Machine Learning for Fluid Dynamics

von Karman Institute Lecture Series

Edited by

M. A. Mendez and A. Parente

Preview Sample

Sint-Genesius-Rode, Belgium



©2024 by the von Karman Institute for Fluid Dynamics, publisher
Waterloosesteenweg 72, B-1640 Sint-Genesius-Rode
To order: https://www.vki.ac.be

von Karman Institute Lecture Series ISSN 0377-8312

International Standard Book Number-13: 978-2-87516-209-0 (Hardback)

D/2024/0238/778, M. A. Mendez & A. Parente, Editors-in-Chief

This book contains information from authentic and highly regarded sources. Reasonable efforts have been made to publish reliable data and information, but the authors and publisher cannot assume responsibility for the validity of all materials or the consequence of their use. The authors and publisher have attempted to trace the copyright holders of all material reproduced in this publication.

No warranty may be created or extended by sales representatives, written sales materials or promotional statements for this work. The fact that an organization, website, or product referred to in this work as a citation and/or potential source for further information does not mean that the publisher and authors endorse the information or services the organization, website, or product may provide or recommendations it may make. This work is sold with the understanding that the publisher is not engaged in rendering professional services. The advice and strategies contained herein may not be suitable for your situation. You should consult with a specialist where appropriate. Further, readers should be aware that the websites listed in this work may have changed or disappeared between when this work was written and when it is read. Neither the publisher nor authors shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

The contents of this book are the property of the von Karman Institute for Fluid Dynamics. No part of this book may be reproduced, transmitted or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission from the publisher.

Preview Sample

Contents

1 Machine Learning for Scientific Discovery **13**
1.1 Introduction . . . 13
1.2 Physics Informed Machine Learning for Fluid Mechanics . . . 15
1.2.1 The problem . . . 16
1.2.2 The data . . . 18
1.2.3 The architecture . . . 19
1.2.4 The loss function . . . 22
1.2.5 The optimization algorithm . . . 23
1.3 Sparse identification of nonlinear dynamics (SINDy) . . . 25
1.4 Parting Thoughts . . . 30

2 Fundamentals of Regression **33**
2.1 A note on notation and style . . . 33
2.2 General Concepts . . . 35
2.2.1 A probabilistic perspective: The MLE . . . 36
2.2.2 Bootstrapping and cross-validation . . . 39
2.2.3 More Cost Functions . . . 45
2.2.4 Methods for parametric regression . . . 46
2.2.5 Methods for non-parametric regression . . . 54
2.3 Data driven... Scientific computing . . . 59
2.4 Summary and Conclusions . . . 64

3 Learning with Learning with Physical Constraints **65**
3.1 Problem 1: "Fill the Gaps" and super-resolution . . . 65
3.1.1 General Context . . . 65
3.1.2 Proposed Exercise . . . 66
3.1.3 Methodology and Results . . . 67
3.2 Problem 2: Turbulence Modeling. What is the Reynolds Stress? 72
3.2.1 General Context . . . 72
3.2.2 Proposed Exercise . . . 74
3.2.3 Methodology and Results . . . 76
3.3 Problem 3: Forecasting and Control . . . 80
3.3.1 General Context . . . 80
3.3.2 The adjoint-based Idendification Problem . . . 81
3.3.3 Proposed exercise . . . 84

Preview Sample

3.4 Conclusions and outlook 86

4 Data-driven Turbulence Modeling 89
4.1 Introduction 89
4.2 RANS models: derivation and uncertainties 94
4.3 State of the Art 98
4.4 Parametric approaches 101
4.4.1 Data-driven update of turbulence model closure coefficients 101
4.4.2 Accounting for uncertain model structure: Bayesian model selection and averaging 107
4.5 Non-parametric Approaches 111
4.5.1 Representation of the data-driven corrections 116
4.5.2 Field inversion and Machine Learning (FIML) approach . 120
4.5.3 Physics-informed correction of Reynolds stress discrepancies 122
4.5.4 Data-driven RANS modeling through Symbolic Regression 125
4.5.5 GEP regression of data-driven RANS closures 131
4.5.6 SpaRTA: Sparse regression of Reynolds sTress Anisotropy 132
4.6 Sample results 136
4.7 Conclusions and research trends 141

5 Prediction of Chaotic Dynamics from Data: An introduction 149
5.1 Chaotic dynamical systems 149
5.1.1 Dynamical systems' equations 150
5.1.2 Linear analysis 151
5.1.3 Largest (dominant) Lyapunov exponent 152
5.1.4 Lyapunov spectrum 156
5.1.5 Metrics and indicators of chaos 160
5.2 Machine learning for dynamical systems 163
5.3 Recurrent Neural Networks 164
5.4 Echo state networks 164
5.4.1 The dynamical systems' interpretation of ESNs 164
5.4.2 Architecture 166
5.4.3 Training 168
5.4.4 ESN variants 172
5.4.5 Closed-loop 174
5.4.6 Validation 174
5.4.7 Jacobian of the ESN 179

