathbf{w}_i(t)]$ is initialized with the following partial derivatives:

$$\frac{\partial J}{\partial \hat{u}_{ii}(t)} = \lambda_{i,t+1} m_i(t), \quad i \in \mathcal{N}, t = 0, 1, \dots, T-1, \quad (9.60)$$

$$\begin{aligned} \frac{\partial J}{\partial \hat{u}_{ij}(t)} = & \left\{ K_L \frac{\partial \rho_{ij}[f_{ij}(t)]}{\partial f_{ij}(t)} + \text{step}[T - (t + p_{ij} + 1)] \lambda_{j,t+p_{ij}+1} \right. \\ & + \sum_{\tau=t+1}^{\min(t+p_{ij}, T)} \beta_{ij\tau} \left. \right\} m_i(t), \\ & i \in \mathcal{N}, j \in \mathcal{S}(i), t = 0, 1, \dots, T-1, \end{aligned} \quad (9.61)$$

where $\text{step}(a) = 1$, if $a \geq 0$, and $\text{step}(a) = 0$, if $a < 0$. The variables λ_{it} can be computed by means of the following equations:

$$\begin{aligned} \lambda_{it} = & \alpha_{it} + \sum_{j \in \mathcal{S}(i)} \left\{ K_L \frac{\partial \rho_{ij}[f_{ij}(t)]}{\partial f_{ij}(t)} + \text{step}[T - (t + p_{ij} + 1)] \lambda_{j,t+p_{ij}+1} \right. \\ & + \sum_{\tau=t+1}^{\min(t+p_{ij}, T)} \beta_{ij\tau} \left. \right\} \hat{u}_{ij}(t) + \lambda_{i,t+1} \hat{u}_{ii}(t) \\ & + \frac{\partial J}{\partial \bar{y}_{ii}(t)} + \text{step}(T - t - 2) \sum_{k \in \mathcal{P}(i)} \frac{\partial J}{\partial \bar{y}_{ki}(t+1)}, \\ & i \in \mathcal{N}, t = 0, 1, \dots, T-1, \end{aligned} \quad (9.62)$$

where the partial derivatives $\frac{\partial J}{\partial \bar{y}_{ii}(t)}$ and $\frac{\partial J}{\partial \bar{y}_{ki}(t+1)}$ are obtained by a simple algebra on the normalization operators (see (9.58)). The recursion is initialized by the conditions

$$\lambda_{iT} = \alpha_{iT}, \quad i \in \mathcal{N}. \quad (9.63)$$

△

On the basis of Equations (9.60) to (9.63), we can now detail the procedure that enables us to compute the gradient in Algorithm (9.55). It consists of the following two “passes,” which alternate throughout the communication network and inside the NRFs acting at each node:

(1) *Forward pass.* At the iteration k , once the realization $\text{col}[\mathbf{x}(0|k), \mathbf{r}(k)]$ has been generated, the DMs distribute the contents of their queues by means of their NRFs (9.52). Then, they measure – or determine – and store in the memory all the terms needed for Equations (9.60) to (9.63).

(2) *Backward pass.* This computation step is performed throughout the communication network and inside the NRFs acting at the nodes. Equation (9.62) clearly explains the messages exchanged for computation: the second term in the summation contains the messages $\lambda_{j,t+p_{ij}+1}$ received by the decision-maker $DM_i(t)$ from the downstream neighbors $DM_j(t + p_{ij} + 1)$, $j \in \mathcal{S}(i)$, and the fifth term is made up of the messages $\frac{\partial J}{\partial \bar{y}_{ki}(t+1)}$ received from the upstream neighbors $DM_k(t+1)$, $k \in \mathcal{P}(i)$.

Once the terms $\frac{\partial J}{\partial \hat{u}_{ij}(t)}$ have been determined (see (9.58)), backpropagation can take place within the NRF of $DM_i(t)$, enabling it to compute the “local gradient vector” $(\partial J / \partial \mathbf{w}_i(t))^\top$. $DM_i(t)$ can now calculate its components of the new vector $\mathbf{w}(k+1)$. Finally, $DM_i(t)$ computes and sends λ_{it} to its upstream neighbors $DM_k(t - p_{ki} - 1)$, $k \in \mathcal{P}(i)$; it also computes and sends $\frac{\partial J}{\partial \bar{y}_{ij}(t)}$ to its downstream neighbors $DM_j(t - 1)$, $j \in \mathcal{S}(i)$.

Remark 9.5 We said previously that the stochastic gradient algorithm can give the method *adaptive* properties. In this respect, we may distinguish between *offline computation* and *online computation*. In the offline case, the realizations $\{\mathbf{x}(0|k), \mathbf{r}(k)\}$ are generated on the basis of their known probability densities. Then, Algorithm (9.55) determines the optimal control law and the computations are completed. In the online case, however, the random variables are generated by the stochastic environment, hence no knowledge of the probability density functions is required. \triangleleft

Remark 9.6 It is interesting to compare the results established in Equations (9.60) to (9.63) with those previously reported in the literature. Equation (9.61) shows similarities with the routing algorithms proposed by Gallager [8]. His model, however, is “quasi-static,” in the sense that the routing nodes do not instantaneously react to changes in the entering traffic flows, but they periodically update their routing functions by following the variations in the expected traffic and in the topology of the communication network. Such a model involves no queues, and the problem is stated in a deterministic context. To be more specific, in (9.62), the first and third terms within the curly brackets are related to the marginal costs appearing in Gallager’s model. Of course, Equation (9.60) and the first and third terms in (9.62) are new as they are associated with the presence of the queues. The fourth term in (9.62) takes into account the capacity of the routing nodes to react instantaneously to changes in

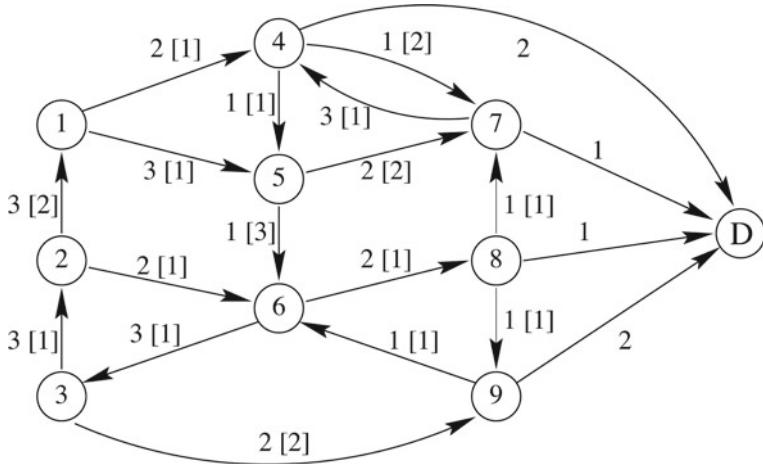


Fig. 9.15 Example of a communication network. ©2001 IEEE. Reprinted, with permission, from [4]

the queue contents by means of the NRFs. A similar term is present in the model proposed in [1] (which also involves no queues). Finally, the fifth term in (9.62) is also new as it takes into account the influence of $DM_i(t)$ on the upstream neighbors $DM_k(t+1)$, $k \in \mathcal{P}(i)$ through the message $m_i(t)$. This term is specific for the particular mechanism of exchange of information messages among the team members. If other mechanisms were chosen, minor changes in such a term should be made. \triangleleft

9.5.5 Numerical Results

To show the effectiveness of the ERIM in a communication network with rather a large number of nodes, let us consider the communication network shown in Fig. 9.15.

The links are labeled by their capacities C_{ij} , and the related delays are indicated by the numbers within the brackets. D is the destination node. Note that many loops are present in the network. The scheme of the information messages exchanged among the DMs is the one assumed in Sect. 9.5.4. No stochastic external inputs entering the nodes are present. The initial lengths of the queues are assumed to be uniformly distributed between 0 and 10 and generated accordingly. The other components of the initial state $\mathbf{x}(0)$, i.e., the traffic flows $f_{ij}(t)$, $(i, j) \in \mathcal{L}$, $t = -p_{ij}, \dots, -1$, have been generated so that they verify the capacity constraints on the network links. The number of control stages is $T = 8$. The cost coefficients are $\alpha_{it} = 1$, $t = 1, \dots, T - 1$, $\alpha_{iT} = 10$, $(i = 1, \dots, 9)$, and $\beta_{ijt} = 1$, $(i, j) \in \mathcal{L}$, $t = 1, \dots, T$. At each node and at all the temporal stages, there is an NRF with 25 sigmoidal units.

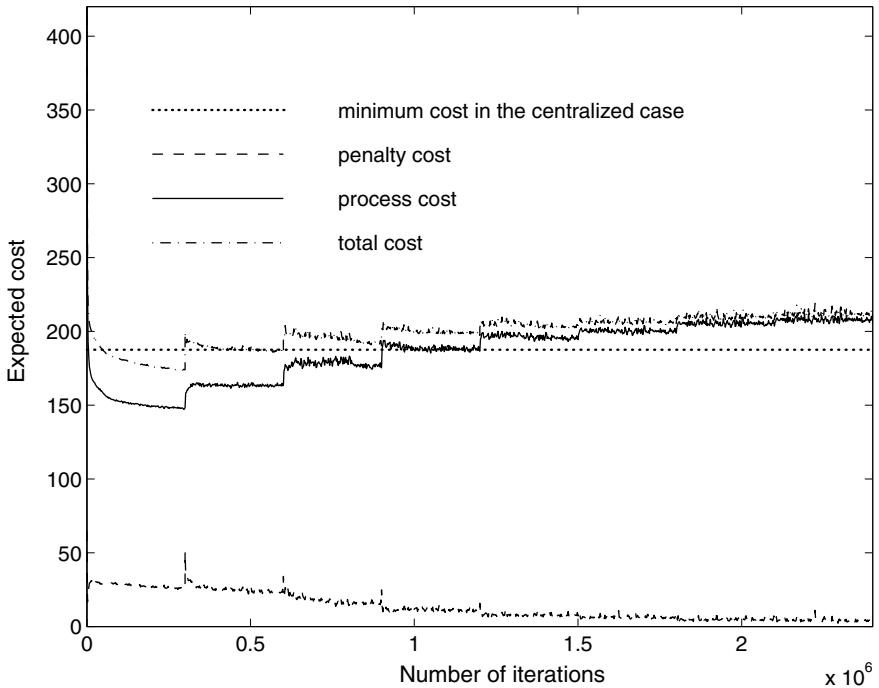


Fig. 9.16 Convergence behavior of the costs. ©2001 IEEE. Reprinted, with permission, from [4]

As we said for Problem ROUT, the example is neither LQG nor provided with a partially nested information structure. Then, no analytical tool is available to derive its optimal solution in closed form. In order to test the correctness of the proposed method, a preliminary routing problem is addressed, in which the nodes are controlled by an *informationally centralized* control system, i.e., by a single DM. This means that all the routing nodes send their personal information vectors instantaneously at each stage t to a central computing center (or to a single DM). Solving the centralized problem by dynamic programming is quite difficult as the dimension of the state vector $\mathbf{x}(t)$ is 9 (the number of routing nodes) plus 22 (the total number of delays in the links). By using Linear State Equation (9.40) and the related linear constraints, the centralized version of Problem ROUT turns out to be a linear programming problem for any fixed value of the initial state $\mathbf{x}(0)$. Then we chose a sufficiently large number of values for $\mathbf{x}(0)$. For each of these values, we solved the corresponding linear programming problem and obtained an estimate of the minimum cost for the centralized optimal routing problem. The value of this cost is given by 187.5 and is shown by the horizontal dotted line in Fig. 9.16.

In the same figure, the behavior of the cost, during the optimization phase of the NRFS by Algorithm (9.55), is also presented (dashed-dotted line). Moreover, periodic steps can be seen in the cost behavior. This fact is due to periodic increments of the

constant K_L multiplying the penalty functions ρ_{ij} (see (9.53)). As is well known, these functions must be handled carefully in order to avoid an ill-conditioned use of the gradient methods. The learning procedure is started using a low value of K_L . Once a reasonable convergence of the cost has been obtained, the value of K_L is increased (clearly, also the cost increases because of the more severe effect imposed by the capacity constraints), and a second training phase is performed. The procedure goes on until the learning mechanism becomes practically independent of the constant K_L . During the learning procedure shown in Fig. 9.16, this constant was increased eight times from the initial value $K_L = 5$ to the final value $K_L = 200$. Convergence of the cost was obtained at the value 211. This value of the cost is not far from the value of the optimal cost in the centralized case. The small difference between the team optimal cost and the optimal centralized cost indicates that the team DMs' scheme for measuring state variables and for exchanging information messages is rather efficient.

References

1. Aicardi M, Davoli F, Minciardi R, Zoppoli R (1990) Decentralized routing, teams and neural networks in communications. In: Proceedings of the IEEE conference on decision and control, pp 2386–2390
2. Başar T (2008) Variations on the theme of the Witsenhausen counterexample. In: Proceedings of the IEEE conference on decision and control, pp 161–1619
3. Başar T, Olsder GJ (1998) Dynamic noncooperative game theory. SIAM Classics Appl Math
4. Baglietto M, Parisini T, Zoppoli R (2001) Distributed-information neural control: the case of dynamic routing in traffic networks. *IEEE Trans. Neural Netw* 12:485–502
5. Baglietto M, Parisini T, Zoppoli R (2001) Numerical solutions to the Witsenhausen counterexample by approximating networks. *IEEE Trans Autom Control* 46:1471–1477
6. Bansal R, Başar T (1987) Stochastic teams with nonclassical information revisited: when is an affine law optimal? *IEEE Trans Autom Control* 32:554–559
7. Deng M, Ho YC (1999) An ordinal optimization approach to optimal control problems. *Automatica* 35:331–338
8. Gallager RG (1977) A minimum delay routing algorithm using distributed computation. *IEEE Trans Commun* 25:73–85
9. Gnecco G, Sanguineti M (2012) New insights into Witsenhausen's counterexample. *Optim Lett* 6:1425–1446
10. Grover P, Park SY, Sahai A (2013) Approximately optimal solutions to the finite-dimensional Witsenhausen's counterexample. *IEEE Trans Autom Control* 58:2189–2204
11. Grover P, Sahai A (2008) A vector version of Witsenhausen's counterexample: a convergence of control, communication and computation. In: Proceedings of the IEEE conference on decision and control, pp 1636–1641
12. Grover P, Sahai A (2010) Witsenhausen's counterexample as assisted interference suppression. *Int J Syst Control Commun* 2:197–237
13. Ho Y-C, Chang TS (1980) Another look at the nonclassical information structure problem. *IEEE Trans Autom Control* 25:537–540
14. Ho Y-C, Chu KC (1972) Team decision theory and information structures in optimal control problems - Part I. *IEEE Trans Autom Control* 17:15–22
15. Ho YC, Chu KC (1973) On the equivalence of information structures in static and dynamic teams. *IEEE Trans Autom Control* 18(2):187–188

16. Ho Y-C, Chu KC (1979) Information structure in dynamic multiperson control problems. *Automatica* 10:341–351
17. Iftar A, Davison EJ (1998) A decentralized discrete-time controller for dynamic routing. *Int J Control* 69:599–632
18. Kim KH, Roush FW (1987) Team theory. Ellis Horwood Limited, Halsted Press
19. Lee JT, Lau E, Ho Y-C (2001) The Witsenhausen counterexample: a hierarchical search approach for nonconvex optimization problems. *IEEE Trans Autom Control* 46:382–397
20. Li N, Marden JR, Shamma JS (2009) Learning approaches to the Witsenhausen counterexample from a view of potential games. In: Proceedings of the 2009 joint ieee conference on decision and control and chinese control conference, pp 157–162
21. Mahjan A, Martins NC, Rotkowitz MC, Yüksel S (2012) Information structures in optimal decentralized control. In: Proceedings of the IEEE conference on decision and control, pp 1291–1306
22. Marschak J (1955) Elements for a theory of teams. *Manag Sci* 1(2):127–137
23. Marschak J, Radner R (1972) Economic theory of teams. Yale University Press
24. Martins N (2006) Witsenhausen's counterexample holds in the presence of side information. In: Proceedings of the IEEE conference on decision and control, pp 1111–1116
25. Mehmetoglu M, Akyol E, Rose K (2014) A deterministic annealing approach to Witsenhausen's counterexample. In: Proceedings of the 2014 IEEE international symposium on information theory (ISIT), pp 3032–3036
26. Papadimitriou CH, Tsitsiklis JN (1986) Intractable problems in control theory. *SIAM J Control Optim* 24:639–654
27. Parisini T, Zoppoli R (1993) Team theory and neural networks for dynamic routing in traffic and communication networks. *Inf Decis Technol* 19:1–18
28. Park SY, Grover P, Sahai A (2009) A constant-factor approximately optimal solution to the Witsenhausen counterexample. In: Proceedings of the 2009 Joint IEEE conference on decision and control and chinese control conference, pp 2881–2886
29. Radner R (1962) Team decision problems. *Ann Math. Stat* 33:857–881
30. Rotkowitz M (2006) Linear controllers are uniformly optimal for the Witsenhausen counterexample. In: Proceedings of the IEEE conference on decision and control, pp 553–558
31. Saldi N, Linder T, Yüksel S (2018) Finite Approximations in discrete-time stochastic control. Birkhäuser
32. Saldi N, Yüksel S, Linder T (2017) Finite model approximations and asymptotic optimality of quantized policies in decentralized stochastic control. *IEEE Trans Autom Control* 62:2360–2373
33. Sandell NR Jr, Athans M (1974) Solution of some nonclassical LQG stochastic decision problems. *IEEE Trans Autom Control* 19:108–116
34. Segall A (1977) The modeling of adaptive routing in data-communication networks. *IEEE Trans Commun* 25:85–95
35. Witsenhausen HS (1968) A counterexample in stochastic optimum control. *SIAM J Control* 6:131–147
36. Witsenhausen HS (1988) Equivalent stochastic control problems. *Math Control Signals Syst* 1:3–11
37. Yoshikawa T (1975) Dynamic programming approach to decentralized stochastic control problems. *IEEE Trans Autom Control* 20:796–797
38. Yoshikawa T (1978) Decomposition of dynamic team decision problems. *IEEE Trans Autom Control* 23:627–632

Chapter 10

Optimal Control Problems over an Infinite Horizon



In this chapter, we address problems where optimal controls have to be generated and applied over an infinite number of time instants or decision stages. Clearly, the assumption of an *infinite horizon* (IH) is a mathematical abstraction. However, it is very useful to derive optimal controls to be applied on quite a long time horizon.

We shall limit ourselves to considering only problems stated in deterministic contexts. Indeed, whenever stochastic disturbances act on a dynamic system, the analysis is very complicated except for specific cases (for example, when the classical LQG assumptions hold true) and few general results are available in the literature.

Most of these results are related to cases where the state, control, and disturbance variables take on their values from countable sets and the IH cost function entails a discount factor. As the treatment of the IH stochastic context is beyond the scope of this book, we refer the reader to specific classic reference textbooks such as [2, 4].

In Sect. 10.1, the IH deterministic optimal control problem will be addressed. We shall call it Problem C1-IH. Clearly, there are connections with the finite-horizon (FH) problem considered in Chap. 6, that is, with Problem C1. The aims pursued by the two problems are however completely different. Problem C1 consists in passing from one state to another in a finite time, and it is generally called the *maneuver problem*. Instead, in the framework in which Problem C1-IH is stated, first we assume that there is an equilibrium point of the nonlinear dynamic system given by the state space origin associated to the constant control vector $\bar{u} = \mathbf{0}$. Then, we suppose that the state may be taken away from the origin, typically under the action of some random perturbation. The characteristics of the perturbations and the instants in which they may occur are unpredictable. Problem C1-IH requires us to generate an infinite sequence of control vectors so that the state returns to the origin while minimizing an additive cost function (we assume that no cost is paid when the system stays in the origin). Therefore, we have to face a *regulation problem*.

It has now to be noted that Problem C1-IH might be considered just as an extension of Problem C1 to the infinite-horizon case. This means that as soon as the value of the state, taken away from the origin, becomes known, the sequence of the optimal control vectors may be computed online, that is, by solving an optimization problem which we call *Problem C1-IH (open-loop form)*. Of course, this is possible if the time required to solve such a problem is short compared with the time needed to return the state to the equilibrium point. If the computation time is not sufficient, an optimal solution to Problem C1-IH needs to be available for any possible state perturbation. This means that a closed-loop optimal control law, computed offline, has to be available. Then, the related optimization problem must be solved and we call it *Problem C1-IH (closed-loop form)*.

In this chapter, owing to its importance for applications, we are essentially interested in a closed-loop optimal control law, that is, to the solution of Problem C1-IH (closed-loop form). Note that, if the state equation, the transition cost, and the possible constraints on the state and control vectors are time-invariant, this optimal control law is *stationary*. However, unless strong assumptions are introduced, solving both Problem C1-IH (open-loop form) and Problem C1-IH (closed-loop form) is a hard, if not impossible task. Then, we resort to two approximations. The first consists in approximating Problem C1-IH (closed-loop form) by using a *model predictive control* (MPC) approach in which the approximate control action is computed at each time instant or decision stage by a *receding-horizon* (RH) control scheme.

The RH approximation of the IH optimal control law will be addressed in Sect. 10.2. Let us refer to Problem C1-IH (closed-loop form). The RH control scheme can be described as follows. At any time instant $t = 0, 1, \dots$, for a given *finite* horizon length T , and for a given terminal cost, we have to find the optimal FH closed-loop control law that minimizes the process cost for any possible initial state belonging to a given set X . Once the sequence of the T optimal control functions has been derived, the first control function of this sequence becomes the optimal control function of the RH optimal control law. Since the quantities defining the FH optimal control problems are time-invariant, all the optimal control functions (giving rise to the FH optimal control law) of the RH optimal control law are *stationary*.

As the RH optimal control law is exerted over an infinite horizon, the important issue of the stability of the equilibrium point of the resulting feedback system has to be addressed. This issue will be considered in Sect. 10.3, where – general framework of input-to-state stability (ISS) – the concept of *regional Input-to-State Stability* (regional ISS) will be described and used. Specifically, regional ISS makes it possible to characterize the stabilizing properties of the RH control scheme also including the case in which RH approximate control laws are used. Actually, it is worth noting that we are in a general non-LQ context similar to Problems C2 and C3 even if the RH optimal control problem is stated in a deterministic framework. Indeed, the RH optimal control problem has to be solved for *any* initial state belonging to the set X . Therefore, as already discussed in Chaps. 6, 7, and 8, the RH optimal control law may – in principle – be derived in the backward phase of a suitable dynamic programming (DP) procedure. Along the lines of the previous chapters, we pursue an approximate approach consisting in constraining the RH control function to take the

form of a *fixed-structure parametrized* (FSP) function. Provided that some assumptions, related to the regional ISS property are verified, the suboptimal control function, obtained by the extended Ritz method (ERIM), turns out to be robustly stabilizing by resorting to a suitably modified version of the ERIM. Numerical results are presented in Sect. 10.6 that show the effectiveness of the approach based on the ERIM and regional ISS.

10.1 Statement of the Basic Infinite-Horizon Optimal Control Problem

In this section, the IH optimal control problem will be formalized. More specifically, in the next section, we shall consider the deterministic problem while in Sects. 10.1.2 and 10.1.3 the stochastic problem with perfect and with imperfect state information will be addressed, respectively. The stochastic scenarios will be briefly described for the sake of completeness but, as mentioned previously, these scenarios will not be developed any further in the rest of the chapter and we refer the reader to [2, 4].

10.1.1 Deterministic Optimal Control over an Infinite Horizon

Consider the following deterministic time-invariant dynamic system (in general, nonlinear):

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t), \quad t = 0, 1, \dots, \quad (10.1)$$

where $\mathbf{x}_0 = \hat{\mathbf{x}}$ is a given initial state and \mathbf{f} is of class $\mathcal{C}^1(\mathbb{R}^d \times \mathbb{R}^m, \mathbb{R}^d)$. The state vector \mathbf{x}_t may be constrained to belong to a nonempty set X , that is,

$$\mathbf{x}_t \in X \subseteq \mathbb{R}^d, \quad t = 0, 1, \dots. \quad (10.2)$$

The control vector \mathbf{u}_t may be constrained as well to take values in a nonempty set U , i.e.,

$$\mathbf{u}_t \in U \subseteq \mathbb{R}^m, \quad t = 0, 1, \dots. \quad (10.3)$$

In both cases, the origin is assumed to be an internal point of the respective constraining set, which is typically chosen to be compact.

We suppose that the cost function has an additive form given by

$$J = \sum_{t=0}^{+\infty} h(\mathbf{x}_t, \mathbf{u}_t), \quad (10.4)$$

where h is a transition cost of class $\mathcal{C}^1(\mathbb{R}^d \times \mathbb{R}^m, \mathbb{R}^+)$, to be paid at each stage t .

Remark 10.1 The regularity assumptions $f \in \mathcal{C}^1(\mathbb{R}^d \times \mathbb{R}^m, \mathbb{R}^d)$ and $h \in \mathcal{C}^1(\mathbb{R}^d \times \mathbb{R}^m, \mathbb{R}^+)$ on the state transition function f and on the transition cost function h , respectively, restrict a bit the generality of the IH optimal control problem statement, but they will be useful in its subsequent formalization and analysis. \triangleleft

We state the general deterministic IH optimal control problem in the following open-loop form.

Problem C1-IH (open-loop form). *Find the infinite sequence of optimal control vectors $\{\mathbf{u}_0^\circ, \mathbf{u}_1^\circ, \dots\}$ that minimize the cost J , subject to Constraints (10.1), (10.2), and (10.3).* \triangleleft

The following two definitions that will be instrumental for the formalization of the IH optimal control problem and for the subsequent analysis are now introduced.

Definition 10.1 (\mathcal{K} -function) A function $r : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ is of class \mathcal{K} (or a “ \mathcal{K} -function”) if it is continuous, positive definite, and strictly increasing. \triangleleft

Definition 10.2 (\mathcal{K}_∞ -function) A function $r : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ is of class \mathcal{K}_∞ if it is a \mathcal{K} -function and $r(s) \rightarrow +\infty$ as $s \rightarrow +\infty$. \triangleleft

Before getting some insights into the qualitative meaning of Problem C1-IH, let us report the following result from the seminal paper [12] concerning the existence of solutions to Problem C1-IH (open-loop form) and adapted to the setting stated in this chapter.

Theorem 10.1 Consider Constraints (10.1), (10.2), and (10.3) and IH Cost Function (10.4). Suppose that the following assumptions hold:

1. There exist a \mathcal{K}_∞ -function $\varrho^+(\cdot)$ and a control horizon $M^* \geq 1$ such that $\forall \mathbf{x}_0 \in \mathbb{R}^d$, there exists a sequence of control vectors $\{\mathbf{u}_t \in \mathbb{R}^m, t \geq 0\}$ that yield a state trajectory $\{\mathbf{x}_t \in \mathbb{R}^d, t \geq 0\}$ ending in the origin of the state space (i.e., $\mathbf{x}_k = \mathbf{0}$, $k \geq M^*$), such that

$$\sum_{t=0}^{M^*-1} |(\mathbf{x}_t, \mathbf{u}_t)| \leq \varrho^+ (|\mathbf{x}_0|) .$$

2. There exists a \mathcal{K}_∞ -function $r^-(\cdot)$ such that

$$h(\mathbf{x}, \mathbf{u}) \geq r^-(|(\mathbf{x}, \mathbf{u})|), \quad \forall (\mathbf{x}, \mathbf{u}) \in \mathbb{R}^d \times \mathbb{R}^m .$$

Finally, let us denote by $X^{\text{IH}} \subseteq X$ the set of initial states $\bar{\mathbf{x}}_0 \in X$, such that there exists an infinite sequence of control vectors $\{\bar{\mathbf{u}}_t \in U, t \geq 0\}$ yielding a state trajectory $\{\mathbf{x}_t \in X, t \geq 0\}$ for which

$$J = \sum_{t=0}^{+\infty} h(\bar{\mathbf{x}}_t, \bar{\mathbf{u}}_t) < \infty .$$

Then, the following facts hold:

- (a) If Assumption 1 is satisfied, then $\mathbf{0} \in X^{\text{IH}}$.
- (b) If Assumption 1 is satisfied and $X \times U = \mathbb{R}^d \times \mathbb{R}^m$, then $X^{\text{IH}} = \mathbb{R}^d$.
- (c) If also Assumption 2 is satisfied and $\mathbf{x}_0 \in X^{\text{IH}}$, then Problem C1-IH (open-loop form) has an optimal solution.

□

It is worth noting that the results presented in [12] concern a rather more general framework than the setting considered in Theorem 10.1 (for instance, the state equation is time-varying and the state vector is not assumed to be available for measurement). However, in the present chapter, we decided to keep the IH optimal control problem in a slightly less general form because our purpose is to focus on existence conditions for the optimal solutions, to state a general framework suitable to carry out the analysis on other important properties of these solutions (e.g., stationarity and stability), and to lay down a suitable framework for the approximate solutions considered later in the chapter.

Now, before providing additional analysis and properties, let us shed some light on the *qualitative* meaning of Problem C1-IH (open-loop form). In this respect, consider the case in which

$$\mathbf{f}(\mathbf{0}, \mathbf{0}) = \mathbf{0},$$

simply meaning that $\bar{\mathbf{x}} = \mathbf{0}$ is an equilibrium state of Nonlinear System (10.1) associated to the constant input vector $\bar{\mathbf{u}} = \mathbf{0}$. Moreover, assume that

$$h(\mathbf{0}, \mathbf{0}) = 0,$$

i.e., that no cost is paid when the system stays in the equilibrium state $\bar{\mathbf{x}} = \mathbf{0}$, with $\bar{\mathbf{u}} = \mathbf{0}$. Finally, let us suppose, for the moment, that Theorem 10.1(c) applies and hence that (i) Problem C1-IH (open-loop form) has an optimal solution and (ii) the optimal cost corresponding to the sequence of optimal control vectors $\{\mathbf{u}_0^\circ, \mathbf{u}_1^\circ, \dots\}$ is *finite*, that is,

$$J^\circ(\mathbf{x}_0) = \sum_{t=0}^{+\infty} h(\mathbf{x}_t^\circ, \mathbf{u}_t^\circ) < \infty, \quad (10.5)$$

where, owing to State Equation (10.1), the optimal state trajectory is given by

$$\mathbf{x}_{t+1}^\circ = \mathbf{f}(\mathbf{x}_t^\circ, \mathbf{u}_t^\circ), \quad t = 0, 1, \dots, \quad (10.6)$$

with $\mathbf{x}_0^\circ \triangleq \mathbf{x}_0$.

Then, in qualitative terms, Problem C1-IH (open-loop form) can be interpreted as follows. We want to eliminate a certain initial perturbation $\mathbf{x}_0 = \hat{\mathbf{x}}$ of the state, without any constraint on the number of the decision stages, while minimizing a cost function. The rather non-rigorous term “eliminate a certain initial perturbation”

means that there is an equilibrium state which we want the system state to return to. In slightly more precise terms, we face an *optimal regulation problem* (see, for instance, the classical reference [13]), in which we want the state to remain permanently in the origin, and if it is taken away from the origin owing to some cause (typically, the action of some perturbation), we want the state to go back to it while minimizing a given cost function. Therefore, loosely speaking, for the first time in our treatment, we address the *convergence to an equilibrium point*. The previously (temporarily) assumed finiteness of the optimal cost (see (10.5)) is clearly related to the convergence to the equilibrium state. Indeed, the convergence to an equilibrium is related with its *stability* properties. Their analysis is a key issue in the IH case and we shall come back to it later in this section in the context of Problem C1-IH and also in the rest of the chapter when referring to the approximate solutions of this problem.

Now, while still referring to the qualitative framework we just dealt with, we wonder if the formalization of Problem C1-IH (in which the state vector is forced to return to the equilibrium point by using an open-loop control law) is always the most effective approach to eliminate an initial perturbation, or if there are situations where it is more convenient to seek a closed-loop control law, that is, to determine a state-feedback IH optimal control law

$$\boldsymbol{\mu}^\circ \triangleq \{ \boldsymbol{\mu}_0^\circ, \boldsymbol{\mu}_1^\circ, \dots \} \quad (10.7)$$

in the form

$$\boldsymbol{u}_t^\circ = \boldsymbol{\mu}_t^\circ(\boldsymbol{x}_t^\circ), \quad t = 1, 2, \dots,$$

$$\boldsymbol{u}_0^\circ = \boldsymbol{\mu}_0^\circ(\boldsymbol{x}_0),$$

and minimize, for each $\boldsymbol{x}_0 \in X$, the cost functional

$$J_\mu(\boldsymbol{x}_0) = \sum_{t=0}^{+\infty} h[\boldsymbol{x}_t, \boldsymbol{\mu}_t(\boldsymbol{x}_t)]. \quad (10.8)$$

Clearly, determining at each stage t a state-feedback IH optimal control function $\boldsymbol{\mu}_t^\circ$ is not practical. On the other hand, it is worth noting that, compared with the FH case considered in Chap. 6 (see Eqs. (6.1), (6.2), (6.3), (6.4)), in the IH setting considered in this chapter we assume that the dynamic system, the transition cost and the possible constraints on the state and control vectors are *stationary*, that is, the functions f and h and the sets X and U do not depend on the decision stage t (see (10.1), (10.2), (10.3), and (10.4)). Therefore, we are much more interested in determining – if it exists – a *stationary* optimal IH control law

$$\bar{\boldsymbol{\mu}}^\circ \triangleq \{ \boldsymbol{\mu}^\circ, \boldsymbol{\mu}^\circ, \dots \} \quad (10.9)$$

in the form

$$\boldsymbol{u}_t^\circ = \boldsymbol{\mu}^\circ(\boldsymbol{x}_t^\circ), \quad t = 1, 2, \dots,$$

$$\boldsymbol{u}_0^\circ = \boldsymbol{\mu}^\circ(\boldsymbol{x}_0).$$

Thus, it is very important to ascertain whether we can confine the search for an IH optimal control law within the class of stationary control laws of the form (10.9). In this connection, let us consider a generic (not necessarily stationary) IH control law of the form $\boldsymbol{\mu} = \{\boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \dots\}$. Moreover, let \mathcal{M} denote the set of all vector functions $\boldsymbol{\mu} : X \rightarrow \mathbb{R}^m$ such that $\boldsymbol{\mu}(\boldsymbol{x}) \in U$ and $f(\boldsymbol{x}, \boldsymbol{\mu}(\boldsymbol{x})) \in X, \forall \boldsymbol{x} \in X$, and let Π denote the set of all infinite sequences of vector functions $\{\boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \dots\}$ such that $\boldsymbol{\mu}_k \in \mathcal{M}, k = 0, 1, \dots$ (that is, the functions $\boldsymbol{\mu}_k$ are consistent with Constraints (10.2) and (10.3)).

Now, for given $\boldsymbol{x}_0 \in X$ suppose that Problem C1-IH (open-loop form) has an optimal solution (see Theorem 10.1) and consider the corresponding optimal IH cost $J^\circ(\boldsymbol{x}_0)$. If, for some $\boldsymbol{\mu}^* = \{\boldsymbol{\mu}_0^*, \boldsymbol{\mu}_1^*, \dots\} \in \Pi$, we have

$$J_{\boldsymbol{\mu}^*}(\boldsymbol{x}_0) = \sum_{t=0}^{+\infty} h[\boldsymbol{x}_t, \boldsymbol{\mu}_t^*(\boldsymbol{x}_t)] = J^\circ(\boldsymbol{x}_0),$$

then we say that the control law $\boldsymbol{\mu}^*$ is optimal at \boldsymbol{x}_0 . Moreover, if $J_{\boldsymbol{\mu}^*}(\boldsymbol{x}_0) = J^\circ(\boldsymbol{x}_0), \forall \boldsymbol{x}_0 \in X$ we say that $\boldsymbol{\mu}^*$ is an IH optimal control law (not necessarily stationary). The following important general result from [5], regarding the *existence* of a stationary IH optimal control law of the form (10.9), can now be given.

Theorem 10.2 *Suppose that the assumptions in Theorem 10.1 are satisfied and assume that for each $\boldsymbol{x}_0 \in X$ there exists an IH control law that is optimal at \boldsymbol{x}_0 . Then, there exists an optimal stationary IH control law.* \square

Some remarks on Theorem 10.2 are now appropriate. First of all, it is worth noting (see [5]) that Assumption 2 given in Theorem 10.1 also plays a key role toward the existence of optimal stationary IH control laws in Theorem 10.2 (this assumption is related to “Assumption P” in [2]). Indeed, the existence of an optimal stationary IH control law clearly makes Problem C1-IH conceptually and practically much more interesting. However, it is worth stressing that this powerful result strongly depends on the deterministic nature of Problem C1-IH: in a stochastic framework, existence results analogous to Theorem 10.2 do not hold, except for very specialized cases (see [2]). Consequently, we are now allowed to restrict our formulation to the search for stationary optimal control laws, and thus, we formulate Problem C1-IH in the following closed-loop form.

Problem C1-IH (closed-loop form). *Find the stationary IH optimal control law $\bar{\boldsymbol{\mu}}^\circ = \{\boldsymbol{\mu}^\circ, \boldsymbol{\mu}^\circ, \dots\}$ (where $\boldsymbol{u}_t^\circ = \boldsymbol{\mu}^\circ(\boldsymbol{x}_t^\circ), t = 1, 2, \dots, \boldsymbol{u}_0^\circ = \boldsymbol{\mu}^\circ(\boldsymbol{x}_0)$) that minimizes cost J , subject to Constraints (10.1), (10.2), and (10.3).* \triangleleft

The results stated in Theorems 10.1 and 10.2 allow the characterization of the stationary optimal IH control law solving Problem C1-IH (closed-loop form). More specifically, let us suppose that the assumptions stated in Theorem 10.1 hold. Then,

from this theorem, it follows that for any $\mathbf{x}_0 \in X^{\text{IH}}$ there exists an infinite sequence of optimal control vectors

$$\mathcal{U}_{\text{IH}}^{\circ}(\mathbf{x}_0) \triangleq \{\mathbf{u}_0^{\text{IH}^{\circ}}, \mathbf{u}_1^{\text{IH}^{\circ}}, \dots\},$$

that depends on the initial state \mathbf{x}_0 and that solves Problem C1-IH (open-loop form). A stationary IH optimal control law $\bar{\mu}^{\circ} = \{\boldsymbol{\mu}^{\circ}, \boldsymbol{\mu}^{\circ}, \dots\}$ can be defined as follows:

$$\forall \mathbf{x} \in X^{\text{IH}}, \quad \boldsymbol{\mu}^{\circ}(\mathbf{x}) \triangleq \text{first control vector of } \mathcal{U}_{\text{IH}}^{\circ}(\mathbf{x}). \quad (10.10)$$

Let us introduce the following further assumption.

Assumption 10.1 The set X^{IH} introduced in Theorem 10.1 satisfies

$$X^{\text{IH}} = X.$$

□

Thanks to Assumption 10.1 and owing to the *principle of optimality* (see Chap. 6) it can be immediately shown that the control law $\bar{\mu}^{\circ}$ given by Definition (10.10) is a stationary optimal IH control law solving Problem C1-IH (closed-loop form) whose stationary control function will be denoted by $\boldsymbol{\mu}^{\text{IH}^{\circ}}$ in the rest of the chapter. The above-mentioned construction of the control function $\boldsymbol{\mu}^{\text{IH}^{\circ}}$ is schematically shown in Fig. 10.1.

Before making some further qualitative considerations about Problem C1-IH (in both open-loop and closed-loop forms), the following important remark is appropriate.