Preview Sample

5.5 Long short-term memory network . . . 180
5.5.1 Architecture . . . 180
5.5.2 Closed-loop . . . 182
5.5.3 Physics-informed architecture (PI-LSTM) . . . 183
5.5.4 Data Preparation . . . 184
5.5.5 Jacobian of the LSTM . . . 185
5.6 Tutorial: Lorenz system . . . 186

6 Reinforcement Learning for Fluid Mechanics 195
6.1 Flow control and reinforcement learning . . . 195
6.1.1 Standard approaches in flow control: a brief overview . . . 198
6.1.2 Reinforcement learning in flow control . . . 200
6.1.3 Organization of the chapter . . . 202
6.2 From nonlinear control to reinforcement learning . . . 202
6.2.1 Nomenclature and state space representation . . . 202
6.2.2 Optimal control in linear system: the Riccati equation . . . 205
6.2.3 The Hamilton-Jacobi-Bellman equation . . . 207
6.2.4 Discrete systems . . . 211
6.2.5 Bellman's principle of optimality . . . 212
6.2.6 Approximating the Bellman equation: iterative methods . . . 215
6.2.7 Reinforcement learning (RL) . . . 220
6.3 Elements of reinforcement learning in practice . . . 221
6.3.1 A short glossary . . . 222
6.3.2 Identifying the policy . . . 223
6.3.3 Numerical approximations . . . 226
6.4 Essential bibliography and final remarks . . . 227

7 Model Predictive Control 231
7.1 What is Model Predictive Control? . . . 231
7.1.1 The rational behind MPC . . . 232
7.1.2 Main advantages and limitations . . . 233
7.1.3 Towards application to chaotic high-dimensional systems and fluid flows . . . 235
7.2 A glimpse of MPC of linear systems . . . 237
7.2.1 Problem formulation . . . 237
7.2.2 Solution via batch approach . . . 239
7.3 Nonlinear MPC . . . 241

Preview Sample

7.3.1 Problem formulation . . . 241
7.3.2 A note on feasibility and stability of MPC . . . 241
7.3.3 Hyperparameter tuning . . . 242
7.4 Application of self-tuning MPC to wake control . . . 243
7.4.1 The fluidic pinball . . . 244
7.4.2 The MPC framework . . . 245
7.4.3 Optimization of hyperparameters and plant model . . . 246
7.4.4 Main results and discussion . . . 247
7.5 Conclusion and outlook . . . 249

8 Fundamentals of Dimensionality Reduction 253
8.1 A note on notation and style . . . 253
8.2 General Concepts: Autoencoders and Manifold Learning . . . 254
8.3 Linear autoencoding and manifold learning: PCA . . . 256
8.4 Data Driven Modal Analysis . . . 259
8.4.1 From Autoencoders to Modal Analysis . . . 260
8.4.2 The Dynamic Mode Decomposition . . . 264
8.4.3 The Spectral POD by Sieber et al. (2016) . . . 266
8.4.4 The Spectral POD by Towne et al. (2018) . . . 267
8.4.5 The Multiscale POD . . . 268
8.5 Nonlinear Methods . . . 271
8.5.1 ANNs Autoencoders . . . 271
8.5.2 Kernel PCA and Kernel POD . . . 272
8.5.3 Isometric Mapping (ISOMAPs) . . . 275
8.5.4 Locally Linear Embedding (LLE) . . . 278
8.5.5 t-Distributed Stochastic Neighbor Embedding (t-SNE) . . . 279
8.6 Problem 1: Modal Analysis of an impinging jet . . . 281
8.7 Problem 2: Manifold Learning of a cylinder wake . . . 289
8.8 Summary and Conclusions . . . 293

9 Data-driven post-processing and reduced order modeling 295
9.1 Introduction . . . 295
9.2 Methodology . . . 298
9.2.1 Modal decomposition . . . 300
9.2.2 Hybrid physics-aware deep learning models . . . 311
9.3 Applications . . . 321
9.3.1 Pattern identification . . . 321

Preview Sample

9.3.2 Data reconstruction . . . 327
9.3.3 Data forecasting . . . 329
9.4 Conclusions . . . 333

10 Data-Driven Modeling for Enhanced Aerodynamic 335
10.1 Motivation . . . 336
10.2 Machine Learning for Efficient Design . . . 336
10.2.1 Surrogate-based Optimization . . . 337
10.2.2 An Airfoil Use Case . . . 340
10.2.3 Optimization Results . . . 342
10.2.4 Surrogate-based Uncertainty Quantification . . . 345
10.2.5 Uncertainty Quantification Results . . . 346
10.2.6 Robust Design Results . . . 347
10.3 Potentials and Limits of Machine Learning for Applied Aerodynamics . . . 349
10.3.1 Data Driven Turbulence Modeling . . . 352
10.3.2 Graph-Neural Networks for Pressure Predictions . . . 353
10.3.3 Physics-Informed Neural Networks . . . 355
10.4 Conclusions . . . 358