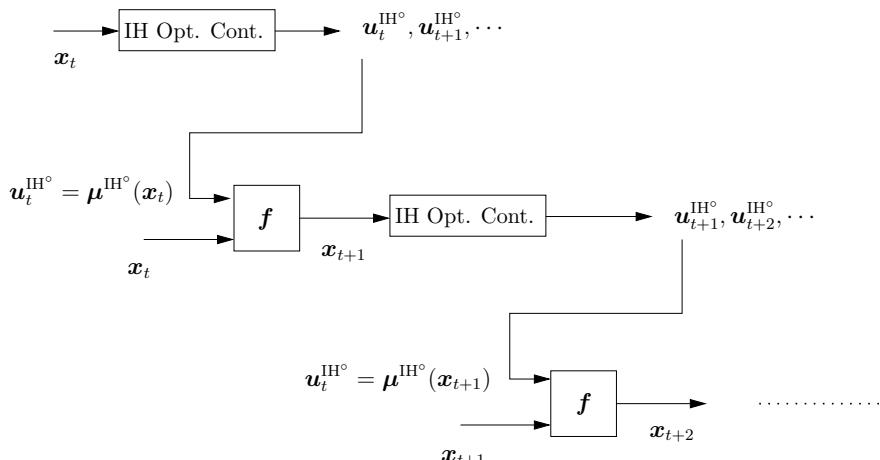


Fig. 10.1 The construction of the optimal stationary IH control law

Remark 10.2 The general statement of Problem C1-IH does not prevent the cost associated with some admissible control law and with some specific initial state to take on an infinite value. Indeed, the solution of Problem C1-IH could very well lead to optimal solutions for which the optimal cost associated with some specific initial states takes on an infinite value (that is, for some non-pathological optimal control law μ^{IH° and for some initial state \bar{x} , it may happen that $J_{\mu^{\text{IH}^\circ}}(\bar{x}) = \infty$). From a *strictly formal* perspective this does not constitute a problem as it is sufficient to extend the set where the cost takes its values to $\mathbb{R}^+ \cup \{\infty\}$, thus preserving the well posedness of the optimization problem (see [2] for an extensive discussion on this issue). However, the possibility of the optimal cost taking on an infinite value is an abstraction and makes the optimal solution of no practical interest (for instance, because it is related to some instability property of the closed-loop system). A typical case in which this scenario can be avoided is the one where the transition cost function h is different from zero except for a suitable cost-free and “absorbing” termination state [2, 12]. For instance, as shown before, this occurs in optimal regulation problems where $h(\bar{x}, \bar{u}) = 0$ with \bar{x} denoting the equilibrium state corresponding to the constant input vector \bar{u} (typically $\bar{x} = 0$ and $\bar{u} = 0$). To gain some more insights on this aspect, we sketch the following simple example (see [2]). \triangleleft

Example 10.1 Consider the scalar deterministic linear system

$$x_{t+1} = a x_t + u_t, \quad t = 0, 1, \dots, \quad (10.11)$$

where $x_t, u_t \in \mathbb{R}$ and $a \in \mathbb{R}$, $a > 0$. The control variable u_t is constrained to take on values in the interval $U = [-1, 1]$, and the cost function takes on the form¹

$$J = \lim_{T \rightarrow \infty} \sum_{t=0}^{T-1} |x_t|.$$

Now, consider the admissible stationary control law $\bar{\mu} = \{\mu, \mu, \dots\}$ where $\mu(x) = 0$, $\forall x \in \mathbb{R}$. Substituting the control law $\bar{\mu}$ and the system equation (10.11) into the cost J , we have

$$J_{\bar{\mu}}(x_0) = \lim_{T \rightarrow \infty} \sum_{t=0}^{T-1} a^t |x_0|.$$

Then

$$J_{\bar{\mu}}(x_0) = \begin{cases} 0 & \text{if } x_0 = 0 \\ \infty & \text{if } x_0 \neq 0 \end{cases}, \quad \text{if } a \geq 1; \quad J_{\bar{\mu}}(x_0) = \frac{|x_0|}{1-a}, \quad \text{if } a < 1.$$

¹This cost function does not satisfy the regularity assumptions introduced in this section. However, the scope of the example is just to show the possibility for the IH optimal cost taking on an infinite value and we preferred to report it anyway because of its simplicity.

Moreover, concerning the optimal cost, it can be shown that, for $a > 1$, $J_{\mu^{\circ}}^{\circ}(x_0) = \infty$ if $|x_0| \geq 1/(a-1)$ and $J_{\mu^{\circ}}^{\circ}(x_0) < \infty$ if $|x_0| < 1/(a-1)$. In this example, the optimal cost takes on an infinite value for $|x_0| \geq 1/(a-1)$ because of the control constraint $u_t \in U$, $t = 0, 1, \dots$ causing the impossibility of forcing the state trajectory to reach the origin for $t \rightarrow \infty$. \triangleleft

The use of the stationary optimal IH control function solving Problem C1-IH (closed-loop form) $\mu^{\text{IH}^{\circ}}$ gives rise to the *closed-loop* system

$$\mathbf{x}_{t+1} = \mathbf{f} [\mathbf{x}_t, \mu^{\text{IH}^{\circ}}(\mathbf{x}_t)], \quad t = 0, 1, \dots \quad (10.12)$$

Remark 10.2 on the possibility of the optimal cost taking on an infinite value for some initial state enhances the basic importance of characterizing the *stability properties* of the origin as an equilibrium state of Closed-Loop System (10.12). To this end, the following, further assumption is needed.

Assumption 10.2 There exists a \mathcal{K}_{∞} -function $r^+(\cdot)$ such that

$$h(\mathbf{x}, \mathbf{u}) \leq r^+(|\mathbf{(x, u)}|), \quad \forall (\mathbf{x}, \mathbf{u}) \in \mathbb{R}^d \times \mathbb{R}^m.$$

\triangleleft

Then, the following very important result can be reported from [12].

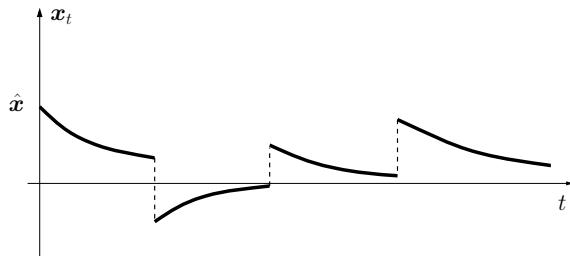
Theorem 10.3 Suppose that the assumptions in Theorem 10.1 are satisfied. Moreover, suppose also that Assumptions 10.1 and 10.2 hold. Then, $\mathbf{x} = \mathbf{0}$ is the unique equilibrium state of Closed-Loop System (10.12) and it is asymptotically stable. \square

A comment can be made similar to the one made after Theorem 10.1. The stability result for the IH optimal control problem reported in [12] is more general in that, for instance, it also addresses the issue of global stability. Nevertheless, our purpose here is again just to put into evidence the nature of conditions implying a stabilizing feature of the IH optimal control law.

Let us now go back to Problem C1-IH (in both open-loop and closed-loop forms). Clearly, the optimal state trajectory $\{\mathbf{x}_t^{\circ}, t = 0, 1, \dots\}$ and the optimal control trajectory $\{\mathbf{u}_t^{\circ}, t = 0, 1, \dots\}$ determined by a closed-loop or an open-loop approach, coincide. As the context in which we operate is completely deterministic, it follows that an open-loop control law is completely adequate, and that Problem C1-IH is just the extension of Problem C1 (dealt with in Chap. 6) to the case for the infinite horizon. However, apart from the extension of the control horizon, there are significant differences between the two problems.

Problem C1 refers to a “maneuver problem.” Typically, this problem consists in passing from one state to another in a finite time. The maneuver might have to be designed once forever or, at least, very rarely. Therefore, an offline computation of an open-loop optimal control sequence $\mathbf{u}_0^{\circ}, \dots, \mathbf{u}_{T-1}^{\circ}$ would be sufficient. Over an infinite-time horizon, the operating context is quite different. It is reasonable

Fig. 10.2 A possible compensation scenario for state perturbations



to suppose that sooner or later disturbances occur that take away the state vector from the equilibrium point (see the qualitative scenario depicted in Fig. 10.2). If the computation time is sufficient, solving Problem C1-IH would – in principle – be possible. Instead, if this computation time is not sufficient and there is a need to force the system state to return to the equilibrium point as quickly as possible, the necessity of the immediate availability of an optimal solution to Problem C1-IH *for any possible state perturbation* is evident. In practice, this means that the closed-loop optimal control law (10.7), computed offline, has to be available. This is possible only in a few cases, typically, under LQ hypotheses (linear dynamic system and quadratic cost) and without constraints imposed on the state and control vectors (i.e., $X \times U = \mathbb{R}^d \times \mathbb{R}^m$).

The aforementioned reason, justifying the introduction of a closed-loop control law instead of an open-loop one, is motivated by caution in that, on one hand, we believe that the control framework is deterministic but, on the other hand, we fear that state perturbations of unknown characteristics may occur. Of course, the caution is mandatory and not an option in the case of an unstable equilibrium point. In Sects. 10.2 and 10.3, an approximate stabilizing solution to Problem C1-IH (closed-loop form) in the deterministic framework exploiting a specific form of the ERIM will be presented.

As mentioned at the beginning of this chapter, the solution of IH optimal control problems in a stochastic framework is out of the scope of this book. In Sects. 10.1.2 and 10.1.3, the extensions to the IH scenario of Problems C2 and C3 addressed in Chaps. 7 and 8, respectively, will be briefly described for the sake of completeness.

10.1.2 Stochastic Optimal Control with Perfect State Information over an Infinite Horizon

In the previous section, we have dealt with an IH optimal control problem stated in a deterministic context. We now consider the case in which random disturbances act on the dynamic system. In this section, we assume the possibility of perfectly measuring the state vector x_t (this assumption will be relaxed in Sect. 10.1.3). Clearly, the possibility of perfectly measuring the state vector x_t turns out to be of fundamental

importance because it enables us to design a control system with a closed-loop structure. In this connection, we previously pointed out that the online use of closed-loop optimal control was an option, not a necessity. Specifically, we observed that this option could be adopted, in a precautionary way, to oppose the action of disturbances, which were not foreseen when the problem was formulated. By contrast, we now have to take explicitly into account this presence. Then, as in the FH case addressed in Chap. 7, we consider the discrete-time stochastic dynamic system

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t, \xi_t), \quad t = 0, 1, \dots, \quad (10.13)$$

where $\mathbf{x}_0 = \hat{\mathbf{x}}$ is a given initial state. Analogously to the FH case, here we do not introduce constraint sets like (10.2) on the state vector.

The control vector \mathbf{u}_t may be constrained to take values from a set U , that is

$$\mathbf{u}_t \in U \subseteq \mathbb{R}^m, \quad t = 0, 1, \dots. \quad (10.14)$$

ξ_0, ξ_1, \dots are mutually independent random variables with time-invariant probability density functions.

Unlike the deterministic case, as the state vector is driven by the action of the control vectors and the random disturbances, we cannot foresee its trajectory over time. However, the possibility of perfectly measuring the state provides us with the maximum possible information. Therefore, the most effective way of acting is to generate the control vector \mathbf{u}_t as a function of the current state \mathbf{x}_t at the stage t . Then, we define the IH *control law* μ as the sequence of *closed-loop control functions*

$$\mathbf{u}_t = \mu_t(\mathbf{x}_t), \quad t = 0, 1, \dots, \quad (10.15)$$

that is,

$$\mu \triangleq \{\mu_0, \mu_1, \dots\}.$$

Moreover, a *stationary* IH control law $\bar{\mu}$ is a control law defined as a sequence of control functions not depending on the decision stage t , that is, $\bar{\mu} \triangleq \{\mu, \mu, \dots\}$ where

$$\mathbf{u}_t = \mu(\mathbf{x}_t), \quad t = 0, 1, \dots. \quad (10.16)$$

Again, we choose to design an optimal control law that minimizes a suitable cost functional. However, unlike the FH framework, the definition of this functional is not immediate and several different choices can be found in the literature. A typical one is the following²:

²Under suitable regularity assumptions [2], Cost (10.17) is equivalent to $J_\mu(\mathbf{x}_0) = E_{\xi_0, \xi_1, \dots} \left\{ \sum_{t=0}^{\infty} h[\mathbf{x}_t, \mu_t(\mathbf{x}_t), \xi_t] \right\}$.

$$J_\mu(\mathbf{x}_0) = \lim_{T \rightarrow \infty} \mathbb{E}_{\xi_0, \xi_1, \dots, \xi_{T-1}} \left\{ \sum_{t=0}^{T-1} h [\mathbf{x}_t, \boldsymbol{\mu}_t(\mathbf{x}_t), \xi_t] \right\} \quad (10.17)$$

where the transition cost function h does not depend on t .

Now, as in the deterministic case, we wonder if we are allowed to confine the search for an optimal control law within the class of stationary control laws of the form (10.16). To this end, the following assumption is introduced.

Assumption 10.3 The transition cost function satisfies

$$h(\mathbf{x}_t, \mathbf{u}_t, \xi_t) \geq 0, \quad \forall \mathbf{x}_t, \mathbf{u}_t, \xi_t. \quad (10.18)$$

□

In [2], it is shown that, if the nonnegativity condition (10.18) is satisfied, it follows that, if there exists an optimal stochastic IH control law for each initial state, then there exists an optimal *stationary* IH control law.

Now, we can formally state the stochastic IH optimal control problem as follows.

Problem C2-IH. Find the optimal stationary control law $\bar{\mu}^\circ$ (or the sequence of optimal time-invariant control functions $\mu^\circ(\mathbf{x}_0), \mu^\circ(\mathbf{x}_1), \dots$) that minimizes the functional $J_\mu(\mathbf{x}_0)$ subject to Constraints (10.13) and (10.14) for each $\mathbf{x}_0 \in \mathbb{R}^d$. □

The closed-loop structure of the optimal control system is enhanced in Fig. 10.3.

A few important remarks about the well-posedness and the practical significance of Problem C2-IH are now appropriate.

Remark 10.3 Considerations like the ones reported in Footnote 1 of Chap. 7 and regarding the rigorous definition of the abstract probabilistic framework underlying the statement of Problem C2-IH can be made (the reader is again referred to [4] and the references cited therein). In particular, nontrivial complications arise when the random variables take on values in *uncountable sets*; most of the results on the existence of optimal stationary control laws and most of the algorithms to compute

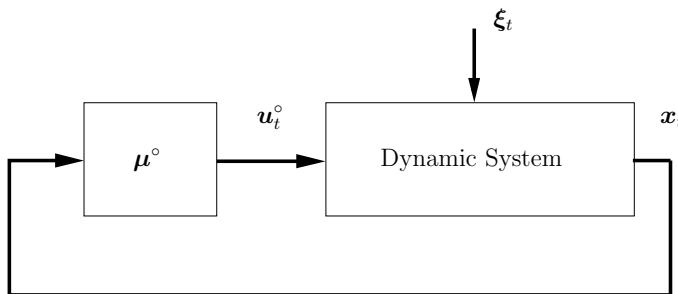


Fig. 10.3 The closed-loop structure of the optimal stationary control law $\bar{\mu}^\circ$

these control laws rely on the assumption that the disturbance ξ_t takes on values in a countable set. Indeed, in several cases, the framework is even more restricted in that it is required that x_t, u_t, ξ_t take values in *finite* sets or that suitable compactness assumptions on the constraint set U are introduced [2]. This restricted context is typically needed in order to show that some DP procedure can be devised to actually compute the optimal stationary control law. \triangleleft

Remark 10.4 Similar to the deterministic case (see Remark 10.2), the general statement of Problem C2-IH does not prevent the cost associated with some admissible control law and the optimal cost taking on an infinite value for some initial state. We noticed that a setting where the possibility of the optimal cost to take on an infinite value for some initial state can be avoided is the one where the transition cost function h is different from zero except for a suitable cost-free and “absorbing” termination state. However, there are several application contexts where the presence of a cost-free absorbing state is not meaningful. In these contexts, a different IH cost can be defined, namely, the so-called *average cost per stage*:

$$\hat{J}_\mu = \lim_{T \rightarrow \infty} \frac{1}{T} \mathbb{E}_{\xi_0, \xi_1, \dots, \xi_{T-1}} \left\{ \sum_{t=0}^{T-1} h[x_t, \mu_t(x_t), \xi_t] \right\}. \quad (10.19)$$

Nevertheless, it should be pointed out that, except for a few specific cases (for example, optimal control of linear systems with quadratic cost functions), the analysis is very complicated and few results are available in the literature, most of them being related to contexts where the state, control, and disturbance spaces are *finite*. Without this assumption, most of these results no longer hold [2]. For instance, it is not necessarily true that the search for an optimal control law can be confined to the class of stationary control laws: examples can be drawn where, in the case of a countable infinite state space, an optimal control law does not exist or there may exist an optimal nonstationary control law but not an optimal stationary one (see the example in [2]). \triangleleft

10.1.3 Stochastic Optimal Control with Imperfect State Information over an Infinite Horizon

In Chap. 8, in the context of the FH stochastic optimal control problem, we considered the more realistic case where the control law has access to a vector y_t , $t = 0, 1, \dots, T - 1$ of measurements that contain only imperfect information on the system state x_t , or may even not be influenced by some of the state components.

Now, we shall extend this problem formulation to the IH case. More specifically, we assume that y_t is a p -dimensional vector of an *observation or measurement channel* of the form

$$\mathbf{y}_t = \mathbf{g}(\mathbf{x}_t, \boldsymbol{\eta}_t), \quad t = 0, 1, \dots, \quad (10.20)$$

where \mathbf{g} is a known function not depending on the decision stage t .

As in the perfect state information case, we suppose that the dynamic system can be described by the state equation

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\xi}_t), \quad t = 0, 1, \dots. \quad (10.21)$$

Unlike Eq. (10.13), the state is observable only through Channel (10.20), and thus, we do not know the initial state \mathbf{x}_0 . Moreover, $\mathbf{x}_0, \boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \dots, \boldsymbol{\eta}_0, \boldsymbol{\eta}_1, \dots$ are mutually independent random variables characterized as in Problem C3. However, the probability density functions of $\boldsymbol{\xi}_i$ and $\boldsymbol{\eta}_i$ are now time-invariant.

The control vector \mathbf{u}_t may be constrained to take values from a set U , that is,

$$\mathbf{u}_t \in U \subseteq \mathbb{R}^m, \quad t = 0, 1, \dots. \quad (10.22)$$

whereas in the perfect state information case, the state vector \mathbf{x}_t “summarized the whole history of the system” up to stage t and contained all necessary information to make a decision \mathbf{u}_t , now \mathbf{x}_t is not available anymore. As in Chap. 8, by “history of the system” we mean all the measures taken up to the current stage t and all the control vectors applied up to the stage $t - 1$. Then, we define the *information vector* as (compare with (8.2))

$$\begin{aligned} \mathbf{I}_0 &\triangleq \mathbf{y}_0, \\ \mathbf{I}_t &\triangleq \text{col}(\mathbf{y}_0, \dots, \mathbf{y}_t, \mathbf{u}_0, \dots, \mathbf{u}_{t-1}), \quad t = 1, 2, \dots \end{aligned} \quad (10.23)$$

and the IH *control law* μ is defined as the sequence of *closed-loop control functions*

$$\mathbf{u}_t = \boldsymbol{\mu}_t(\mathbf{I}_t), \quad t = 0, 1, \dots,$$

that is

$$\mu \triangleq \{ \boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \dots \}.$$

Unlike the perfect state information case, the dimension of the vector \mathbf{I}_t , which is the argument of the control function $\boldsymbol{\mu}_t$, is growing as the number of decision stages increases. Clearly, the increasing dimension of the information vector with time rules out the possibility of looking for an optimal stationary control law. Therefore, some type of approximation is needed. For example, we may resort to the same approximation proposed in Sect. 8.4.1, where we defined the *limited-memory information vector*. This vector is now given by (see (8.2))

$$\mathbf{I}_t^M \triangleq \text{col}(\mathbf{y}_{t-M+1}, \dots, \mathbf{y}_t, \mathbf{u}_{t-M+1}, \dots, \mathbf{u}_{t-1}), \quad t = M, M + 1, \dots,$$

where M denotes again the “sliding window” of data that the decision-maker (DM) stores in its memory. To mitigate the abrupt truncation of the information vector \mathbf{I}_t , in Sect. 8.4.1 we introduced the *memory vector* \mathbf{s}_t by means of which the DM can maintain some form of memory of the measures and controls that have been lost at stage $t - M$. We do not consider the memory vector for simplicity as this section has only a very qualitative meaning. Of course, for stages $t = 0, 1, \dots, M - 1$, the DM is able to recall the information vector \mathbf{I}_t .

The time-invariant dimension of the vector \mathbf{I}_t^M enables us to address control functions of the form

$$\mathbf{u}_t = \boldsymbol{\mu}_t(\mathbf{I}_t), \quad t = 0, 1, \dots, M - 1, \quad (10.24a)$$

$$\mathbf{u}_t = \boldsymbol{\mu}_t(\mathbf{I}_t^M), \quad t = M, M + 1, \dots. \quad (10.24b)$$

It follows that Control Functions 10.24b no longer have arguments with time-increasing dimension. Now, a cost similar to (10.19) is considered, that is,

$$\begin{aligned} \hat{J}_\mu &= \underset{x_0, \xi_0, \dots, \xi_{M-1}, \eta_0, \dots, \eta_{M-1}}{\text{E}} \frac{1}{M} \left\{ \sum_{t=0}^{M-1} h [x_t, \boldsymbol{\mu}_t(\mathbf{I}_t), \xi_t] \right\} \\ &\quad + \lim_{T \rightarrow \infty} \frac{1}{T} \underset{x_0, \xi_0, \dots, \xi_{T-1}, \eta_0, \dots, \eta_{T-1}}{\text{E}} \left\{ \sum_{t=M}^{T-1} h [x_t, \boldsymbol{\mu}_t(\mathbf{I}_t), \xi_t] \right\}. \end{aligned}$$

Now, the imperfect state information IH optimal control problem can be formally stated as follows.