11 Dimensionality reduction, classification and reduced-order modelling for reacting flow simulations 359
11.1 Introduction . . . 360
11.2 Feature extraction using dimensionality reduction techniques . . . 362
11.2.1 Combustion data sets . . . 362
11.2.2 Theory . . . 363
11.2.3 Results . . . 366
11.3 Transport of principal components . . . 372
11.3.1 Regression Models . . . 373
11.3.2 Validation in simple reactors . . . 375
11.3.3 Validation in LES simulations . . . 380
11.4 Chemistry acceleration via adaptive-chemistry . . . 383
11.4.1 Introduction . . . 385
11.4.2 Results . . . 387
11.5 Digital twin of a combustion furnace . . . 392
11.5.1 ROM development . . . 394
11.5.2 Reconstruction of test data . . . 394

Preview Sample

11.5.3 Prediction of new data 397
11.5.4 ROM testing 399
11.6 Self-updating digital twins 401
11.6.1 Recursive Kalman Filter 403
11.6.2 Steady-state Kalman Filter 406
11.6.3 Sensitivity analysis 410

12 Digital Twins: Challenges and Enablers 413
12.1 Digital Twin: Taxonomy and Terminology 413
12.2 Capability Levels of a Digital Twin 415
12.3 The Exchange between Digital Twins and Fluid Dynamics 416
12.4 Challenges 417
12.5 Enablers 421
12.6 Prospects and Outlook 426

Preview Sample

Preface

This book originated as a collection of lecture notes from the VKI lecture series *Machine Learning for Fluid Dynamics*, jointly organized by the von Karman Institute (VKI) and the Université Libre de Bruxelles (ULB) and held in Brussels from January 29 to February 2, 2024. The course program, the recording of some of the lectures, and additional information on all course editions are available on the course webpage `https://www.datadrivenfluidmechanics.com/`.

Book Overview

The book is organized around four main topics: (1) modeling (Chapters 2 to 5), (2) control (Chapters 6 and 7), (3) dimensionality reduction (Chapters 8 and 9), and (4) applications in aerodynamics and reactive flows (Chapters 10 and 11). These core sections are framed by an introductory chapter on the machine learning workflow (Chapter 1) and a concluding chapter discussing the future of Digital Twinning technology (Chapter 12).

Chapter 1 with a high-level overview of machine learning for scientific discovery by Steve Brunton. This chapter reviews the process of formulating a machine-learning problem for scientific purposes in its five key stages: problem formulation, data collection, model architecture, loss function design, and optimization algorithm selection. It emphasizes the role of physics-informed machine learning in enhancing model accuracy and offers an example of sparse identification of nonlinear dynamics (SINDy).

The session on modeling opens in **Chapter 2**, by Miguel A. Mendez, which delves into the fundamentals of regression, covering parametric and non-parametric regression methods, the notion of maximum likelihood estimation, bootstrapping and cross-validation, and the definition of various cost functions. It closes with the applications of these tools for data-driven scientific computing. This chapter is complemented by **Chapter 3**, which bridges theory with hands-on applications in data-driven fluid mechanics. This chapter, by Miguel A. Mendez and co-workers, provides three Python tutorials in physics-constrained regression. Three exercises are considered, from "gap filling" and super-resolution to turbulence modeling and real-time data assimilation. **Chapter 4**, by Paola Cinnella, explores data-driven approaches for identifying turbulent stress closures using high-fidelity simulation and experimental data, focusing on Reynolds-Averaged Navier-Stokes (RANS) modeling. Bayesian methods and non-parametric approaches are introduced to handle uncertainty and

Preview Sample

enhance model adaptability, illustrated through several case studies highlighting key research trends in the field. **Chapter 5**, by Luca Magri and co-workers, dives into the prediction with chaotic dynamical systems. Starting from an introduction to key concepts such as Lyapunov exponents to measure sensitivity to initial conditions, this chapter introduces machine learning methods such as Recurrent Neural Networks (RNN) and Echo State Networks (ESNs) and their applications to forecast short-term behaviors for seemingly unpredictable systems.

Within the session on flow control, **Chapter 6**, by Onofrio Semeraro, provides an introduction to Reinforcement Learning (RL) and its potential for flow control. This chapter links fundamental notions of RL –such as the Bellman equation, rewards, and policies– to more traditional optimal control settings and its standard tools – such as the Riccati equation and Linear Quadratic Regulators. It also provides a broad literature review and perspective on its applications in fluid mechanics. **Chapter 7**, by Stefano Discetti and co-workers, complements the tour in control application with an introduction to Model Predictive Control (MPC) for linear and nonlinear systems. This chapter features a case study on wake control to illustrate how MPC can stabilize fluid flows by predicting and adjusting control actions in real time, making it valuable for applications requiring high responsiveness.

Within the session on dimensionality reduction, **Chapter 8**, by Miguel A. Mendez and co-workers, reviews the fundamentals of dimensionality reduction, starting from the classic Principal Component Analysis (PCA) and moving towards classic data-driven decompositions and more complex autoencoders and manifold learning methods. **Chapter 9**, by Soledad Le Clainche and co-workers complements the session on dimensionality reduction with an overview of hybrid linear and nonlinear methods, combining traditional modal analysis with deep learning architecture to enhance pattern identification performances.

In the session on applications, **Chapter 9**, by Philippe Bekemeyer and co-workers, gives a broad overview of machine learning methods for aerodynamic design optimization, focusing on surrogate models for computational efficiency, uncertainty quantification, and robust design. The presentation is enriched by an example case study on airfoil design to illustrate how machine learning can create faster and more efficient optimization workflows. **Chapter 10**, by Alessandro Parente and co-workers, presents recent advances in dimensionality reduction and classification for reactive flows, such as those encountered in combustion and chemical reactions. Several topics linked to the aim of reducing computational

Preview Cambridge

costs are discussed: from feature extraction to transport of principal components and chemistry acceleration techniques.

Finally, the book closes with **Chapter 12** by Omer San, exploring the notion of digital twins, their potential, challenges, and enabling technologies. The chapter introduces the main scopes of digital twinning, including predictive maintenance, operational efficiency, and performance optimization, and discusses how these could be applied in complex fluid systems like engines, wind turbines, and HVAC systems.

Miguel A. Mendez
von Karman Institute for Fluid Dynamics

Alessandro Parente
Université libre de Bruxelles

Preview Sample

About the VKI lecture series

The von Karman Institute (VKI) Lecture Series offers specialized, intensive courses in fluid dynamics, aerodynamics, thermodynamics, and computational methods designed for graduate students, researchers, and industry professionals. Each series focuses on a single, advanced topic, taught by leading experts actively contributing to cutting-edge research. Known for its immersive approach, VKI combines lectures with hands-on sessions, case studies, and practical labs, particularly in computational areas like CFD and, more recently, machine learning for fluid dynamics. These courses deepen technical expertise and foster international collaboration and networking, drawing a global audience each year. VKI's lecture series is widely recognized for its high-quality, specialized content, providing participants with unparalleled learning experiences in applied physics and engineering. For more information about the upcoming VKI lecture series and the VKI educational programs, please visit `www.vki.ac.be`.

About the Lecturers and Authors

Steven L. Brunton (Senior Member, IEEE) earned his B.S. in mathematics from Caltech in 2006, focusing on control and dynamical systems, and his Ph.D. in mechanical and aerospace engineering from Princeton in 2012. He is a Professor of Mechanical Engineering and Data Science Fellow at the Science Institute, University of Washington. Renowned for his work in sparse sensing, Koopman operator theory, and machine learning methods like dynamic mode decomposition (DMD) and sparse identification of nonlinear dynamics (SINDy), Prof. Brunton's research spans diverse scientific and engineering applications. He is also known for his educational outreach, making computational science widely accessible through online courses and tutorials.

Miguel A. Mendez received his Ph.D. in Engineering Science from the Université Libre de Bruxelles (ULB) in 2018. He is an Associate Professor at the von Karman Institute for Fluid Dynamics (VKI), where he teaches fluid modeling, measurement techniques, and machine learning for fluid flows. His research focuses on coating flows, experimental fluid mechanics, reduced-order modeling, and flow control in low-speed and cryogenic applications. In 2021, he launched the VKI lecture series *Hands-On Machine Learning for Fluid Dynamics*. He is currently leading the ERC Starting Grant project *RE-TWIST*, integrating reinforcement learning and model-based control with real-time data assimilation.

Preview Sample

Paola Cinnella is a full professor at the Institut Jean Le Rond D'Alembert, Sorbonne Université, France. She graduated summa cum laude in Mechanical Engineering from Politecnico di Bari (Italy) in 1995 and earned her Ph.D. in Fluid Mechanics (summa cum laude) from ENSAM, Paris, in 1999. She is Editor-in-Chief of *Computers & Fluids* and serves as Associate Editor for the *International Journal of Heat and Fluid Flow* and as board member of *Flow, Turbulence and Combustion*. Her research focuses on computational fluid dynamics (CFD), covering high-order methods, uncertainty quantification, data-driven modeling, and applications in compressible flows with real gas effects.

Luca Magri is a Full Professor at Imperial College London, where he holds the first chair in Scientific Machine Learning. He is also a Fellow and Group Leader at The Alan Turing Institute and a Full Professor of Fluid Dynamics at Politecnico di Torino. His research bridges the gap between physical sciences and artificial intelligence, focusing on optimizing complex engineering systems for net-zero aerospace propulsion, healthcare, and more. He also explores quantum algorithms for nonlinear equations. Magri's innovative work has garnered significant recognition, including awards such as the ERC Starting Grant and the Royal Academy of Engineering Fellowship.

Soledad Le Clainche is Professor of Applied Mathematics at the School of Aeronautics of the Universidad Politécnica de Madrid (UPM). She earned her Ph.D. in 2013 at the same university, in the Department of Fluid Dynamics and Aerospace Propulsion. Her research focuses on computational fluid dynamics, data analysis, machine learning, and the development of predictive reduced-order models based on physical principles. She leads several national and EU-funded projects aimed at developing strategies to reduce air pollution, improve combustion systems and aerodynamic designs, and advance personalized medicine through data-driven modeling.