Problem C3-IH. *Find the optimal control law μ° (or the sequence of optimal control functions $\boldsymbol{\mu}_0^\circ(\mathbf{I}_0), \boldsymbol{\mu}_1^\circ(\mathbf{I}_1), \dots, \boldsymbol{\mu}_{M-1}^\circ(\mathbf{I}_{M-1}), \boldsymbol{\mu}_t^\circ(\mathbf{I}_t^M)$, $t = M, M + 1, \dots$) that minimizes the functional \hat{J}_μ subject to Constraints (10.20), (10.21), and (10.22).* \triangleleft

Note that the limited-memory approximation enables Problem C3-IH to provide optimal control functions which, after stage $t = M$, depend on a vector of time-invariant dimension. However, the approximate optimal control functions remain time-varying. Perhaps as the number of decision stages increases the optimal control could tend to become stationary but studying this possibility is too difficult. It is interesting to look at the example in Sect. 8.6.2, where the optimal control of a freeway traffic is reported (see Fig. 8.10a, b). The dynamic system is strongly nonlinear, constraints are present, and the measurement channel is affected by non-Gaussian noises. Owing to the rather large number of decision stages, the control law is based on both a limited-memory information vector and a memory vector. The finite-horizon optimal control problem is solved by the ERIM that provides an approximate optimal control law driving the state to a neighborhood of an equilibrium point starting from a variety of initial states. Such results may be interpreted as a sort of stabilizing property.

Summing up, owing to the very general assumptions under which Problems C1-IH, C2-IH, and C3-IH have been stated, they can be solved analytically only in very few cases, typically in the linear-quadratic-Gaussian setting. Moreover, referring to Problem C3-IH, in addition to the difficulties also arising in the context of the solution in the FH case, the major additional difficulty turns out to be the previously mentioned growing dimension of the information vector \mathbf{I}_t , as the number of decision stages increases. Therefore, when addressing the solution of Problems C1-IH, C2-IH, and C3-IH, suboptimal solution methodologies have to be devised. As already mentioned, in the rest of this chapter only the approximate solution of Problem C1-IH is addressed.

10.2 From Finite to Infinite Horizon: The Deterministic Receding-Horizon Approximation

As stated in Sect. 10.1, unless suitable conditions hold (e.g., Dynamic System (10.1) is linear, Cost (10.4) is quadratic, and no constraints are imposed on the state and control vectors, i.e., $X \times U = \mathbb{R}^d \times \mathbb{R}^m$), deriving the optimal IH control law $\bar{\mu}^\circ$ given by Definition (10.10) and characterized by the optimal IH control function μ^{IH° is a very hard, if not impossible task.³

Then, we resort to a different approach leading toward *approximate stationary IH control laws* – namely, the MPC paradigm in which an RH approximation of the optimal IH control law μ^{IH° is used.

Nowadays, the MPC control scheme is the most successful multivariable control approach in industrial applications owing to its properties in terms of optimization of a complex cost functional for nonlinear and possibly large-scale systems while simultaneously satisfying state and control constraints. A detailed treatment of MPC is out of the scope of this book and the reader is referred to the vast literature on the subject (see, for instance, the very recent book [30] and the well-known survey papers [3, 8, 20, 28] as well as the references cited therein).

In this chapter, the MPC paradigm and the RH implementation of the control scheme are of instrumental interest because they provide a useful and practical framework to design stationary control laws approximating the optimal IH control law μ^{IH° along the line of the seminal paper [12] while preserving the IH optimal control stabilizing properties illustrated in Sect. 10.1. Moreover, this paradigm paves the way to the implementation of the RH approximate control law by the ERIM.

A basic MPC control scheme for discrete-time nonlinear dynamic systems of the form (10.1) giving rise to an RH control scheme can be conceptually described as shown in Fig. 10.4.

Specifically, when the controlled system is in the state \mathbf{x}_t at stage t , a suitable FH T -stage optimal control problem is solved (see Problem C1 stated in Chap. 6), thus

³From now on, for the sake of notational simplicity and with some abuse of notation, any stationary control law $\bar{\mu} = \{\mu, \mu, \dots\}$ will simply be denoted using the control function μ .

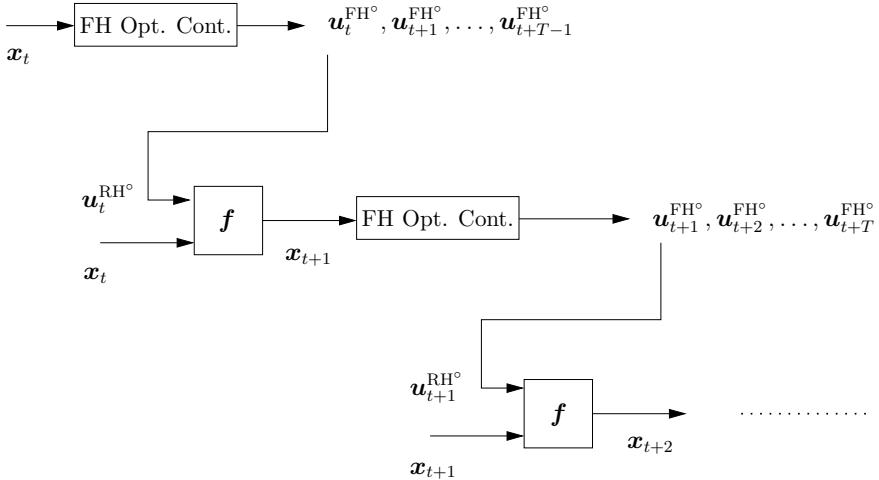


Fig. 10.4 The RH control scheme in the MPC paradigm. At each time instant t , an FH optimal control problem is solved and the first optimal control action is applied; at the subsequent time instant $t + 1$, the procedure is repeated, and so on

the sequence of optimal control vectors (dependent on the “initial” state \mathbf{x}_t)

$$\mathcal{U}_{\text{FH}}^{\circ}(\mathbf{x}_t) \triangleq \{ \mathbf{u}_t^{\text{FH}^\circ}, \dots, \mathbf{u}_{t+T-1}^{\text{FH}^\circ} \}$$

is derived, and the *first* control vector of this sequence becomes the control action $\mathbf{u}_t^{\text{RH}^\circ}$ generated by the RH control scheme at stage t , that is

$$\mathbf{u}_t^{\text{RH}^\circ} \triangleq \mathbf{u}_t^{\text{FH}^\circ} = \text{first control vector of } \mathcal{U}_{\text{FH}}^{\circ}(\mathbf{x}_t).$$

This procedure is repeated stage after stage and a *feedback* control law is *implicitly* obtained, as the RH control vector $\mathbf{u}_t^{\text{RH}^\circ}$ depends on the current state \mathbf{x}_t .

It is worth noting the similarity between the two schemes shown in Figs. 10.1 and 10.4. This similarity should not come as a surprise. In fact, as illustrated in [12], the RH control scheme is the natural candidate for approximating stationary optimal IH control laws.

Let us now state the RH optimal control problem in more precise terms. We consider the same setting considered in Sect. 10.1 to state Problem C1-IH (State Equation (10.1) and Constraints (10.2), (10.3)) except for the cost that now takes on the form

$$J^{\text{FH}}(\mathbf{x}_t, \mathbf{u}_t, \mathbf{u}_{t+1}, \dots, \mathbf{u}_{t+T-1}, T) = \sum_{i=t}^{t+T-1} h(\mathbf{x}_i, \mathbf{u}_i) + h_T(\mathbf{x}_{t+T}) \quad t = 0, 1, \dots, \quad (10.25)$$

where, as in Cost (10.4), $h \in \mathcal{C}^1(\mathbb{R}^d \times \mathbb{R}^m, \mathbb{R}^+)$ is a transition cost function to be paid at each stage t , $h_T \in \mathcal{C}^1(\mathbb{R}^d, \mathbb{R}^+)$ is a *terminal cost function* with $h_T(\mathbf{0}) = 0$, and T is a positive integer denoting the length of the control horizon. Moreover, we suppose that all the assumptions in Theorems 10.1, 10.2, and 10.3 are satisfied.

Remark 10.5 As shown in [12], the introduction of the terminal cost function h_T into Cost (10.4) is key to obtaining several important results concerning the approximating properties of the RH control scheme with respect to the optimal IH one and to preserve a sort of stabilizing property. A more detailed characterization of h_T will be given later on in this chapter in the context of the closed-loop stability under the action of the RH optimal control law and its approximate implementations through the ERIM. \triangleleft

Now, the following FH optimal control problem can be stated.

Problem C1* (open-loop form). At every time instant $t \geq 0$, for a given horizon length T , for given transition and terminal cost functions h, h_T , for a given terminal set $X_f \subset X$, and for a given initial state $\mathbf{x}_t \in X$, find the optimal FH control sequence $\mathcal{U}_{\text{FH}}^\circ(\mathbf{x}_t) = \{\mathbf{u}_t^{\text{FH}^\circ}, \dots, \mathbf{u}_{t+T-1}^{\text{FH}^\circ}\}$ that minimizes Cost (10.25) subject to:

1. System Dynamics (10.1) with \mathbf{x}_t as initial state;

2. Constraints (10.2) and (10.3), that is $\mathbf{x}_i \in X, \mathbf{u}_i^{\text{FH}} \in U, i = t, \dots, t + T - 1$;

3. the terminal state constraint $\mathbf{x}_{t+T} \in X_f$. \triangleleft

It is worth noting that Problem C1* (open-loop form) has been stated in terms of computing the optimal FH control sequence $\mathcal{U}_{\text{FH}}^\circ(\mathbf{x}_t)$ consistently with the well-established literature on MPC control (see [30] and the survey papers [8, 20, 28] for details). However, along the same lines as Sect. 10.1.1 as far as the IH optimal control problem is concerned, this FH problem can be equivalently stated in closed-loop form as follows. Let us look for an FH optimal control law

$$\boldsymbol{\mu}^{\text{FH}^\circ} \triangleq \{\boldsymbol{\mu}_i^{\text{FH}^\circ}(\mathbf{x}_i), i = t, \dots, t + T - 1\}$$

providing the optimal solution to Problem C1* (open-loop form) in feedback form. More precisely, let us rewrite Cost (10.25) as

$$J_\mu(\mathbf{x}_t, \boldsymbol{\mu}_t, \boldsymbol{\mu}_{t+1}, \dots, \boldsymbol{\mu}_{t+T-1}, T) = \sum_{i=t}^{t+T-1} h[\mathbf{x}_i, \boldsymbol{\mu}_i^{\text{FH}}(\mathbf{x}_i)] + h_T(\mathbf{x}_{t+T}) \quad t = 0, 1, \dots \quad (10.26)$$

Now, we are able to reformulate Problem C1* (open-loop form) in closed-loop form.

Problem C1* (closed-loop form). At every time instant $t \geq 0$, for a given horizon length T , for given transition and terminal cost functions h, h_T , and for a given terminal set $X_f \subset X$, find the optimal FH control law $\boldsymbol{\mu}^{\text{FH}^\circ} \triangleq \{\boldsymbol{\mu}_i^{\text{FH}^\circ}(\mathbf{x}_i), i = t, \dots, t + T - 1\}$ that minimizes Cost (10.26) for every possible initial state $\mathbf{x}_t \in X$ subject to:

1. System Dynamics (10.1) with \mathbf{x}_t as initial state;
2. Constraints (10.2) and (10.3), that is $f(\mathbf{x}_i, \mu_i^{\text{FH}}(\mathbf{x}_i)) \in X$ and $\mu_i^{\text{FH}}(\mathbf{x}_i) \in U$, $\forall \mathbf{x}_i \in X_{i,\mu^{\text{FH}}}$, where $X_{i,\mu^{\text{FH}}} \subseteq X$ is the set of states reachable at time i from any initial state $\mathbf{x}_t \in X$ by applying the control functions $\mu_i^{\text{FH}}, i = t, \dots, t+T-1$;
3. the terminal state constraint $f(\mathbf{x}_{t+T-1}, \mu_{t+T-1}^{\text{FH}}(\mathbf{x}_{t+T-1})) \in X_f$, which has to be satisfied $\forall \mathbf{x}_{t+T-1} \in X_{t+T-1,\mu^{\text{FH}}}$.

△

Because of the time invariance of Dynamic System (10.1), of Cost Function (10.26), and of the constraint sets X, U, X_f , instead of $\mu_i^{\text{FH}}(\mathbf{x}_i)$, we consider the control functions

$$\mu_{i-t}^{\text{FH}^\circ}(\mathbf{x}_i), \quad t \geq 0, \quad i = t, \dots, t+T-1.$$

Thus, it follows immediately that the RH optimal control law is *stationary* and its control function is given by

$$\mu^{\text{RH}^\circ}(\mathbf{x}_t) \triangleq \mu_0^{\text{FH}^\circ}(\mathbf{x}_t), \quad \forall \mathbf{x}_t \in X, \quad t \geq 0.$$

Hence, we are able to state the following problem.

Problem C1-RH. Find the RH optimal control law μ^{RH° , whose optimal control function is the first function of the sequence $\mu_0^{\text{FH}^\circ}, \dots, \mu_{T-1}^{\text{FH}^\circ}$ solving Problem C1' (closed-loop form) for $t = 0$. △

10.3 Stabilizing Properties of Deterministic RH Control Laws

As we are dealing with control problems over an infinite number of decision stages, the issue of the stability of the equilibrium of the feedback system when the IH control law is exerted has to be addressed. In Sect. 10.1.1, the analysis of the stability of the equilibrium under the action of the optimal IH control law has been presented (see Theorem 10.3). Then, in Sect. 10.2, the RH approximation to the IH optimal control problem has been illustrated. In the present section, we shall address the stabilizing properties of the RH control law also providing the main notation and background on useful regional ISS results while in Sect. 10.4, the analysis of the stabilizing properties of the approximate RH control law obtained by the ERIM will be presented. More specifically, since the stability properties have to deal with the presence of approximation errors, the analysis is carried out by resorting to the concept of regional ISS [7, 18] that makes it possible to consider available results on RH control in a natural way and also to include in the analysis the case where RH approximate control laws are used (e.g., when FSP functions are considered for the practical implementation of the RH control law).

Let us first introduce the main notation and definitions⁴ used in the sequel. We denote as $\boldsymbol{\varsigma}_{[t \geq 0]}$ the infinite-length discrete-time sequence $\boldsymbol{\varsigma}_{[t \geq 0]} \triangleq \{\boldsymbol{\varsigma}_t \in \mathbb{R}^r, t = 0, 1, \dots\}$ and with $\boldsymbol{\varsigma}_{[t]}$ the finite-length subsequence up to time instant t extracted from $\boldsymbol{\varsigma}_{[t \geq 0]}$, and

$$\boldsymbol{\varsigma}_{[t]} \triangleq \{\boldsymbol{\varsigma}_k \in \mathbb{R}^r, k = 0, 1, \dots, t\}.$$

Moreover, for any sequence $\boldsymbol{\varsigma}_{[t \geq 0]}$, we let

$$\|\boldsymbol{\varsigma}_{[t \geq 0]}\| \triangleq \sup_{k \geq 0} \{|\boldsymbol{\varsigma}_k|\}$$

and, analogously,

$$\|\boldsymbol{\varsigma}_{[t]}\| \triangleq \sup_{0 \leq k \leq t} \{|\boldsymbol{\varsigma}_k|\}.$$

The set of discrete-time sequences $\boldsymbol{\varsigma}_{[t \geq 0]}$ taking values in some subset $\Phi \subset \mathbb{R}^r$ is denoted by Σ_Φ . We let

$$\Phi^{sup} \triangleq \sup_{\boldsymbol{\varsigma}_{[t \geq 0]} \in \Sigma_\Phi} \{\|\boldsymbol{\varsigma}_{[t \geq 0]}\|\}.$$

The symbol *id* represents the *identity function* from \mathbb{R} to \mathbb{R} , while $\gamma_1 \circ \gamma_2$ is the composition of two functions γ_1 and γ_2 from \mathbb{R} to \mathbb{R} . Given a set $A \subseteq \mathbb{R}^d$, $\text{int}(A)$ denotes the interior of A . Given a vector $\mathbf{x} \in \mathbb{R}^d$,

$$d(\mathbf{x}, A) \triangleq \inf \{|\tilde{\mathbf{x}} - \mathbf{x}|, \tilde{\mathbf{x}} \in A\}$$

is the point-to-set distance from $\mathbf{x} \in \mathbb{R}^d$ to A . Given two sets $A \subseteq \mathbb{R}^d$, $B \subseteq \mathbb{R}^d$,

$$\text{dist}(A, B) \triangleq \inf \{d(\mathbf{x}, A), \mathbf{x} \in B\}$$

is the minimal set-to-set distance. The difference between two given sets $A \subseteq \mathbb{R}^d$ and $B \subseteq \mathbb{R}^d$, with $B \subseteq A$, is denoted as $A \setminus B$ while, given two sets $A \subseteq \mathbb{R}^d$, $B \subseteq \mathbb{R}^d$, the *Pontryagin difference set* C is defined as

$$C = A \sim B \triangleq \{\mathbf{x} \in \mathbb{R}^d : \mathbf{x} + \tilde{\mathbf{x}} \in A, \forall \tilde{\mathbf{x}} \in B\},$$

while the *Minkowski sum set* is defined as

$$S = A \oplus B \triangleq \{\mathbf{x} \in \mathbb{R}^d : \exists \tilde{\mathbf{x}} \in A \text{ and } \exists \bar{\mathbf{x}} \in B \text{ for which } \mathbf{x} = \tilde{\mathbf{x}} + \bar{\mathbf{x}}\}.$$

Given a vector $\boldsymbol{\eta} \in \mathbb{R}^p$ and a positive scalar ρ , the closed ball in \mathbb{R}^p centered in $\boldsymbol{\eta}$ and of radius ρ , is denoted as

⁴The notation and definitions introduced in this section are fairly standard in the literature (see, for instance, [11]).

$$\mathcal{B}^p(\eta, \rho) \triangleq \{\tilde{\eta} \in \mathbb{R}^p : |\tilde{\eta} - \eta| \leq \rho\}.$$

The shorthand $\mathcal{B}^p(\rho)$ is used when the ball is centered in the origin. The *domain* of a generic function $\kappa : \text{dom}(\kappa) \subseteq \mathbb{R}^a \rightarrow \mathbb{R}^b$ is denoted as $\text{dom}(\kappa)$. Moreover, we recall the following well-known further definition (compare with Definition 10.1).

Definition 10.3 (\mathcal{KL} -function) A function $\beta : \mathbb{R}^+ \times \mathbb{Z}^+ \rightarrow \mathbb{R}^+$ is of class \mathcal{KL} , if, for each fixed $t \geq 0$, $\beta(\cdot, t)$ is of class \mathcal{K} , for each fixed $s \geq 0$, $\beta(s, \cdot)$ is decreasing, and $\beta(s, t) \rightarrow 0$ as $t \rightarrow \infty$. \triangleleft

Let us now introduce the stability framework that will be used in this chapter. To this end, we consider the following discrete-time dynamic system:

$$\mathbf{x}_{t+1} = f^{(0)}(\mathbf{x}_t, \boldsymbol{\varsigma}_t), \quad t \geq 0, \quad (10.27)$$

with $\mathbf{x}_0 = \bar{\mathbf{x}}$, $f^{(0)}(\mathbf{0}, \mathbf{0}) = \mathbf{0}$ and where $\mathbf{x}_t \in \mathbb{R}^d$ denotes the state vector. System (10.27) has no control input but its dynamics is affected by a bounded exogenous variable $\boldsymbol{\varsigma}_t \in \Phi \subset \mathbb{R}^r$. Given the initial state $\bar{\mathbf{x}}$, the exogenous sequence $\boldsymbol{\varsigma}_{[t \geq 0]} \in \Sigma_\Phi$ yields the discrete-time state trajectory $\{\mathbf{x}(\bar{\mathbf{x}}, \boldsymbol{\varsigma}_{[t]}, t), t \geq 0\}$ and hence $\mathbf{x}_t = \mathbf{x}(\bar{\mathbf{x}}, \boldsymbol{\varsigma}_{[t]}, t)$.

Remark 10.6 It is worth noting that System (10.27) is introduced in order to establish a mathematical setting for the stability analysis that allows the taking into account of the effects induced on the closed-loop dynamics by the unavoidable approximation errors associated with the implementation of the approximate RH control law. \triangleleft

Further definitions and concepts are needed.

Definition 10.4 (*RPI set*) A set $\Upsilon \subset \mathbb{R}^d$ is a robust positively invariant (RPI) set for System (10.27) if $f^{(0)}(\mathbf{x}, \boldsymbol{\varsigma}) \in \Upsilon$, $\forall \mathbf{x} \in \Upsilon$ and $\forall \boldsymbol{\varsigma} \in \Phi$. \triangleleft

The regional ISS property is now recalled [18]. To this end, let us first introduce a useful class of Lyapunov functions.