Stefano Discetti is Full Professor in the Department of Aerospace Engineering of Universidad Carlos III de Madrid. He received his Ph.D. in Aerospace Engineering from the University of Naples and was a visiting researcher at Arizona State University. He serves on the Editorial Board of *Measurement Science and Technology*. His research interests include non-intrusive measurement techniques in aerodynamics, machine learning-based flow control, and low-order modelling techniques. He was awarded a Starting Grant from the European

Preview Sample

Research Council for his project *NEXTFLOW*, developing and applying machine learning to optical flow diagnostics and flow control.

Onofrio Semeraro graduated in Aerospace Engineering from the University of Naples (Italy) in 2008 and received his Ph.D. in Mechanical Engineering from KTH Royal Institute of Technology (Sweden) in 2013. He is currently a Research Scientist at the Centre national de la recherche scientifique (CNRS) and is associated with Université Paris-Saclay. His research focuses on modeling and control of fluid flows, combining system identification and model reduction with data-driven and reinforcement learning techniques.

Philipp Bekemeyer leads the "Surrogates and Uncertainty Management" group at the German Aerospace Center's (DLR) Institute of Aerodynamics and Flow Technology. He holds bachelor's and master's degrees in aerospace engineering from TU Braunschweig, Germany, and a Ph.D. from the University of Liverpool, UK. With his team, he explores how machine learning can address key aerodynamic challenges, from wind turbines in subsonic conditions to commercial aircraft at transonic and supersonic regimes, aiming to industrialize ML techniques across the aerospace sector.

Alessandro Parente earned his Master's (2005) and Ph.D. (2009) in Chemical Engineering from the Università di Pisa. He was a Research Associate at the University of Utah (2007–2009) and joined the von Karman Institute in 2009. Since 2010, he has been at the Université Libre de Bruxelles, where he became a Full Professor in 2019. Co-chair of the Brussels Institute for Thermal-Fluid Systems and Clean Energy (BRITE) since 2021, his research spans experimental and numerical studies of reacting and non-reacting flows, with applications in air quality and industrial decarbonization. He also leads efforts in machine learning for fluid simulations and digital twins, coordinating the European network *cypher.ulb.be* for decarbonizing hard-to-abate sectors.

Omer San graduated in Aeronautical Engineering from Istanbul Technical University in 2005 and earned his Ph.D. in Engineering Mechanics from Virginia Tech in 2012. He is currently an Associate Professor of Aerospace Engineering at the University of Tennessee, Knoxville. His research focuses on advancing computational methods in science and engineering, with emphasis on scientific machine learning for fluid dynamics across various scales. He has led or co-led projects funded by DOD, DOE, NSF, NASA, ASHRAE, the Research Council of Norway, and NVIDIA.

Preview Sample

1

Machine Learning for Scientific Discovery

Steve Brunton,*

University of Washington, Seattle WA, USA

Preview Sample

These notes provide a brief overview of how machine learning is being used to advance scientific discovery. Specifically, we focus on how to use machine learning to build data-driven models in fluid mechanics. The process of machine learning is broken down into five stages: (1) formulating a problem to model, (2) collecting and curating training data to inform the model, (3) choosing an architecture with which to represent the model, (4) designing a loss function to assess the performance of the model, and (5) selecting and implementing an optimization algorithm to train the model. At each stage, we discuss how prior physical knowledge may be embedded into the process, with specific examples from the field of fluid mechanics. Finally, we give an example of the sparse identification of nonlinear dynamics (SINDy) approach to model discovery.

1.1 Introduction

The field of fluid mechanics is rich with data and rife with problems, which makes it a perfect playground for machine learning. Machine learning is the art of building models from data using optimization and regression algorithms. Many of the challenges in fluid mechanics may be posed as optimization problems, such as designing a wing to maximize lift while minimizing drag at cruise velocities, estimating a flow field from limited measurements, controlling turbulence for mixing enhancement in a chemical plant, or reducing drag behind a vehicle, among myriad others. These optimization tasks fit well with machine learning algorithms, which are designed to handle nonlinear and high-dimensional problems. In fact, machine learning and fluid mechanics both tend to rely on

* sbrunton@uw.edu.

the same assumption that there are patterns that can be exploited, even in high-dimensional systems (Taira et al., 2017). Often, the machine learning algorithm will model some aspect of the fluid, such as the lift profile given a particular airfoil geometry, providing a *surrogate* that may be optimized over. Machine learning may also be used to directly solve the fluid optimization task, such as designing a machine learning model to manipulate the behavior of the fluid for some engineering objective with active control (Rabault et al., 2019; Ren et al., 2020; Zhou et al., 2020).