Definition 10.5 (*ISS-Lyapunov Function*) Given System (10.27) and a pair of compact sets $\Upsilon \subset \mathbb{R}^d$ and $\Omega \subseteq \Upsilon$, with $\{\mathbf{0}\} \subset \Omega$, a function $V : \mathbb{R}^d \rightarrow \mathbb{R}^+$ is called a (regional) ISS-Lyapunov function in Υ , if there exist some \mathcal{K}_∞ -functions $\alpha_1, \alpha_2, \alpha_3$, and a \mathcal{K} -function σ such that:

1. The following inequalities hold $\forall \boldsymbol{\varsigma}_{[t \geq 0]} \in \Sigma_\Phi$:

$$V(\mathbf{x}) \geq \alpha_1(|\mathbf{x}|), \quad \forall \mathbf{x} \in \Upsilon, \quad (10.28)$$

$$V(\mathbf{x}) \leq \alpha_2(|\mathbf{x}|), \quad \forall \mathbf{x} \in \Omega, \quad (10.29)$$

$$V(f^{(0)}(\mathbf{x}, \boldsymbol{\varsigma})) - V(\mathbf{x}) \leq -\alpha_3(|\mathbf{x}|) + \sigma(|\boldsymbol{\varsigma}|), \quad \forall \mathbf{x} \in \Upsilon; \quad (10.30)$$

2. There exists a suitable \mathcal{K}_∞ -function ρ (with ρ such that $(id - \rho)$ is a \mathcal{K}_∞ -function, too) such that, introducing the compact set

$$\Theta \triangleq \{\mathbf{x} : V(\mathbf{x}) \leq b(\Phi^{sup})\},$$

with $b(s) \triangleq \alpha_4^{-1} \circ \rho^{-1} \circ \sigma(s)$, $\alpha_4 \triangleq \alpha_3 \circ \alpha_2^{-1}$, the following inclusion is verified:

$$\{\mathbf{x} : \mathbf{x} \in \Theta, d(\mathbf{x}, \partial\Omega) > c\} \subset \Omega,$$

for some suitable constant $c \in \mathbb{R}^+$.

□

Further to the introduction of the concept of an ISS-Lyapunov function, we now introduce the definitions of regional ISS and *regional input-to-state practical stability* (ISpS). The ISpS will be key to establishing closed-loop stability properties under the action of approximate RH control laws. In this respect, the concept of ISpS generalizes the practical stability characterization of nonlinear MPC used in the seminal paper [21] and exploited for the first time in [24] and subsequently in [23] in the context of the stability analysis of RH approximate control laws implemented by OHL networks.

Definition 10.6 (*Regional ISS and ISpS*) Consider a given compact set $\Upsilon \subset \mathbb{R}^d$. If Υ is RPI for (10.27) and if there exist a \mathcal{KL} -function β , a \mathcal{K} -function γ , and a positive number $\eta \in \mathbb{R}^+$ such that

$$|\mathbf{x}(\bar{\mathbf{x}}, \boldsymbol{\varsigma}_{[t]}, t)| \leq \max \{\beta(|\bar{\mathbf{x}}|, t), \gamma(\|\boldsymbol{\varsigma}_{[t]}\|)\} + \eta, \quad \forall t \geq 0, \forall \bar{\mathbf{x}} \in \Upsilon, \quad (10.31)$$

then:

1. If $\{\mathbf{0}\} \in \Upsilon$ and Inequality (10.31) is satisfied for $\eta = 0$, then System (10.27) is said to be regional input-to-state stable (ISS) for initial states belonging to Υ .
2. System (10.27), with $\boldsymbol{\varsigma}_{[t \geq 0]} \in \Sigma_\phi$, is said to be regional input-to-state practically stable (ISpS) in Υ .

□

The following important result from [18] can be stated.

Theorem 10.4 (*Regional ISS*) *If System (10.27) admits an ISS-Lyapunov function in Υ , and Υ is RPI for (10.27), then:*

1. *System (10.27) is regional ISS in Υ .*
2. *All state trajectories starting in Υ converge to a neighborhood of the origin, that is,*

$$\lim_{t \rightarrow \infty} d(\mathbf{x}(\bar{\mathbf{x}}, \boldsymbol{\varsigma}_{[t]}, t), \Theta) = 0, \quad \forall \bar{\mathbf{x}} \in \Upsilon.$$

□

As mentioned before, Theorem 10.4 is key to characterizing the stability properties in the presence of approximation errors within the implementation of the RH control law. In this connection, we now need to address the stability analysis in a slightly more general framework than the one considered in Sect. 10.2. This generalization will enable us to quantify the effects of the above approximation errors on the corresponding perturbed state trajectories. This framework is the one introduced with reference to System (10.27) (also recall Remark 10.6). More specifically, we include the control input \mathbf{u}_t in System (10.27), thus getting

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\varsigma}_t), \quad t = 0, 1, \dots, \quad (10.32)$$

where $\boldsymbol{\varsigma}_{[t \geq 0]}$ is the discrete-time sequence denoting the *bounded* exogenous input already considered in System (10.27). Again, we assume that the origin is an equilibrium state for $\mathbf{u}_t = \mathbf{0}$, $\boldsymbol{\varsigma}_t = \mathbf{0}$, $\forall t \geq 0$, that is,

$$\mathbf{f}(\mathbf{0}, \mathbf{0}, \mathbf{0}) = \mathbf{0}. \quad (10.33)$$

The state and control variables are subject to Constraints (10.2) and (10.3).

Given State Equation (10.32), let $\widehat{\mathbf{f}}(\mathbf{x}_t, \mathbf{u}_t)$, with $\widehat{\mathbf{f}}(\mathbf{0}, \mathbf{0}) = \mathbf{0}$, denote the *nominal* model used for the purpose of designing the RH control law, such that

$$\mathbf{x}_{t+1} = \widehat{\mathbf{f}}(\mathbf{x}_t, \mathbf{u}_t) + \delta_t^f, \quad t = 0, 1, \dots, \quad (10.34)$$

where $\mathbf{x}_0 = \bar{\mathbf{x}}$ and where $\delta_t^f \triangleq \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\varsigma}_t) - \widehat{\mathbf{f}}(\mathbf{x}_t, \mathbf{u}_t) \in \mathbb{R}^d$ denotes the so-called discrete-time state transition uncertainty (see [26]).

Let us introduce the following further definition.

Definition 10.7 ($\mathcal{C}_1(\Upsilon)$) Given a set $\Upsilon \subset \mathbb{R}^d$, the (one-step) controllability set to Υ , denoted as $\mathcal{C}_1(\Upsilon)$, is defined as

$$\mathcal{C}_1(\Upsilon) \triangleq \{\mathbf{x} \in \mathbb{R}^d : \exists \mathbf{u} \in U \text{ such that } \widehat{\mathbf{f}}(\mathbf{x}, \mathbf{u}) \in \Upsilon\},$$

i.e., $\mathcal{C}_1(\Upsilon)$ is the set of state vectors that can be steered to Υ in one time step by a control action under the map $\widehat{\mathbf{f}}(\mathbf{x}, \mathbf{u})$. \triangleleft

From now on, the following regularity assumptions on the state-dynamics map will be needed.

Assumption 10.4 The function $\widehat{\mathbf{f}} : X \times U \rightarrow X$ is Lipschitz continuous with respect to $\mathbf{x} \in X$, with Lipschitz constant $L_{\widehat{\mathbf{f}}_x} \in \mathbb{R}^+$, that is, for any $\mathbf{u} \in U$, one has

$$|\widehat{\mathbf{f}}(\mathbf{x}, \mathbf{u}) - \widehat{\mathbf{f}}(\mathbf{x}', \mathbf{u})| \leq L_{\widehat{\mathbf{f}}_x} |\mathbf{x} - \mathbf{x}'|, \quad \forall \mathbf{x}, \mathbf{x}' \in X.$$

Furthermore, the function $\widehat{\mathbf{f}}$ is uniformly continuous in \mathbf{u} , that is, there exists a \mathcal{K} -function η_u such that

$$|\widehat{f}(\mathbf{x}, \mathbf{u}) - \widehat{f}(\mathbf{x}, \mathbf{u}')| \leq \eta_u(|\mathbf{u} - \mathbf{u}'|), \quad \forall \mathbf{x} \in X, \forall \mathbf{u}, \mathbf{u}' \in U.$$

△

It is worth noting that in cases when the function \widehat{f} satisfies the regularity assumptions stated in Sects. 10.1 and 10.2 and concerning the optimal IH and RH control laws (that is, \widehat{f} of class $\mathcal{C}^1(\mathbb{R}^d \times \mathbb{R}^m, \mathbb{R}^d)$), then Assumption 10.4 is satisfied.

Assumption 10.5 The additive transition uncertainty verifies

$$|\delta_t^f| \leq \mu(|\varsigma_t|), \quad \forall t \geq 0,$$

where μ is a \mathcal{K} -function. Moreover, δ_t^f is bounded in a compact ball Δ , that is

$$\delta_t^f \in \delta \triangleq \mathcal{B}^d(\bar{\delta}_f), \quad \forall t \geq 0,$$

with $\bar{\delta}_f \in \mathbb{R}^+$ finite. △

Our objective is to show that a compact set $\widetilde{\Upsilon} \subset X$, with $\mathbf{0} \in \widetilde{\Upsilon}$ and a state-feedback control law $\boldsymbol{\mu}(\mathbf{x}_t) : \widetilde{\Upsilon} \rightarrow U$ can be designed such that, for the closed-loop system

$$\mathbf{x}_{t+1} = \widehat{f}(\mathbf{x}_t, \boldsymbol{\mu}(\mathbf{x}_t)) + \delta_t^f, \quad t = 0, 1, \dots, \quad (10.35)$$

with $\mathbf{x}_0 = \bar{\mathbf{x}}$, the following properties hold:

- (a) System (10.35) is ISS in $\widetilde{\Upsilon}$ with respect to additive disturbances $\delta_t^f \in \delta$. In particular, there exists an ISS-Lyapunov function for which Points 1 and 2 of Definition 10.5 hold.
- (b) The set $\widetilde{\Upsilon}$ is RPI for System (10.35) with the additive disturbances $\delta_t^f \in \delta$.

In particular, our goal is to show that the above stabilizing control law $\boldsymbol{\mu}(\mathbf{x}_t)$ takes on the form of a suitable RH control law, that is, $\boldsymbol{\mu}(\mathbf{x}_t) = \boldsymbol{\mu}^{\text{RH}}(\mathbf{x}_t)$.

It is worth noting that an RH optimal control law $\boldsymbol{\mu}^{\text{RH}}(\mathbf{x}_t)$ meeting the objective above can be designed by resorting to techniques developed in the context of nonlinear MPC. For instance, the open-loop MPC formulations described in [14–16, 18, 27] guarantee the ISS of the closed-loop system with respect to bounded additive disturbances and allows the evaluation of the bounds on additive uncertainties under which the feasible set of the optimization problem associated with the MPC control can be rendered robustly positively invariant. In the following, we will refer in particular to the open-loop MPC design procedure described in [16, 18], showing that the maximal admissible uncertainty for practical stability can be determined by set-theoretic arguments. This result will be exploited in the next section to address approximate RH control laws obtained by the application of the ERIM and guaranteeing the practical stability of the closed-loop system.

Before addressing the FH optimal control problem from which to derive the RH control law, let us recall from the literature (see, for example, [16]) that the satisfaction of State Constraints (10.2) can be achieved, for any bounded disturbance sequence,

by imposing suitable *restricted state constraints* within the FH time window of Problem C1* (closed-loop form). The tightened constraints can be computed as in Lemma 10.1 from [16]. Before stating it, we introduce the following notation. For a given horizon length T , at every time instant $t \geq 0$, let

$$\mathbf{u}_{t,t+T-1|t} \triangleq \{\mathbf{u}_{t|t}, \mathbf{u}_{t+1|t}, \dots, \mathbf{u}_{t+T-1|t}\}$$

denote a generic sequence of input variables computed at time t over the control horizon T . Moreover, given the initial state \mathbf{x}_t and a control sequence $\mathbf{u}_{t,t+T-1|t}$, let $\hat{\mathbf{x}}_{t+j|t}$ denote the state “predicted” within the control horizon by means of the nominal model according to

$$\hat{\mathbf{x}}_{t+j|t} = \hat{\mathbf{f}}(\hat{\mathbf{x}}_{t+j-1|t}, \mathbf{u}_{t+j-1|t}), \quad j = 1, \dots, T, \text{ with } \hat{\mathbf{x}}_{t|t} = \mathbf{x}_t. \quad (10.36)$$

Lemma 10.1 (Constraints tightening) *Let $0 < L_{\hat{f}_x} < 1$. Assuming knowledge of an upper bound $\bar{\delta}_f$ on the additive uncertainty as specified by Assumption 10.5 and given the state vector \mathbf{x}_t at time t , if a control sequence $\mathbf{u}_{t,t+T-1|t}$ is feasible for the nominal model (10.36) with respect to the state constraints $\hat{X}_{t+j|t}$, where⁵*

$$\hat{X}_{t+j|t} \triangleq X \sim \mathcal{B}^d \left(\frac{L_{\hat{f}_x}^j - 1}{L_{\hat{f}_x} - 1} \bar{\delta}_f \right), \quad (10.37)$$

then, the control sequence $\mathbf{u}_{t,t+T-1|t}$, applied to System (10.32) in open loop, guarantees that $\mathbf{x}_{t+j} \in X$, $j = 1, \dots, T$. \square

Now, we restate the open-loop FH optimal control Problem C1* (closed-loop form) in an equivalent – though slightly different – form.

Problem 10.1 *Given a transition cost function h , a terminal cost function h_T , a terminal set X_f , and a sequence of constraint sets $\hat{X}_{t+j|t} \subseteq X$, $j \in \{1, \dots, T-1\}$ computed according to (10.37), find the optimal FH control sequence $\mathbf{u}_{t,t+T-1|t}^\circ$ that minimizes the cost*

$$J^{\text{FH}}(\mathbf{x}_t, \mathbf{u}_{t,t+T-1|t}, T) = \sum_{i=t}^{t+T-1} h(\hat{\mathbf{x}}_{i|t}, \mathbf{u}_{i|t}) + h_T(\hat{\mathbf{x}}_{t+T|t}), \quad (10.38)$$

subject to:

1. Nominal Dynamics (10.36);
2. The control and state constraints $\mathbf{u}_{t+j|t} \in U$, $\hat{\mathbf{x}}_{t+j|t} \in \hat{X}_{t+j|t}$, $j = 1, \dots, T-1$;
3. The terminal state constraint $\hat{\mathbf{x}}_{t+T|t} \in X_f$. \triangleleft

⁵The very special case $L_{\hat{f}_x} = 1$ can be trivially addressed by a few suitable modifications to Lemma 10.1.

From now on, we address the design of an RH control law based on a regional ISS approach; in this regard, it is worth noting that several works on MPC (see, for instance, [14, 16, 17, 29]) make use of ISS as a natural way to exploit the inherent stabilizing properties of MPC. More specifically, let us introduce the following further assumptions.

Assumption 10.6 The transition cost function h is such that $\underline{h}(|\mathbf{x}|) \leq h(\mathbf{x}, \mathbf{u})$, $\forall \mathbf{x} \in X$, $\forall \mathbf{u} \in U$, where \underline{h} is a \mathcal{K}_∞ -function. Moreover, h is Lipschitz continuous with respect to \mathbf{x} , uniformly in \mathbf{u} , with Lipschitz constant $L_h > 0$. \triangleleft

A comment similar to the one made about Assumption 10.4 can be repeated concerning Assumption 10.6. Specifically, if the transition cost function h satisfies the regularity assumptions stated in Sects. 10.1 and 10.2 and concerning the optimal IH and RH control laws (that is, h of class $\mathcal{C}^1(\mathbb{R}^d \times \mathbb{R}^m, \mathbb{R}^+)$ and h satisfying the respective assumptions in Theorem 10.1), then Assumption 10.6 is satisfied.

Assumption 10.7 A terminal cost function h_f , an auxiliary control function κ_f , and a set X_f are given such that

1. $X_f \subset X$, X_f closed, $\mathbf{0} \in X_f$;
2. $\exists \delta > 0 : \kappa_f(\mathbf{x}) \in U$, $\forall \mathbf{x} \in X_f \oplus \mathcal{B}^d(\delta)$;
3. $\widehat{f}(\mathbf{x}, \kappa_f(\mathbf{x})) \in X_f$, $\forall \mathbf{x} \in X_f \oplus \mathcal{B}^d(\delta)$;
4. $h_f(\mathbf{x})$ is Lipschitz in X , with Lipschitz constant $L_{h_f} > 0$;
5. $h_f(\widehat{f}(\mathbf{x}, \kappa_f(\mathbf{x}))) - h_f(\mathbf{x}) \leq -h(\mathbf{x}, \kappa_f(\mathbf{x}))$, $\forall \mathbf{x} \in X_f \oplus \mathcal{B}^d(\delta)$. \triangleleft

Assumption 10.8 Suppose that there exists a compact set $X_{\kappa_f} \supset X_f$ for which, being $\hat{\mathbf{x}}_{t|t} = \mathbf{x}_t \in X_{\kappa_f}$, it follows that

$$\bar{\mathbf{u}}_{t,t+T-1|t} \triangleq \{\kappa_f(\hat{\mathbf{x}}_{t|t}), \kappa_f(\hat{\mathbf{x}}_{t+1|t}), \dots, \kappa_f(\hat{\mathbf{x}}_{t+T-1|t})\}$$

is a feasible control sequence for the FH optimal control Problem 10.1 and such that Points 1, 2, and 5 of Assumption 10.7 hold true. \triangleleft

It is worth noting that the uncertainty bound for practical stability provided by several works in the literature [14–16, 18, 27] depends on the particular choice of the auxiliary control function $\kappa_f(\mathbf{x})$. To overcome this limitation, in the following paragraphs we state a result (see [25]) in which the worst-case uncertainty depends on the invariant properties of X_f through the computation of $\mathcal{C}_1(X_f)$. The possibility of computing a bound on the uncertainties based on set computations, rather than relying on contractivity under the auxiliary control function, results in a significant enlargement of the approximation level that it is possible to tolerate for practical stability and this is important in order to enable the use of approximate control laws determined offline by the ERIM. This constitutes a significant extension of the early and more conservative results obtained in [23, 24].

Indeed, for the computation of the predecessor of the terminal set, we can take full advantage of the whole input constraint set U and we are not restricted to the sole auxiliary control function. It is also important to notice that the above

advantages – important for the design of stabilizing approximate RH control laws – do not introduce additional conservativeness with respect to results available in the literature (for instance, this can be immediately ascertained by comparing Assumption 10.7 with Assumption 8 in [14]).

Under the previously stated assumptions, the following important theorem (see [25] for the proof) characterizes the ISS property of the closed-loop system with respect to bounded additive uncertainties.

Theorem 10.5 (Regional ISS) *Let us denote as $X^{RH} \subset \mathbb{R}^d$ the set of state vectors for which the FH optimal control Problem 10.1 admits feasible solutions. Under Assumptions 10.4, 10.5, 10.6, 10.7, and 10.8, System (10.32), driven by the RH optimal control law $\mu^{RH^\circ}(x_t)$ obtained by solving Problem 10.1 is regional ISS in X^{RH} with respect to additive disturbances $\delta_t^f \in \delta$ satisfying Assumption 10.5, that is, $\delta = \mathcal{B}^d(\bar{\delta}_f)$ and*

$$\bar{\delta}_f \leq L_{\hat{f}_x}^{1-T} \text{dist}(\mathbb{R}^d \setminus \mathcal{C}_1(X_f), X_f). \quad (10.39)$$

□

It is worth noting that, from the perspective of determining regionally ISS stabilizing RH control laws, a key aspect is the design of an auxiliary control function $\kappa_f(x)$ such that Assumption 10.7 holds. In this respect, under slightly more restrictive hypotheses on the model of the dynamic system and on the FH cost function, we give the following useful *constructive design* result (the proof is reported in [7]).

Lemma 10.2 *Assume that f, \hat{f} are of class \mathcal{C}^2 , $h(x, u) = x^\top Qx + u^\top Ru$, with Q and R being positive-definite matrices. Furthermore, suppose that there exists a matrix K such that $A_{cl} = A + BK$ is stable with $A \triangleq \left. \frac{\partial \hat{f}}{\partial x} \right|_{x=0; u=0}$, $B \triangleq \left. \frac{\partial \hat{f}}{\partial u} \right|_{x=0; u=0}$. Let $\tilde{Q} \triangleq \beta(Q + K^\top RK)$ with $\beta > 1$, and denote by Π the unique symmetric positive-definite solution of the following Lyapunov equation:*

$$A_{cl}^\top \Pi A_{cl} - \Pi + \tilde{Q} = 0. \quad (10.40)$$

Then, there exist a constant $\chi \in \mathbb{R}^+$ and a finite integer \bar{T} such that for all $T \geq \bar{T}$ the final set $X_f \triangleq \{x \in \mathbb{R}^d : x^\top \Pi x \leq \chi\}$ satisfies Assumption 10.7 with $\kappa_f(x) = Kx$, $h_T = x^\top \Pi x$. □

Summing up, in this section some notation and some recent useful results on ISS have been provided and we have shown how to design an input-to-state stabilizing RH control law which makes the set $X^{RH} \subseteq X$ RPI with respect to additive disturbances $\delta_t^f \in \Delta$.

In the next section, on the basis of the above-described stabilizing properties of μ^{RH° , some results will be given allowing us to analyze the practical stability properties of the closed-loop system driven by an approximate RH control law obtained by the ERIM.

10.4 Stabilizing Approximate RH Control Laws

In this section, we consider a given ISS-stabilizing RH control law μ^{RH° and we analyze the closed-loop stability properties when the control law μ^{RH° is replaced by an *approximate* control law $\tilde{\mu}^{\text{RH}}$ obtained by the ERIM. In particular, for any given parameter vector w , the effects of the approximation error $\mu^{\text{RH}^\circ}(x) - \tilde{\mu}^{\text{RH}}(x, w)$, $x \in X$ on the closed-loop stability have to be analyzed.

In deriving the RH optimal control law μ^{RH° (both online and offline), it is worth noting that computational errors may affect the vector $u_t^{\text{RH}^\circ} = \mu^{\text{RH}^\circ}(x_t)$. Indeed, beyond these computational errors, another source of errors in the computation of the control vector $u_t^{\text{RH}^\circ}$ may be induced by the FSP function that approximates the optimal RH control law μ^{RH° within the context of the application of the ERIM. The ISS and ISpS concepts developed in the previous section turn out to be very useful in devising a *constructive approach* to design practical stabilizing RH control laws addressing the above-mentioned sources of error without spoiling the closed-loop stability property guaranteed by a properly designed optimal RH control law.

The statement of Problem 10.1 (or its equivalent formulation given in Problem C1' (open-loop form)) does not impose any particular way of computing the control vector $u_t^{\text{RH}^\circ}$ as a function of x_t . Actually, we have two possibilities to determine the optimal control law numerically:

- (a) *Online computation.* Problem 10.1 is an open-loop optimal control problem and may be regarded as a nonlinear programming (NLP) one. The main advantage of this approach (adopted in most works in the literature) is that many well-established NLP techniques are available to solve Problem 10.1.
- (b) *Offline computation.* This approach (see the seminal paper [24]) implies that the optimal RH control law $\mu^{\text{RH}^\circ}(x_t)$ – constructed by solving Problem 10.1 for every possible x_t – has to be computed a priori and stored in the regulator's memory. Clearly, the offline computation has advantages and disadvantages that are opposite to the ones of the online approach. No significant online computational effort is required from the regulator, but a (possibly) very large amount of computer memory may be required to store the closed-loop optimal control law (as regards the FSP functions, we have to store the parameter vectors in the memory).

Consistently with what we did in previous chapters concerning the FH optimal control problems, we are mainly interested in computing the RH control law offline. However, within our general non-LQ context, deriving analytically the function $\mu^{\text{RH}^\circ}(x_t)$ is practically an impossible task. At least in principle, this function could be determined in the backward phase of a DP procedure. However, as already discussed in the previous chapters dealing with FH optimal control problems, we do not resort to this algorithm. As we have the possibility of computing (offline) any number of open-loop optimal control sequences $u_{t,t+T-1|t}^\circ$ solving Problem 10.1 for different vectors $x_t \in X$, similar to what has been done in previous chapters (see, for example, Sect. 6.5), we approximate the optimal RH control function μ^{RH° by an

FSP function of the form $\tilde{\mu}^{\text{RH}}(\mathbf{x}_t, \mathbf{w}_n)$ obtained by the ERIM, where the superscript “RH” enhances the fact that this FSP function plays the role of the approximate RH control law $\tilde{\mu}^{\text{RH}}$ mentioned in Sect. 10.2 and \mathbf{w}_n is a vector of parameters to be tuned.

More specifically, for any given parameter vector \mathbf{w}_n , the effects of the approximation error $\mu^{\text{RH}^\circ}(\mathbf{x}_t) - \tilde{\mu}^{\text{RH}}(\mathbf{x}_t, \mathbf{w}_n)$, $\mathbf{x}_t \in X$ on the closed-loop stability have to be analyzed. To this end, let us consider the following dynamic system:

$$\mathbf{x}_{t+1} = \hat{f}(\mathbf{x}_t, \mu^{\text{RH}^\circ}(\mathbf{x}_t)) + \mathbf{v}_t, \quad t \geq 0, \quad (10.41)$$

with $\mathbf{x}_0 = \bar{\mathbf{x}} \in \Upsilon \subset \mathbb{R}^d$ and where $\mathbf{v}_t \in N \triangleq \mathcal{B}^d(\bar{\delta}_v)$ is a generic bounded disturbance input. We are going to show that the stability properties of System (10.41) in a suitably constructed set Υ can be inferred from those of the generic system (10.35) considered in Sect. 10.3 in the RPI set $\widetilde{\Upsilon}$ with bounded input disturbances $\mathbf{v}_t \in N$.

In order to proceed with this stability analysis, let us split up the state transition uncertainty into three contributions, affecting the control function, the state vector, and the whole system model, respectively, according to the following assumption.

Assumption 10.9 Consider the \mathcal{K}_∞ -function $\eta_x(s) = L_{\hat{f}_x} s + s$ for $s \geq 0$, where $L_{\hat{f}_x}$ is the Lipschitz constant given in Assumption 10.4. Moreover, let $\bar{\delta}_q \in \mathbb{R}^+$ and $\bar{\delta}_v \in \mathbb{R}^+$ be two positive scalars satisfying the following inequality

$$\bar{\delta}_q + \bar{\delta}_v + \bar{\delta}_v \leq \bar{\delta}_f, \quad (10.42)$$

where $\bar{\delta}_f$ is given by Assumption 10.5. By defining $\bar{v} \triangleq \eta_u^{-1}(\bar{\delta}_v)$ and $\bar{q} \triangleq \eta_x^{-1}(\bar{\delta}_q)$, suppose $\exists \lambda \in (0, 1)$ such that $\forall \mathbf{x} \in X$, $\exists \mathbf{x}' \in \mathcal{B}^d(\mathbf{x}, \bar{q}) \cap X$ for which

1. $|\mu^{\text{RH}^\circ}(\mathbf{x}') - \tilde{\mu}^{\text{RH}}(\mathbf{x}')| \leq (1 - \lambda)\bar{v};$
2. $|\tilde{\mu}^{\text{RH}}(\mathbf{x}) - \tilde{\mu}^{\text{RH}}(\mathbf{x}')| \leq \lambda\bar{v}.$

Finally, suppose that $\tilde{\mu}^{\text{RH}}(\mathbf{x}) \in U$, $\forall \mathbf{x} \in X$. \square

The following result concerning the perturbed state trajectory caused by approximation errors on the optimal RH control law will be instrumental for the subsequent analysis (see the proof in [25]).

Lemma 10.3 (Approximation-induced perturbations) *Under Assumption 10.9, given $\mathbf{x}_t \in X$, there exist two vectors $\mathbf{q}_t \in Q \triangleq \mathcal{B}^d(\bar{q})$ and $\mathbf{v}_t \in V \triangleq \mathcal{B}^m(\bar{v})$, with $\mathbf{q}_t \triangleq \mathbf{x}'_t - \mathbf{x}_t$ and $\mathbf{v}_t \triangleq \tilde{\mu}^{\text{RH}}(\mathbf{x}_t) - \mu^{\text{RH}^\circ}(\mathbf{x}'_t)$, such that System (10.41) can be rearranged as*

$$\mathbf{x}_{t+1} = \hat{f}(\mathbf{x}_t, \mu^{\text{RH}^\circ}(\mathbf{x}_t + \mathbf{q}_t) + \mathbf{v}_t) + \mathbf{v}_t, \quad t \geq 0, \quad (10.43)$$

with $\mu^{\text{RH}^\circ}(\mathbf{x}_t + \mathbf{q}_t) + \mathbf{v}_t \in U$. \square

Now, we are able to state the following important result concerning the stability properties of the closed-loop system driven by the approximate RH control law $\tilde{\mu}^{\text{RH}}$ (see the proof in [25]).

Theorem 10.6 Suppose that Assumptions 10.4–10.9 hold and let $\Upsilon \triangleq \tilde{\Upsilon} \sim Q$. Then, the following statements hold:

1. The set $\Upsilon \subset X$ is RPI for Closed-Loop System (10.43) with $v_t \in V$, $q_t \in Q$, and $v_t \in N$.
2. Closed-loop system (10.43) is ISS in Υ with respect to the perturbations v_t , q_t and v_t .

□

Theorem 10.6 is the main result enabling us to address the design of stabilizing approximate RH control laws: indeed it yields admissible bounds for the errors caused by the approximation of the optimal RH control law in order to ensure the ISpS property for the closed-loop system.

It is worth noting that Assumption 10.9 (and hence Theorem 10.6) does not require the control law μ^{RH° to be smooth and, in this regard, Theorem 10.6 has been exploited in [25] in the context of approximate discontinuous RH control laws. On the other hand, the application of the ERIM described in the previous chapters implies an assumption that the ambient space to which the optimal RH control law is supposed to belong is characterized by suitable smoothness properties. For instance, the space $\mathcal{C}(X, \mathbb{R}^m)$ is required to ensure that the \mathcal{C} -density property holds, thus allowing us to bound the maximum approximation error on X when suitable FSP functions are used in the ERIM to implement the approximate RH control laws (see (2.66) and (2.67)). Indeed, higher degrees of smoothness are required when the aim is to mitigate the curse of dimensionality. Therefore, from now on Assumption 10.9 is restricted to the case for RH control laws $\mu^{\text{RH}^\circ}(\mathbf{x})$ that are at least Lipschitz continuous with respect to \mathbf{x} . Thus, we introduce the following slightly different assumption.

Assumption 10.10 Let us introduce the \mathcal{K}_∞ -function $\eta_x(s) = L_{\hat{f}_x} s + s$ for $s \geq 0$ and let $\bar{\delta}_q \in \mathbb{R}^+$ and $\bar{\delta}_v \in \mathbb{R}^+$ be two positive scalars satisfying the following inequality:

$$\bar{\delta}_q + \bar{\delta}_v + \bar{\delta}_v \leq \bar{\delta}_f, \quad (10.44)$$

where $\bar{\delta}_f$ is given by Assumption 10.5. Defining $\bar{q} \triangleq \eta_x^{-1}(\bar{\delta}_q)$, assume that $\forall \mathbf{x} \in X$, $\exists \mathbf{x}' \in \mathcal{B}^d(\mathbf{x}, \bar{q}) \cap X$ such that

$$\eta_u(|\tilde{\mu}^{\text{RH}}(\mathbf{x}) - \mu^{\text{RH}^\circ}(\mathbf{x}')|) + \eta_x(|\mathbf{x}' - \mathbf{x}|) + \bar{\delta}_v \leq \bar{\delta}(\mathbf{x}'), \quad (10.45)$$

where $\bar{\delta}(\mathbf{x}')$ is given by

$$\bar{\delta}(\mathbf{x}') \triangleq \inf\{c \in \mathbb{R}^+ : \exists \delta \in \mathcal{B}^d(c) \text{ such that } \hat{f}(\mathbf{x}', \mu^{\text{RH}^\circ}(\mathbf{x}')) + \delta \notin \Upsilon\}.$$

Moreover, let us assume that $\mu^{\text{RH}^\circ}(\mathbf{x}) \in U$, $\forall \mathbf{x} \in X$. □

Remark 10.7 Notice that $\bar{\delta}(\mathbf{x}')$ characterizes a compact ball centered in \mathbf{x}' containing all the state vectors that can be steered from \mathbf{x}' into Υ by the control law μ^{RH° . □

From Theorem 10.6, the following result on the stability properties of the closed-loop system under the action of the approximate RH control law can be easily derived (see [25]).

Corollary 10.1 *Suppose that Assumptions 10.4–10.8, and 10.10 hold and let $\Upsilon \triangleq \widetilde{\Upsilon} \sim Q$. If the optimal RH control law μ^{RH^o} is ISS-stabilizing in $\widetilde{\Upsilon}$, then any approximate RH control law $\tilde{\mu}^{\text{RH}}$ verifying (10.45) is ISS-stabilizing in $\Upsilon = \widetilde{\Upsilon} \sim Q$. \square*

It is worth noting that, being $\bar{\delta}_f \leq \bar{\delta}(x')$ (by using a local bound on perturbations in place of the semi-global one), Condition (10.45) turns out to be less restrictive than Requirements 1 and 2 in Assumption 10.9, formulated concerning a more general case in which the optimal RH control law μ^{RH^o} is not required to satisfy a smoothness condition like the one stated in Assumption 10.10.

10.5 Offline Design of the Stabilizing Approximate RH Control Law Obtained by the ERIM

As mentioned in the previous sections, we are interested in computing the RH control law offline and we aim at approximating the optimal RH control function μ^{RH^o} by an FSP function of the form $\tilde{\mu}^{\text{RH}}(x_t, w_n)$ obtained by the ERIM and that guarantees the ISpS property. Owing to the fact that the FSP functions considered in this book enjoy a smoothness degree for which Assumption 10.10 is satisfied, referring to (10.41), let us assume that a bound $|v_t| \leq \bar{\delta}_v$ on the additive transition uncertainty is given. Then, the first step is to set the values of the scalars $\bar{\delta}_v$ and $\bar{\delta}_q$ such that Inequality (10.44) holds true.

Next,⁶ similar to (6.20), we discretize the set X by selecting L discretized points $x^{(l)}$, $l = 1, \dots, L$, in the admissible set X , thus defining the discrete set

$$X' \triangleq \{x^{(l)} \in X : l = 1, \dots, L\}. \quad (10.46)$$

Moreover, analogously to (6.51), we let

$$\Sigma^{\text{RH}} \triangleq \{(x^{(l)}, \mu^{\text{RH}^o}(x^{(l)})) : x^{(l)} \in X'\}. \quad (10.47)$$

The design of the approximate RH control law $\tilde{\mu}^{\text{RH}}$ should guarantee that closed-loop ISpS is achieved – this is a necessity, not an option. In this respect, Corollary 10.1 is a key result to be exploited. To this end – as will be seen later on in this section – the way in which the set X' is constructed and the *dispersion* of the discretized points $x^{(l)}$, $l = 1, \dots, L$, in the admissible set X are related to the satisfaction of the assumptions stated in Corollary 10.1.

As discussed in Chap. 6, there are three main possibilities of discretizing the set X to obtain the set X' :

⁶In this section, for the sake of simplicity the subscript “ t ” is dropped.

1. The discretized set X' takes on the form of a *full uniform grid*.
2. The set X' is obtained via the Monte Carlo approach by drawing i.i.d. sample vectors $\mathbf{x}^{(l)}$ according to a uniform probability density.
3. The set X' is given by deterministic sequences of nodes that are spread in the “most uniform way.”

In this section, we devise a *constructive procedure* for the design of a stabilizing approximate RH control law $\tilde{\mu}^{\text{RH}}$ based on suitably forming the discretized set X' as a *full uniform grid*. This construction of X' is straightforward and related in a simple way to the satisfaction of the assumptions stated in Corollary 10.1. Unfortunately, as illustrated in Chap. 6, this choice leads by construction to a curse of dimensionality – i.e., an unacceptable growth of memory and computational requirements with d – and limits the applicability of full uniform grids to a small dimension of the state (see Sect. 6.4.1 for more details). As discussed in Sect. 4.4, the use of low-discrepancy sequences to generate the discretized points $\mathbf{x}^{(l)}$, $l = 1, \dots, L$ of the set X' can help to mitigate this effect when the dimension of the state is large. Later in this section (see Remark 10.10), some further comments will be given on the possible use of this different construction of the set X' . On the other hand, we anticipate that, as shown in the numerical results reported later on in this chapter, the use of a uniform grid does not necessarily imply the need for a very large model complexity n .

The uniform grid has to meet the constraints contained in the following further assumption.

Assumption 10.11 Given the set X and the positive scalar $\bar{\delta}_q \in \mathbb{R}^+$ satisfying Inequality (10.44), let the set X' verify the following conditions:

1. $\exists \bar{\delta} \in \mathbb{R}^+ : \forall \mathbf{x} \in X, \exists l \in \{1, \dots, L\}$ such that $|\mathbf{x} - \mathbf{x}^{(l)}| \leq \bar{\delta} < \eta_x^{-1}(\bar{\delta}_q)$;
2. $\exists \underline{\delta} \in \mathbb{R}^+ : |\mathbf{x}^{(l')} - \mathbf{x}^{(l'')}| \geq \underline{\delta}, \forall l', l'' \in \{1, \dots, L\}, l' \neq l''$.

□

Notice that Assumption 10.11 takes on a form which is typical in the literature (see, for instance [1] in which $\bar{\delta}$ and $\underline{\delta}$ are named “knot density parameter” and “knot separation parameter,” respectively).

Let us now consider a grid X' satisfying Assumption 10.11 and a given parameter vector \mathbf{w} . Then, Inequality (10.45) can be rewritten as

$$\begin{aligned} \eta_u(|\tilde{\mu}^{\text{RH}}(\mathbf{x}', \mathbf{w}) - \tilde{\mu}^{\text{RH}}(\mathbf{x}, \mathbf{w})| + |\tilde{\mu}^{\text{RH}}(\mathbf{x}, \mathbf{w}) - \mu^{\text{RH}^\circ}(\mathbf{x}')|) + \eta_x(|\mathbf{x}' - \mathbf{x}|) \\ \leq \bar{\delta}(\mathbf{x}') - \bar{\delta}_v. \end{aligned} \quad (10.48)$$

Owing to the smoothness of the FSP function, it follows that $\tilde{\mu}^{\text{RH}}(\cdot, \mathbf{w})$ is locally Lipschitz in X . Specifically, for any parameter vector \mathbf{w} , there exists a function $L_{\tilde{\mu}}(\mathbf{x}, \mathbf{w}) : \mathbb{R}^d \rightarrow \mathbb{R}^+$ such that

$$|\tilde{\mu}^{\text{RH}}(\mathbf{x}, \mathbf{w}) - \tilde{\mu}^{\text{RH}}(\mathbf{x}', \mathbf{w})| \leq L_{\tilde{\mu}}(\mathbf{x}, \mathbf{w})|\mathbf{x} - \mathbf{x}'|, \quad \forall \mathbf{x}' \in \mathcal{B}^d(\mathbf{x}, \bar{q}).$$

Now, the problem consists in determining an FSP function $\tilde{\mu}^{\text{RH}}(\mathbf{x}, \mathbf{w})$ such that Assumption 10.10 is verified over the uniform grid X' determined according to Assumption 10.11. In this respect, consider a given state $\mathbf{x} \in X$; from (10.48) it follows that, for all $\mathbf{x}' \in \mathcal{B}^d(\mathbf{x}, \bar{q})$, we have

$$\begin{aligned} & \eta_u(|\tilde{\mu}^{\text{RH}}(\mathbf{x}', \mathbf{w}) - \tilde{\mu}^{\text{RH}}(\mathbf{x}, \mathbf{w})| + |\tilde{\mu}^{\text{RH}}(\mathbf{x}', \mathbf{w}) - \mu^{\text{RH}^\circ}(\mathbf{x}')|) + \eta_x(|\mathbf{x}' - \mathbf{x}|) \\ & \leq \eta_u(L_{\tilde{\mu}}(\mathbf{x}', \mathbf{w})|\mathbf{x}' - \mathbf{x}| + |\mu^{\text{RH}}(\mathbf{x}', \mathbf{w}) - \mu^{\text{RH}^\circ}(\mathbf{x}')|) + \eta_x(|\mathbf{x}' - \mathbf{x}|) \\ & \leq \eta_u(L_{\tilde{\mu}}(\mathbf{x}', \mathbf{w})\bar{q} + |\mu^{\text{RH}}(\mathbf{x}', \mathbf{w}) - \mu^{\text{RH}^\circ}(\mathbf{x}')|) + \eta_x(\bar{q}) \\ & \leq \eta_u(L_{\tilde{\mu}}(\mathbf{x}', \mathbf{w})\bar{q} + |\mu^{\text{RH}}(\mathbf{x}', \mathbf{w}) - \mu^{\text{RH}^\circ}(\mathbf{x}')|) + L_{\hat{f}_x}\bar{q} + \bar{q}. \end{aligned}$$

Conversely, the satisfaction of

$$\eta_u(L_{\tilde{\mu}}(\mathbf{x}^{(l)}, \mathbf{w})\bar{q} + |\tilde{\mu}^{\text{RH}}(\mathbf{x}^{(l)}, \mathbf{w}) - \tilde{\mu}^{\text{RH}^\circ}(\mathbf{x}^{(l)})|) + (L_{\hat{f}_x} + 1)\bar{q} \leq \bar{\delta}(\mathbf{x}^{(l)}) - \bar{d}_v, \quad (10.49)$$

for some $\mathbf{x}^{(l)} \in X'$ implies that (10.45) is verified for all $\mathbf{x}' \in \mathcal{B}^d(\mathbf{x}^{(l)}, \bar{q})$. Therefore, if Inequality (10.49) holds for $\mathbf{x}^{(l)} \in X'$, $l = 1, \dots, L$, then $\tilde{\mu}^{\text{RH}}$ verifies Assumption 10.10 in the whole domain X^{RH} defined in Theorem 10.5.