In either case, it is important to realize that machine learning is *not* an automatic or turn-key procedure for extracting models from data. Instead, it requires expert human guidance at every stage of the process, from deciding on the problem to collecting and curating data that might inform the model, to selecting the machine learning architecture best capable of representing or modeling the data, to designing custom loss functions to quantify performance and guide the optimization, to implementing specific optimization algorithms to train the machine learning model to minimize the loss function over the data. A better name for machine learning might be "expert humans teaching machines how to learn a model to fit some data," although this is not as catchy. Particularly skilled (or lucky!) experts may design a learner or a learning framework capable of learning a variety of tasks, generalizing beyond the training data, and mimicking other aspects of intelligence. However, such artificial intelligence is rare, even more so than human intelligence. The majority of machine learning models are just that, models, which should fit directly into the decades old practice of model-based design, optimization, and control (Brunton et al., 2020).

With its unprecedented success on many challenging problems in computer vision and natural language processing, machine learning is rapidly entering the physical sciences, and fluid mechanics is no exception. The simultaneous promise and overpromise of machine learning are causing many researchers to have a healthy mixture of optimism and skepticism. In both cases, there is a strong desire to understand machine learning's uses and limitations and best practices for incorporating it into existing research and development workflows. It is also important to realize that while it is now relatively simple to train a machine learning model for a well-defined task, creating a new model that outperforms traditional numerical algorithms and physics-based models is still difficult. Incorporating partially known physics into the machine learning pipeline improves model generalisation and interpretability, which are key elements of modern machine learning (Du et al., 2019; Molnar, 2020).

Preview Sample

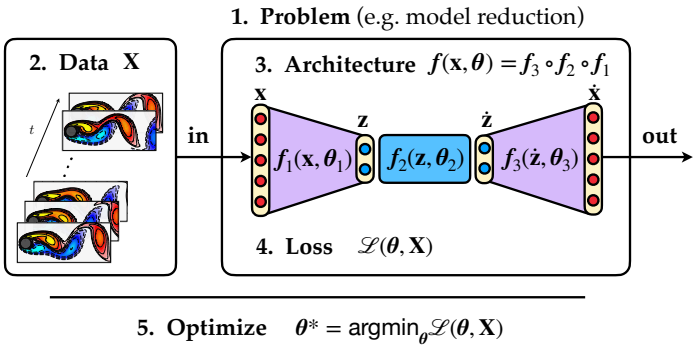


Figure 1.1: Schematic of the five stages of machine learning on an example of reduced-order modeling. In this case, the goal is to learn a low dimensional coordinate system $\mathbf{z} = \boldsymbol{f}_1(\mathbf{x},\boldsymbol{\theta}_1)$ from data in a high-dimensional representation $\mathbf{x}$, along with a dynamical system model $\dot{\mathbf{z}} = \boldsymbol{f}_2(\mathbf{z},\boldsymbol{\theta}_2)$ for how the state $\mathbf{z}$ evolves in time. Finally, this latent state derivative $\dot{\mathbf{z}}$ must be able to approximate the high dimensional derivative $\dot{\mathbf{x}}$ through the decoder $\dot{\mathbf{x}} \approx \boldsymbol{f}_3(\dot{\mathbf{z}},\boldsymbol{\theta}_3)$. The loss function $\mathscr{L}(\boldsymbol{\theta},\mathbf{X})$ defines how well the model performs, averaged over the data $\mathbf{X}$. Finally, the parameters $\boldsymbol{\theta} = \{\boldsymbol{\theta}_1,\boldsymbol{\theta}_2,\boldsymbol{\theta}_3\}$ are found through optimization.

1.2 Physics Informed Machine Learning for Fluid Mechanics

Applied machine learning may be separated into a few canonical steps, each of which provides an opportunity to embed prior physical knowledge: (1) choosing the problem to model or the question to answer; (2) choosing and curating the data used to train the model; (3) deciding on a machine learning architecture to best represent or model this data; (4) designing loss functions to quantify performance and to guide the learning process; and (5) implementing an optimization algorithm to train the model to minimize the loss function over the training data. See Fig. 1.1 for a schematic of this process on the example of reduced-order modeling. This organization of steps is only approximate, and there are considerable overlaps and tight interconnections between each stage. For example, choosing the problem to model and choosing the data to inform this model are two closely related decisions. Similarly, designing a custom loss function and implementing an optimization algorithm to minimize this loss function are tightly coupled. In most modern machine learning workflows, it is common to iteratively revisit earlier stages based on the outcome at later stages so that the machine learning researcher is constantly asking new questions and

Preview Sample

revising the data, the architecture, the loss functions, and the optimization algorithm to improve performance. Here, we discuss these canonical stages of machine learning, investigate how to incorporate physics, and review examples in the field of fluid mechanics. This discussion is largely meant to be a high-level overview, and many more details can be found in recent reviews (Duraisamy et al., 2019; Brenner et al., 2019; Brunton et al., 2020; Brenner & Koumoutsakos, 2021).