Setting $\epsilon(\mathbf{x}^{(l)}) = \bar{\delta}(\mathbf{x}^{(l)}) - \bar{\delta}_v - (L_{\hat{f}_x} + 1)\bar{q}$, Inequality (10.49) can be rewritten as

$$\eta_u(L_{\tilde{\mu}}(\mathbf{x}^{(l)}, \mathbf{w})\bar{q} + |\tilde{\mu}^{\text{RH}}(\mathbf{x}^{(l)}, \mathbf{w}) - \mu^{\text{RH}^\circ}(\mathbf{x}^{(l)})|) \leq \epsilon(\mathbf{x}^{(l)}),$$

which can be finally rearranged as

$$L_{\tilde{\mu}}(\mathbf{x}^{(l)}, \mathbf{w})\bar{q} + |\tilde{\mu}^{\text{RH}}(\mathbf{x}^{(l)}, \mathbf{w}) - \mu^{\text{RH}^\circ}(\mathbf{x}^{(l)})| \leq \eta_u^{-1}(\epsilon(\mathbf{x}^{(l)})).$$

From an operational point of view, we first construct a uniform grid X' with $\bar{\delta} = \bar{q}$. Afterward, by letting

$$\epsilon'(\mathbf{x}^{(l)}) \triangleq \eta_u^{-1}(\epsilon(\mathbf{x}^{(l)})), \quad l = 1, \dots, L,$$

we evaluate the function $\epsilon'(\cdot)$ on the sample states $\mathbf{x}^{(l)} X'$, $l = 1, \dots, L$.

Then, a sufficient condition for (10.45) to hold (and hence, for the approximate RH control law $\tilde{\mu}^{\text{RH}}$ to guarantee the ISpS property for the closed-loop system according to Corollary 10.1) can be expressed as

$$L_{\tilde{\mu}}(\mathbf{x}^{(l)}, \mathbf{w})\bar{q} + |\tilde{\mu}^{\text{RH}}(\mathbf{x}^{(l)}, \mathbf{w}) - \mu^{\text{RH}^\circ}(\mathbf{x}^{(l)})| \leq \epsilon'(\mathbf{x}^{(l)}), \quad l = 1, \dots, L. \quad (10.50)$$

Before proceeding with the design of the practical stabilizing approximate RH control law $\tilde{\mu}^{\text{RH}}$, the following remark is in place.

Remark 10.8 In order to compute $\epsilon'(\mathbf{x}^{(l)})$, the \mathcal{K} -function $\eta_u(\cdot)$ must be determined. In general, for a nonlinear function \hat{f} , a local bound on $\eta_u(\cdot)$ can be easily computed

by resorting to the computation of a local Lipschitz constant $L_{\hat{f}_u}(\mathbf{x}^{(l)})$ such that

$$|\hat{f}(\mathbf{x}, \mathbf{u}) - \hat{f}(\mathbf{x}, \mathbf{u}')| \leq L_{\hat{f}_u}(\mathbf{x}^{(l)})|\mathbf{u} - \mathbf{u}'|, \quad \forall \mathbf{x} \in \mathcal{B}^d(\mathbf{x}^{(l)}, \bar{q}), \forall \mathbf{u}, \mathbf{u}' \in U.$$

A (possibly conservative) upper bound on $L_{\hat{f}_u}(\mathbf{x}^{(l)})$ is given by

$$L_{\hat{f}_u}(\mathbf{x}^{(l)}) = \max_{(\bar{\mathbf{x}}, \bar{\mathbf{u}}) \in \mathcal{B}^d(\mathbf{x}^{(l)}, \bar{q}) \times U} \sum_{j=1}^m \sum_{i=1}^d \left| \frac{\partial \hat{f}^{(i)}}{\partial u^{(j)}} \right|_{\mathbf{x}=\bar{\mathbf{x}}, \mathbf{u}=\bar{\mathbf{u}}},$$

where $\hat{f}^{(i)}$ and $u^{(j)}$ are the i -th and the j -th components of \hat{f} and \mathbf{u} , respectively. \triangleleft

To exploit Corollary 10.1 by satisfying Condition (10.50), for any parameter vector \mathbf{w}_n , we have to specify quantitatively the magnitude of the approximation errors

$$\tilde{\mu}^{\text{RH}}(\mathbf{x}^{(l)}, \mathbf{w}) - \mu^{\text{RH}^\circ}(\mathbf{x}^{(l)}), \quad l = 1, \dots, L,$$

generated by the approximate RH control law when it approximates the control law μ^{RH° and evaluated on every sample state $\mathbf{x}^{(l)} \in X'$, $l = 1, \dots, L$.

As we did in Chap. 6, either for the approximation of the optimal cost-to-go functions $\tilde{J}_t^\circ(\mathbf{x}_t)$ by the networks $\tilde{J}(\mathbf{x}_t, \mathbf{w}_{tn_t \Sigma_t})$ (see Problems 6.3) or for the approximation of the optimal closed-loop control functions $\tilde{\mu}_t^\circ(\mathbf{x}_t)$ by another family of FSP functions $\mathbf{u}_t = \tilde{\mu}(\mathbf{x}_t, \mathbf{w}_{tn_t}^u)$, we approximate the optimal RH control law μ^{RH° by an FSP function $\tilde{\mu}^{\text{RH}}(\mathbf{x}^{(l)}, \mathbf{w}_n)$.

The integer n and the parameter vector \mathbf{w}_n have to be computed by solving the following problem.

Problem 10.2 For a given family of FSP functions $\tilde{\mu}^{\text{RH}}(\mathbf{x}^{(l)}, \mathbf{w}_n)$, for a given set $\Sigma^{\text{RH}} = \{(\mathbf{x}^{(l)}, \mu^{\text{RH}^\circ}(\mathbf{x}^{(l)})) : \mathbf{x}^{(l)} \in X'\}$ determined according to (10.47), and for a given set of bounds $\epsilon'(\mathbf{x}^{(l)})$, $l = 1, \dots, L$, find the smallest integer n such that

$$\begin{aligned} \min_{\mathbf{w}_n} \max_{\mathbf{x}^{(l)} \in X', l=1, \dots, L} & \left\{ L_{\tilde{\mu}}(\mathbf{x}^{(l)}, \mathbf{w}_n) \bar{q} + |\tilde{\mu}^{\text{RH}}(\mathbf{x}^{(l)}, \mathbf{w}_n) - \mu^{\text{RH}^\circ}(\mathbf{x}^{(l)})| \right\} \\ & \leq \epsilon'(\mathbf{x}^{(l)}). \end{aligned} \quad (10.51)$$

\triangleleft

Remark 10.9 It is important to emphasize that the smoothness properties of the function \hat{f} and of the optimal RH control law μ^{RH° influence the specific values taken on by the terms $L_{\tilde{\mu}}(\mathbf{x}^{(l)}, \mathbf{w}_n)$ and \bar{q} in the argument of the maximum operator of (10.51). Hence, depending on the FSP structure $\tilde{\mu}^{\text{RH}}(\mathbf{x}^{(l)}, \mathbf{w}_n)$, the integer n , and the state sample set $X' = \{\mathbf{x}^{(l)} \in X, l = 1, \dots, L\}$, it may well occur that $L_{\tilde{\mu}}(\mathbf{x}^{(l)}, \mathbf{w}_n) \bar{q} > \epsilon'(\mathbf{x}^{(l)})$ thus making it impossible to solve Problem 10.2. On the other hand, it is straightforward to show that $L_{\tilde{\mu}}(\mathbf{x}^{(l)}, \mathbf{w}_n) \bar{q} \rightarrow 0$ as $\bar{\delta} \rightarrow 0$ thus

implying that there always exists a “suitably dense” state sample set Σ^{RH} such that $L_{\bar{\mu}}(\mathbf{x}^{(l)}, \mathbf{w}_n)\bar{q} < \epsilon'(\mathbf{x}^{(l)})$ which, in turn, allows us to solve Problem 10.2, if one can also make the terms $|\bar{\mu}^{\text{RH}}(\mathbf{x}^{(l)}, \mathbf{w}_n) - \mu^{\text{RH}}(\mathbf{x}^{(l)})|$ sufficiently small. Under suitable assumptions, this can be guaranteed by the approximation error bounds expressed in terms of the supremum norm, such as the ones discussed in Sect. 3.7.2 for the case of sigmoidal OHL networks (similar bounds in the supremum norm hold for radial OHL networks [9, 10]). It is worth remarking that, in some cases, they allow mitigating the curse of dimensionality. \triangleleft

Remark 10.10 The possible necessity of constructing a “dense” state sample set leading to a suitably small value of $\bar{\delta}$ mentioned in Remark 10.9 clearly implies a very large number of nodes $\mathbf{x}^{(l)}$ in the sample set X' which, in turn, gives rise to the curse of dimensionality. As stated at the beginning of this section, another option is to construct the sample set X' by generating the nodes $\mathbf{x}^{(l)}$ as a low-discrepancy sequence. Remarkably, the conditions in Assumption 10.11 can be easily expressed in terms of the *dispersion* of the set X' (see Eq. (5.102)) for the definition of “dispersion” of a sequence of points in a set). As a consequence, low-discrepancy sequences guarantee the fulfillment of Assumption 10.11, since it can be proved [22] that the dispersion of a set of points can be made arbitrarily small when its star discrepancy $D_L^*(X')$ converges to zero as L grows. In practice though, the possible necessity of constructing a “dense” state sample set leading to a suitably small value of $\bar{\delta}$ mentioned in Remark 10.9 may still imply a very large number of nodes $\mathbf{x}^{(l)}$ in the sample set X' , even when this kind of sequence is employed. This is due to the slow rate of convergence of the dispersion which, roughly speaking, goes as the d th root of the star discrepancy. This clearly leaves the dimensionality issues unavoidable, since in order to make the dispersion decrease linearly we have to add points in such a way that the star discrepancy decreases exponentially (which means, for instance, that if we want to halve the dispersion, we need to add points until the star discrepancy is 2^d -times smaller). Yet, the use of a low-discrepancy sequence characterized by a suitably small discrepancy $D_L^*(X')$ can be a viable alternative to a full uniform grid (see also the recent paper [6]). \triangleleft

Owing to Remark 10.9, it follows that Problem 10.2 can be solved by an offline trial-and-error procedure which, of course, can be significantly simplified on specific cases by exploiting possible a priori knowledge on the smoothness characteristics of the RH optimal control law.

In the next section, some numerical results are presented (see [25]), showing the effectiveness of the ERIM approach in designing offline a practical stabilizing approximate RH control law based on the ERIM and on the ISS-based approach addressed in the present section.

10.6 Numerical Results

Consider the following nonlinear system:

$$\begin{aligned} x_{1,t+1} &= x_{1t} + \delta \left[-x_{2t} + \frac{1}{2} (1 + x_{1t}) u_t \right], \\ x_{2,t+1} &= x_{2t} + \delta \left[x_{1t} + \frac{1}{2} (1 - 4x_{2t}) u_t \right], \end{aligned} \quad (10.52)$$

which is the discretized version of the undamped oscillator addressed in [19], where $\delta = 0.05$ is the sampling period.

The FH cost function is quadratic and is given by

$$J^{\text{FH}}(\mathbf{x}_t, \mathbf{u}_{t,t+T-1|t}, T) = \sum_{i=t}^{t+T-1} (\hat{\mathbf{x}}_{i|t}^\top Q \hat{\mathbf{x}}_{i|t} + R u_{i|t}^2) + \hat{\mathbf{x}}_{t+T|t}^\top Q_T \hat{\mathbf{x}}_{t+T|t}, \quad (10.53)$$

where $T = 8$ and

$$Q = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}, \quad R = 1, \quad Q_T = \begin{bmatrix} 91.56 & -23.61 \\ -23.61 & 167.28 \end{bmatrix}.$$

The state constraint set X is depicted in Fig. 10.5 whereas the input constraint set is given by $U \triangleq \{u \in \mathbb{R} : |u| \leq 2\}$.

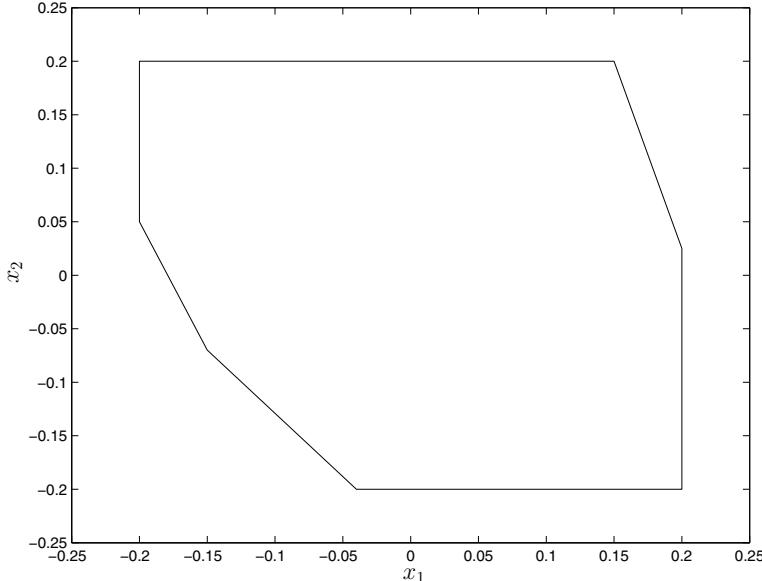


Fig. 10.5 The state constraint set X

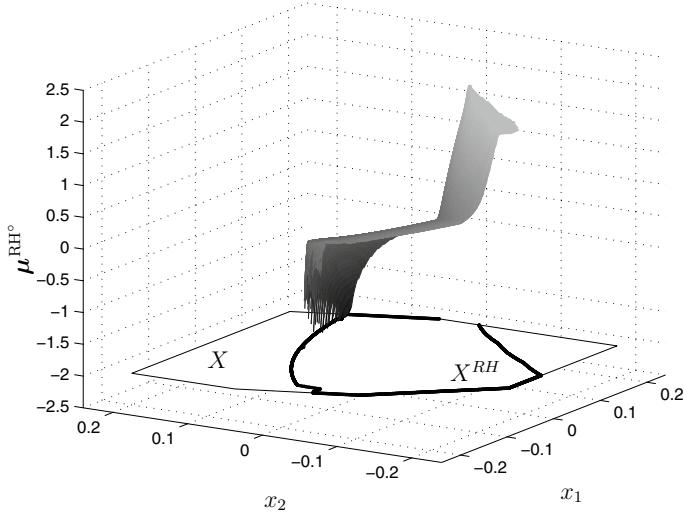


Fig. 10.6 The optimal RH control function. The state constraint set X and the feasible set X^{RH} are enhanced. ©2013 Taylor & Francis. Reprinted, with permission, from [25]

The Lipschitz constant for the system is $L_{\hat{f}_x} = 1.1390$. An auxiliary control law can be designed following, for example, the procedure described in [23], thus obtaining the auxiliary control law $u_t = \kappa_f(\mathbf{x}_t) = \mathbf{k}^\top \mathbf{x}_t$ with $\mathbf{k}^\top = [0.5955 \ 0.9764]$; likewise, the set X_f computed according to our assumptions, is given by

$$X_f = \left\{ \mathbf{x}_t \in \mathbb{R}^2 : \mathbf{x}_t^\top \begin{bmatrix} 114.21 & -29.45 \\ -29.45 & 208.67 \end{bmatrix} \mathbf{x}_t \leq 1 \right\}.$$

By suitable offline numerical procedures, we obtain the following bounds:

$$\bar{d} = 5.94 \cdot 10^{-4}, \quad \bar{q} = 2.75 \cdot 10^{-4}.$$

Hence, the constraint set X has been gridded in a uniform way obtaining a set X' made of $L = 1.4 \cdot 10^6$ sample states $\mathbf{x}^{(l)}$, $l = 1, \dots, L$.

The offline determination of the set $\Sigma^{RH} = \{(\mathbf{x}^{(l)}, \mu^{RH^o}(\mathbf{x}^{(l)})) : \mathbf{x}^{(l)} \in X'\}$ has been carried out in approximately 40 h of computational time on a Xeon 3 GHz CPU equipped with 4 GB RAM. In Fig. 10.6, the optimal RH control function is shown pictorially with evidence of the constraint set X and of the feasible set X^{RH} .

As FSP structure $\tilde{\mu}^{RH}(\mathbf{x}, \mathbf{w}_n)$, we choose an OHL network with a Radial Basis Function structure of the form (3.9) of the Gaussian type (3.10).

A trial-and-error procedure to solve Problem 10.2 has been carried out in order to find the appropriate model complexity n and location of the RBF centers, thus obtaining an RBF structure characterized by 453 centers suitably chosen within the set X^{RH} . The location of these centers is shown in Fig. 10.7.

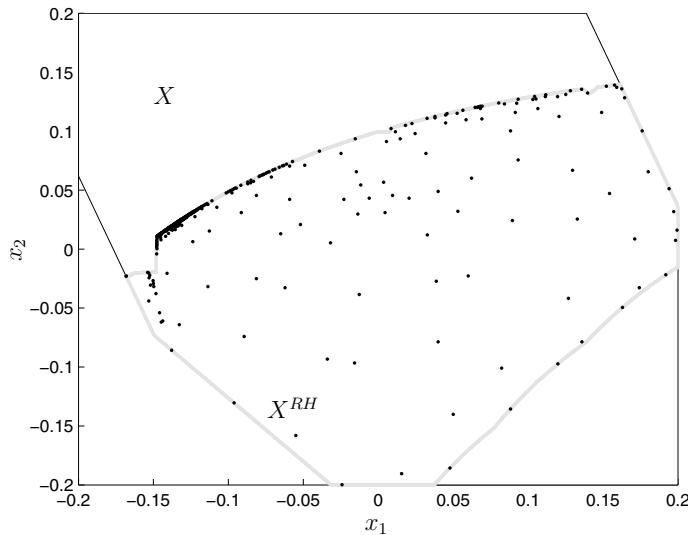


Fig. 10.7 A view of the distribution of the RBF centers over the feasible region X^{RH} ; notice that the density of centers increases in the zones of the domain in which the slope of the optimal RH control function is higher. ©2013 Taylor & Francis. Reprinted, with permission, from [25]

It is worth noting that the number of centers is significantly smaller than the number of state samples in the set X' which makes the storage requirements and

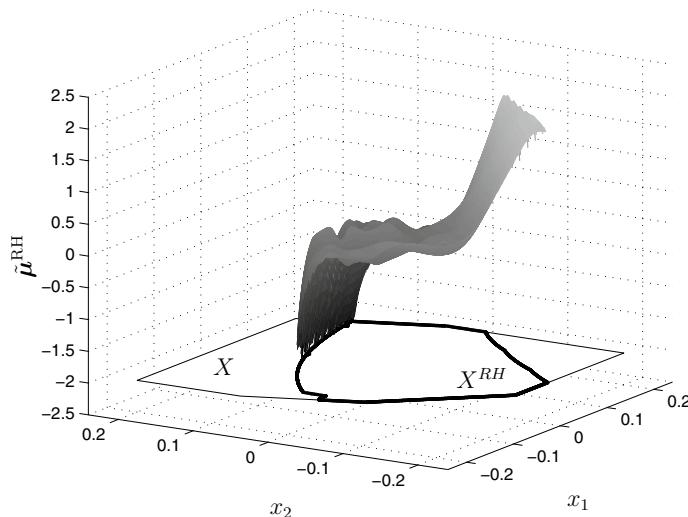


Fig. 10.8 The RBF approximate RH control function. ©2013 Taylor & Francis. Reprinted, with permission, from [25]

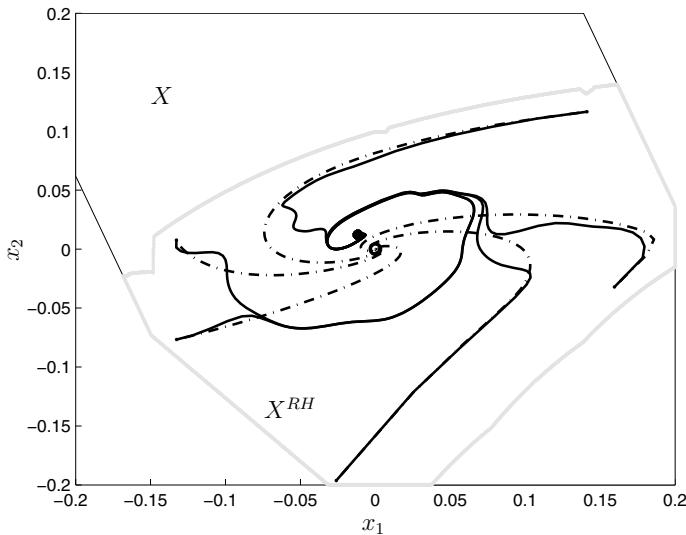


Fig. 10.9 The trajectories starting from five different initial states under the action of the optimal RH control law (dashed-dotted lines) are compared with those generated by the approximate RBF RH control function (solid lines). ©2013 Taylor & Francis. Reprinted, with permission, from [25]

online computational burden to compute the control action very manageable. In Fig. 10.8, the RBF approximate RH control function is drawn (compare with the optimal RH control function depicted in Fig. 10.6).