1.2.1 The problem

Data science is the art of asking and answering questions with data. The sub-field of machine learning is concerned with leveraging historical data to build models that may be deployed to automatically answer these questions, ideally in real-time, given new data. It is critical to select the right system to model, motivated by a problem that is both important and tractable. Choosing a problem involves deciding on input data that will be readily available in the future and output data that will represent the desired output, or prediction, of the model. The output data should be determinable from the inputs, and the functional relationship between these is precisely what the machine learning model will be trained to capture.

The nature of the problem, specifically what outputs will be modeled given what inputs, determines the large classes of machine learning algorithms: *supervised*, *unsupervised*, and *reinforcement learning*. In supervised learning, the training data will have expert labels that should be predicted or modeled with the machine learning algorithm. These output labels may be discrete, such as a categorical label of a 'dog' or a 'cat' given an input image, in which case the task is one of *classification*. If the labels are continuous, such as the average value of lift or drag given a specified airfoil geometry, then the task is one of *regression*. In unsupervised learning, there are no expert labels, and structure must be extracted from the input data alone; thus, this is often referred to as *data mining*, and constitutes a particularly challenging field of machine learning. Again, if the structure in the data is assumed to be discrete, then the task is *clustering*. After the clusters are identified and characterized, these groupings may be used as proxy labels to then classify new data. If the structure in the data is assumed to be continuously varying, then the task is typically thought of as an *embedding* or *dimensionality reduction* task. Principal component analysis (PCA) or proper orthogonal decomposition (POD) may be thought of

Preview Sample

as unsupervised learning tasks that seek a continuous embedding of reduced dimension (Brunton & Kutz, 2019). Reinforcement learning is a third, large branch of machine learning research, in which an *agent* learns to make control decisions to interact with an environment for some high-level objective (Sutton & Barto, 1998). Examples include learning how to play games (Mnih et al., 2015; Silver et al., 2017), such as chess and go.

Embedding physics: Deciding on what phenomena to model with machine learning is often inherently related to the underlying physics. Although classical machine learning has been largely applied to "static" tasks, such as image classification and the placement of advertisements, increasingly it is possible to apply these techniques to model physical systems that evolve in time according to some *rules* or *physics*. For example, we may formulate a learning problem to find and represent a conserved quantity, such as a Hamiltonian, purely from data (Kaiser et al., 2018a). Alternatively, the machine learning task may be to model time-series data as a differential equation, with the learning algorithm representing the dynamical system (Schmidt & Lipson, 2009; Schmid, 2010; Brunton et al., 2016b; Pathak et al., 2017; Vlachas et al., 2018). Similarly, the task may involve learning a coordinate transformation where these dynamics become simplified in some *physical* way; i.e., coordinate transformations to linearize or diagonalize/decouple dynamics (Lusch et al., 2018; Wehmeyer & Noé, 2018; Mardt et al., 2018; Takeishi et al., 2017; Li et al., 2017; Yeung et al., 2017; Otto & Rowley, 2019; Champion et al., 2019).

Examples in fluid mechanics: There are many *physical* modeling tasks in fluid mechanics that are benefiting from machine learning (Brenner et al., 2019; Brunton et al., 2020). A large field of study focuses on formulating turbulence closure modeling as a machine learning problem (Duraisamy et al., 2019; Ahmed et al., 2021a), such as learning models for the Reynolds stresses (Ling et al., 2016b; Kutz, 2017) or sub-gridscale turbulence (Maulik et al., 2019b; Novati et al., 2021). Designing useful input features is also an important way that prior physical knowledge is incorporated into turbulence closure modeling (Wang et al., 2017; Zhu et al., 2019a, 2021). Similarly, machine learning has recently been focused on the problem of improving computational fluid dynamics (CFD) solvers (Bar-Sinai et al., 2019; Thaler et al., 2019; Stevens & Colonius, 2020; Kochkov et al., 2021). Other important problems in fluid mechanics that benefit from machine learning include super-resolution (Erichson et al., 2020; Fukami et al., 2019), robust modal decompositions (Taira et al., 2017, 2020; Scherl et al., 2020), network and cluster modeling (Nair & Taira, 2015; Kaiser et al.,

Preview Sample

2014; Fernex et al., 2021), control (Maceda et al., 2021; Zhou et al., 2020) and reinforcement learning (Fan et al., 2020; Verma et al., 2018), and design of experiments in cyberphysical systems (Fan et al., 2019). Aerodynamics is a large related field with significant data-driven advances (Kou & Zhang, 2021). The very nature of these problems embeds the learning process into a larger physics-based framework so that the models are more physically relevant by construction.

1.2.2 The data

Data is the lifeblood of machine learning, and our ability to build effective models relies on what data is available or may be collected. As discussed earlier, choosing data to inform a model is closely related to choosing what to model in the first place, and therefore this stage cannot be strictly separated from the choice of a problem above. The quality and quantity of data directly affect the resulting machine learning model. Many machine learning architectures, such as deep neural networks, are essentially sophisticated interpolation engines, and so having a diversity of training data is essential to these models being useful on unseen data. For example, modern deep convolutional neural networks rose to prominence with their unprecedented classification accuracy (Krizhevsky et al., 2012) on the ImageNet data base (Deng et al., 2009), which contains over 14 million labeled images with over $20,000$ categories, providing a sufficiently large and rich set of examples for training. This pairing of a vast labeled data set with a novel deep learning architecture is widely regarded as the beginning of the modern era of deep learning (Goodfellow et al., 2016).