Finally, in Fig. 10.9, some state trajectories under the action of the optimal RH control law and the approximate RBFRH control law are shown. It is worth noting that the optimal RH control law achieves the asymptotic convergence of the trajectories to the origin, while the RBF RH control law drives the state to a point close to the origin (little circle) according to the ISpS property.

References

1. Arya A, Mount DM, Netanyahu NS, Silverman R, Wu AY (1998) An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *J ACM* 45:891–923
2. Bertsekas DP (2000) Dynamic programming and optimal control, vol 2. Athena Scientific
3. Bertsekas DP (2005) Dynamic programming and suboptimal control: a survey from ADP to MPC. *Eur J Control* 11:310–334
4. Bertsekas DP, Shreve SE (1978) Stochastic optimal control - the discrete time case. Academic Press
5. Bertsekas DP, Shreve SE (1979) Existence of optimal stationary policies in deterministic optimal control. *J Math Anal Appl* 69:607–620
6. Chakrabarty A, Dinh D, Corless MJ, Rundell AE, Zak SH, Buzzard GT (2017) Support vector Machine Informed Explicit Nonlinear Model Predictive Control Using Low-Discrepancy Sequences. *IEEE Trans Autom Control* 62:135–148

7. Franco E, Magni L, Parisini T, Polycarpou MM, Raimondo DM (2008) Cooperative constrained control of distributed agents with nonlinear dynamics and delayed information exchange: a stabilizing receding-horizon approach. *IEEE Trans Autom Control* 53:324–338
8. Garcia CE, Prett DM, Morari M (1989) Model predictive control: theory and practice - a survey. *Automatica* 25:335–348
9. Girosi F (1995) Approximating error bounds that use VC bounds. In: Proceedings of the international conference on artificial neural networks, pp 295–302
10. Gnecco G, Sanguineti M (2008) Estimates of the approximation error using Rademacher complexity: learning vector-valued functions. *J Inequalities Appl* 2008:1–16
11. Jiang ZP, Wang Y (2001) Input-to-state stability for discrete-time nonlinear systems. *Automatica* 37:857–869
12. Keerthi SS, Gilbert EG (1988) Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: stability and moving-horizon approximations. *J Optim Theory Appl* 57:265–293
13. Kwakernaak H, Sivan R (1972) Linear optimal control systems. Wiley
14. Lazar M, Heemels WPMH (2009) Predictive control of hybrid systems: input-to-state stability results for sub-optimal solutions. *Automatica* 45:180–185
15. Lazar M, Heemels WPMH, Bemporad A, Weiland S (2007) Discrete-time non-smooth nonlinear MPC: stability and robustness. In: Findeisen R, Allgöwer F, Biegler L (eds) Assessment and future directions of nonlinear model predictive control. Lecture notes in control and information sciences, vol 358. Springer, pp 93–103
16. Limón D, Alamo T, Camacho EF (2002) Input-to-state stable MPC for constrained discrete-time nonlinear systems with bounded additive uncertainties. In: Proceedings of the IEEE conference on decision and control, pp 4619–4624
17. Limón D, Alamo T, Salas F, Camacho EF (2006) Input to state stability of min-max MPC controllers for nonlinear systems with bounded uncertainties. *Automatica* 42:797–803
18. Magni L, Raimondo DM, Scattolini R (2006) Regional input-to-state stability for nonlinear model predictive control. *IEEE Trans Autom Control* 51(9):1548–1553
19. Mayne DQ, Michalska H (1990) Receding horizon control of nonlinear systems. *IEEE Trans Autom Control* 35:814–824
20. Mayne DQ, Rawlings JB, Rao CV, Scokaert POM (2000) Constrained model predictive control: stability and optimality. *Automatica* 36:789–814
21. Michalska H, Mayne DQ (1993) Robust receding horizon control of constrained nonlinear systems. *IEEE Trans Autom Control* 38:1623–1633
22. Niederreiter H (1992) Random number generation and quasi-Monte Carlo methods. SIAM
23. Parisini T, Sanguineti M, Zoppoli R (1998) Nonlinear stabilization by receding-horizon neural regulators. *Int J Control* 70:341–362
24. Parisini T, Zoppoli R (1995) A receding-horizon regulator for nonlinear systems and a neural approximation. *Automatica* 31:1443–1451
25. Pin G, Filippo M, Pellegrino FA, Fenu G, Parisini T (2013) Approximate model predictive control laws for constrained nonlinear discrete-time systems: analysis and off-line design. *Int J Control* 86:804–820
26. Pin G, Parisini T (2011) Networked predictive control of uncertain constrained nonlinear systems: recursive feasibility and input-to-state stability analysis. *IEEE Trans Autom Control* 56:72–87
27. Pin G, Raimondo DM, Magni L, Parisini T (2009) Robust model predictive control of nonlinear systems with bounded and state-dependent uncertainties. *IEEE Trans Autom Control* 54:1681–1687
28. Qin SJ, Badgwell TA (2003) A survey of industrial model predictive control technology. *Control Eng Pract* 11:733–764
29. Raimondo DM, Limon D, Lazar M, Magni L, Camacho EF (2009) Min-max model predictive control of nonlinear systems: a unifying overview on stability. *Eur J Control* 15:1–17
30. Rawlings JB, Mayne DQ, Diehl MM (2018) Model predictive control: theory, computation, and design. Nob Hill Publishing

Index

A

- Adjoint equation, 348, 397, 406
- ADP, *see* approximate dynamic programming
- \mathcal{M}^d -approximating FSP function, 68, 286
- \mathcal{M}^d -approximating sequence of sets, 68, 286
- \mathcal{S}^d -approximating sequence of sets, 75
- Approximate dynamic programming, 187, 255, 264, 299, 307, 392
 - approximation issues, 323
 - discretization issues, 320
 - error propagation, 316
- Approximate estimator, 416
- Approximating FSP function, 68, 98
- Approximation error, 152, 161, 256, 275, 285, 286
- Approximation of functions, 10
- Approximation rate, 89

B

- Backpropagation, 351, 395, 397, 403, 447, 463, 464
- Basis, 23
- Basis cardinality, 42, 119, 156, 172, 179, 180, 186
- Batch processing mode, 225, 228
- Bellman's principle of optimality, 262
- Best approximation, 67, 114
- Bias-variance dilemma, 164

C

- CE, *see* certainty equivalence principle
- CEOLF control, *see* certainty equivalent open-loop feedback control

Certainty equivalence principle, 336, 385, 408

Certainty equivalent open-loop feedback control, 423

Control parametrization Ritz method, 136

Convergence

- Robbins–Monro algorithm, 235
- general gradient algorithm, 231

Curse of dimensionality, 7, 14, 40, 51, 64, 65, 73, 84, 90, 94, 136, 208, 210, 268, 272, 282, 288, 300, 308, 341, 388, 501, 503, 506

comparison between the ERIM and the Ritz method, 135

- first, 312

- fourth, 312

- in the Ritz method, 135, 141

- second, 312

- third, 312

D

- Data error, 162
- Decision function, 5
- Decision-maker, 2, 40, 299, 383, 427, 486
- Deep network, 127
- Deterministic learning theory, 16, 152, 155, 187, 207
- Deterministic optimal control over an infinite horizon, 471, 473
- Problem C1-IH (closed-loop form), 472, 477, 478, 480
- Problem C1-IH (open-loop form), 472, 474, 475
 - stationary optimal control law, 476–478
- Deviation, 70
- Dimension-independent estimate, 128

Direct search method, 51, 208, 222, 340
 Discrepancy, 152, 190
 Dispersion, 249, 502, 506
 DLT, *see* deterministic learning theory
 DM, *see* decision-maker
 DP, *see* dynamic programming
 Dynamic programming, 17, 59, 207, 255, 257, 261, 300, 384, 439, 472
 approximation issues, 323
 backward procedure, 261
 discretization issues, 320
 error propagation, 316
 forward procedure, 261
 recursive equation, 265
 Dynamic routing in communication networks, *see* dynamic routing in traffic networks
 Dynamic routing in traffic networks, 443, 456

E

ε -near minimum point, 143
 ε -pc \mathcal{M}^d -approximating FSP function, 72, 286
 ε -pc P^d -optimizing FSP function, 63
 ε -polynomially complex \mathcal{M}^d -approximating sequence of sets, 72, 286, 318
 ε -polynomially complex P^d -optimizing sequence, 63
 Empirical risk, 152, 158, 165, 187, 189, 198, 203
 Epi-convergence, 54
 ERIM, *see* extended Ritz method
 ERM principle, *see* principle of empirical risk minimization
 Estimation error, 152, 162, 166, 171, 256, 275, 285, 289, 291
 Euler–Lagrange equation, 294
 EVPI, *see* expected value of perfect information
 Expected risk, 152, 156, 187, 198, 203
 Expected value
 computation, 207, 217, 219, 311
 computation by MC method, 219
 computation by quasi-MC method, 218, 219
 of a function, 51, 207
 Expected value of perfect information, 375
 Extended Ritz method, 7, 39, 59, 137, 141, 158, 208, 255, 299, 320, 333, 384, 391, 428, 446, 460, 466, 473, 487, 489, 495, 499, 500

comparison with the classical Ritz method, 89, 135
 rate of approximate optimization, 89, 142

F

FDO, *see* finite-dimensional optimization
 FH, *see* finite horizon
 Final time, 257
 Finite-dimensional optimization, 3, 23, 25, 140, 158, 257, 299, 443
 Finite horizon, 17, 471
 Fixed-structure parametrized function, 7, 39, 41, 42, 151, 255, 300, 307, 325, 329, 384, 391, 401, 415, 428, 442, 446, 473, 490, 499, 500
 Freeway traffic optimal control, 418
 FSP, *see* fixed-structure parametrized function
 Full uniform grid, 268, 308, 321
 Functional optimization, 3, 299
 Function approximation, 9, 10
 Function approximation problem, 67

G

\mathcal{G} -variation norm, 130
 estimation of the \mathcal{G} -variation, 133
 General gradient algorithm, 229, 231
 Generalization error, 152, 160, 161, 164, 187, 256, 284–286
 for Gaussian OHL networks, 176
 for Sigmoidal OHL networks, 181
 Global model complexity, 47, 392, 443
 Global optimization, 242
 Gradient-based descent method, 208
 Gradient-based method, 51, 223, 224, 340, 341, 393

H

Hypothesis class, 156, 158, 162, 164, 174, 177, 181

I

IDO, *see* infinite-dimensional optimization
 IH, *see* infinite horizon
 Infinite-dimensional optimization, 3, 20, 23, 25, 89, 156, 207, 257, 299, 384, 443
 Infinite horizon, 1, 471
 Information state vector, *see* information vector
 Information vector, 383, 385, 463, 485
 Input-to-state stability, 472, 497

- Integration, 207
 by MC method, 210, 219
 by quasi-MC method, 214, 219
 by regular grids, 209
- ISpS, *see* regional input-to-state practical stability
- ISS, *see* input-to-state stability
- J**
- Jackson rate, 125
- Jackson-type bound, *see* Jackson rate
- K**
- Kernel function, 102
- Kernel smoothing model, 43, 101, 278, 314
- KH inequality, *see* Koksma–Hlawka inequality
- Koksma–Hlawka inequality, 197, 213, 214, 291, 313, 341
- Kolmogorov n -width, 113
- KSM, *see* kernel smoothing model
- L**
- LCFBF, *see* linear combination of fixed-basis functions
- Limited-memory information vector, 384, 398, 485
- Linear combination of fixed-basis functions, 43, 91, 170, 186
- Linear-quadratic, 255
- Linear-quadratic-Gaussian, 1, 19, 384
- LM, *see* limited-memory information vector
- Loss function, 165
- Low-discrepancy sequence, 152, 199, 202, 204, 214, 215, 221, 248, 251, 269, 292, 314, 322, 341, 362, 388, 503, 506
- LQ, *see* linear-quadratic
- LQG, *see* linear-quadratic-Gaussian
- M**
- Mathematical programming, 3
- Maurey–Jones–Barron lemma, *see* Maurey–Jones–Barron theorem
- Maurey–Jones–Barron theorem, 90, 114, 115, 183, 455
- extension, 128
- for sigmoidal OHL network, 119
- geometric rate of approximation, 129
- upper bound in terms of the \mathcal{G} -Variation Norm, 130
- MC, *see* Monte Carlo method
- Memory function, 399
- Memory vector, 384, 398, 424, 486
- MHL, *see* multi-hidden-layer network
- Minimax control, 306
- MJB theorem, *see* Maurey–Jones–Barron theorem
- MLFNN, *see* multi-hidden-layer network
- Model complexity, 9, 39, 42, 47, 58, 60, 62, 71, 72, 90, 93, 113, 124, 172, 268, 287, 318, 334, 335, 392, 503
- Model predictive control, 13, 407, 472
- Modulus of continuity, 143
- Modulus of convexity, 143
- Momentum, 241
- Monte Carlo estimate, 213
- Monte Carlo method, 16, 51, 152, 155, 208, 212, 269, 300, 308, 311, 340
- Monte Carlo random search, 245
- Monte Carlo search for minima, 245
- MPC, *see* model predictive control
- Multi-hidden-layer network, 9, 43, 99, 349, 384, 394, 402, 416
- approximation accuracy, 127
- N**
- n -width, *see* Kolmogorov n -width
- Nadaraya–Watson kernel-weighted average, 102
- NDP, *see* neuro-dynamic programming
- Nestedness property, 42, 46, 105, 158, 174
- Neural network, 100
- Neuro-dynamic programming, 326
- NLP, *see* nonlinear programming
- Nonlinear programming, 6, 158, 256, 292, 299, 384, 443
- Numerical computation of integrals, 208
- O**
- OA, *see* orthogonal array
- Offline design of the stabilizing approximate RH control law, 502
- sampling the state space using full uniform grids, 503
- sampling the state space using low-discrepancy sequences, 503
- sampling the state space using MC method, 503
- OHL, *see* one-hidden-layer network
- OHL approximating sequence of sets, 114

- One-hidden-layer network, 9, 42, 43, 77, 78, 92, 101, 151, 189, 271, 325, 349, 354, 362, 418, 460, 493, 506
 \mathcal{C} -density property, 106
 \mathcal{L}_p -density property, 106
density property, 106
extension to multi-hidden-layers and multiple outputs, 112
lower bound on approximation rate, 123
radial OHL network, 78, 111, 152, 176, 196, 290, 506
ridge OHL network, 107, 127
sigmoidal OHL network, 78, 119, 152, 181, 186, 195, 290, 506
upper bound on approximation rate, *see* Maurey–Jones–Barron theorem
- Optimal control
finite horizon, stochastic, with imperfect state information, 383
finite horizon, stochastic, with perfect state information, 299
- Optimal regulation problem, 476, 479
- Optimizing FSP function, 56
- Optimizing sequence, 56
- Orthogonal array, 321
- P**
 P^d -optimizing sequence of FSP functions, 75, 78
Parametrized basis function, 92
Parametrized control, 13
Pc-optimizing FSP function, 63
Pc-optimizing sequence, 63
Polynomially complex P^d -optimizing sequence of FSP functions, 78
P-optimizing FSP function, 55
P-optimizing sequence, 55
Principle of empirical risk minimization, 159, 166, 167, 173
Principle of structural risk minimization, 175
Problem P, 44, 85
Problem PM, 46
Problem BB, 448
Problem C1', 267
Problem C2'-minimax, 306
Problem C1, 17, 255, 256, 480
solution by dynamic programming, 264
Problem C1tf, 297
Problem C1* (closed-loop form), 489
Problem C1* (open-loop form), 489
Problem C1-IH, *see* deterministic optimal control over an infinite horizon
- Problem C1-RH, 490
Problem C2, 17, 299, 302
solution by dynamic programming, 302
Problem C2', 299, 302
solution by dynamic programming, 302
Problem C2n, 300, 334, 340
Problem C2'n, 336, 338
Problem C2-IH, 483
Problem C3, 18, 384, 386, 442
approximate solution by the ERIM, 391
solution by dynamic programming, 386
Problem C3n, 392
Problem C3n-LM, 402
Problem C3-IH, 486
Problem C3-LM, 399, 401
approximate solution by the certainty equivalence principle and the ERIM, 407
approximate solution by the ERIM, 401
Problem DC, 440, 441
Problem DCn, 443
Problem P, 14, 21, 39, 54, 94, 138, 156, 225
Problem Pn, 45, 49, 50, 207, 221–223, 334, 392, 401
Problem PA_n^d, 67, 90, 112, 118
Problem PBP, 433
Problem PM, 14, 22, 85, 302, 386, 400, 430, 460
Problem ROUT, 443, 459
approximate solution by the ERIM, 460
Problem ROUTn, 462
Problem T, 430, 433, 459
Problem W, 443, 444
approximate solution by the ERIM, 446
Problem Wn, 447
- Q**
Q-function, 331
Q-learning, 331
Quasi-MC, *see* quasi-Monte Carlo method
Quasi-Monte Carlo estimate, 214
Quasi-Monte Carlo method, 16, 51, 152, 155, 199, 207, 208, 218, 256, 269, 300, 308, 311, 322, 340
Quasi-Monte Carlo random search, 248
Quasi-Monte Carlo search for minima, 245
- R**
Receding-horizon, 472
approximate control law, 472, 487, 489, 490, 492–494, 497, 498, 500

- approximation of the IH optimal control law, 472, 487, 490
control law approximated by an FSP function, 500
control scheme, 19, 472, 487
optimal control law, 472, 490
optimal control problem, 472, 488
stabilizing approximate control law, 499, 500
stabilizing properties of RH control laws, 472, 490
Regional input-to-state practical stability, 493, 495, 501
Regional input-to-state stability, 472, 490, 492, 493, 497, 498
Regional ISS, *see* regional input-to-state stability
Regression function, 156
Regular grid, 208, 311
Regulation problem, 471
Reinforcement learning, 326
Reoptimization, 262, 282, 315
RH, *see* receding-horizon
Ritz method, 7, 141
RL, *see* reinforcement learning
Robbins–Monro algorithm, 226, 228, 229, 238
Root-finding problem, 226
- S**
SA, *see* stochastic approximation
Sample cardinality, 152, 158, 173, 178, 180, 184, 268, 269, 292, 321
Sampling of the state space, 307
Search for minima, 207, 208
Separation property, 400, 411, 417
Set-membership constraint for the disturbance, 303
Shallow network, 127
SLT, *see* statistical learning theory
Sobol's low-discrepancy method, 199, 322, 362
SRM principle, *see* principle of structural risk minimization
Star discrepancy, 190, 197, 200, 207, 214, 291, 506
State increment dynamic programming, 321
Statistical learning theory, 151
Stepsize sequence, 237
Stochastic approximation, 208, 222, 223, 242, 393, 402
convergence, 233, 239
- Stochastic gradient algorithm, 52, 225, 229, 342, 363, 370, 393, 402, 415, 418, 443, 447, 448, 462, 465
Stochastic optimal control over an infinite horizon, 473
with imperfect state information, 473, 484
with perfect state information, 473, 481
Support vector machine, 101
SVM, *see* support vector machine
- T**
Team optimal control problem, *see* Problem T
Team theory, 427
basic concepts, 428
dynamic team, 428, 431, 437, 438
information structure, 428, 429
linear information structure, 429
LQG dynamic team with partially nested information structure, 437
LQG static team, 435
memory communication diagram, 432
optimal decentralized control, 435, 439
partially nested information structure, 428, 433
person-by-person optimality, 429, 433, 434
precedence diagram, 431, 432
sequential partition, 438
static team, 35, 428, 430, 431, 435
- U**
Universal approximation property, 98, 106, 111
- V**
Vapnik–Chervonenkis dimension, *see* VC dimension
Variation, 152
in the sense of Hardy and Krause, 204, 214, 218, 290, 313
in the sense of Vitali, 192
of a multivariable function, 191
VC dimension, 167, 168, 173, 174
- W**
Weierstrass' theorem, 69
Witsenhausen counterexample, 2, 428, 443
improved numerical results, 452
variants, 452