Embedding physics: The training data provides several opportunities to embed prior physical knowledge. If a system is known to exhibit symmetry, such as translational or rotational invariance, then it is possible to augment and enrich the training data with shifted or rotated examples. More generally, it is often assumed that with an abundance of training data, these physical invariances will automatically be learned by a sufficiently expressive architecture. However, this approach tends to require considerable resources, both to collect and curate the data, as well as to train increasingly large models, making it more appropriate for industrial scale, rather than academic scale research. In contrast, it is also possible to use physical intuition to craft new features from the training data, for example, by applying a coordinate transformation that may simplify the representation or training. Physical data often comes from multiple

Preview Sample

Not available in preview

Acknowledgments

SLB acknowledges many valuable discussions and perspectives gained from collaborators and coauthors J. Nathan Kutz, Jean-Christophe Loiseau, and Bernd Noack.

Preview Sample

Preview Sample

2

Fundamentals of Regression

Miguel A. Mendez,*

von Karman Institute for Fluid Dynamics,
Sint-Genesius-Rode, Belgium

This chapter opens with a review of classic tools for regression, a subset of machine learning that seeks to find relationships between variables. With the advent of scientific machine learning† this field has moved from a purely data-driven (statistical) formalism to a constrained or "physics-informed" formalism, which integrates physical knowledge and methods from traditional computational engineering. In the first part, we introduce the general concepts and the statistical flavor of regression versus other forms of curve fitting. We then move to an overview of traditional methods from machine learning and their classification and ways to link these to traditional computational science. Finally, we close with a note on methods to combine machine learning and numerical methods for physics

Preview Sample

2.1 A note on notation and style

Vectors, Matrices and lists. We use lowercase letters for scalar quantities, i.e. $a \in \mathbb{R}$. Bold lowercase letters are used for vectors, i.e., $\boldsymbol{a} \in \mathbb{R}^{n_a}$. The i-th entry of a vector is denoted with a subscript as $\boldsymbol{x}_i$ or with Python-like notation as $\boldsymbol{x}[i]$. We use square brackets to create vectors from a set of scalars, e.g $\boldsymbol{a} = [a_0, a_1, \ldots a_{n_a-1}] \in \mathbb{R}^{n_a}$. Unless otherwise stated, a vector is a column

* mendez@vki.ac.be.

† The term "scientific machine learning" refers to the integration of machine learning with models, principles, and data arising from the natural sciences and engineering. It does not imply that other forms of machine learning are non-scientific. Instead, it highlights a focus on problems where physical laws (e.g., conservation laws, PDEs, thermodynamics) play a central role. Scientific machine learning typically involves combining data-driven methods with domain knowledge, physics-based modeling, numerical simulation, and uncertainty quantification. The emphasis is on developing algorithms that are constrained by—or informed by—scientific theory, enabling improved prediction, interpretability, and generalization in complex physical systems.

vector. The use of transposition when defining a vector embedded within the text of a paragraph or sentence (inline) is omitted.
We use the upper case bold letters for matrices, e.g., $\boldsymbol{A} \in \mathbb{R}^{n_r \times n_c}$, with n_r the number of rows and n_c the number of columns. The matrix entry at the i-th row and j-th column are identified as $\boldsymbol{A}_{i,j}$ or with the Python-like notation as $\boldsymbol{A}[i, j]$. When the Python notation is used, the indices begin with 0. We occasionally work with lists of quantities. Following a Python notation, we enclose lists within round brackets and use bold letters for a list of vectors or matrices, e.g. $\boldsymbol{\Gamma}_* = (\boldsymbol{x}, \boldsymbol{y})$.

Functions and Calculus. Lowercase letters followed by parentheses indicate functions, regardless of whether these are scalar or vector-valued functions, i.e. $a(\boldsymbol{x}) : \mathbb{R}^{n_i} \to \mathbb{R}^{n_o}$. In a vector value function, $\boldsymbol{y} = f(\boldsymbol{x})$ and the subscript is used to define the mapping to each component, i.e. $\boldsymbol{y}_i = f_i(\boldsymbol{x})$. The partial derivative of a function with respect to the input variables $\boldsymbol{x}_i$ is denoted with the compact notation $\partial_{x_i} f$. The total derivative of a function $f(\boldsymbol{x})$, with $f : \mathbb{R}^{n_I} \to \mathbb{R}^{n_o}$ is denoted as $df/d\boldsymbol{x} \in \mathbb{R}^{n_i \times n_o}$ or with the short-hand notation $d_{\boldsymbol{x}} f \in \mathbb{R}^{n_i \times n_o}$. The partial derivative of vector valued functions are collected produce the *Jacobian* $\mathbf{d}f/\mathbf{dx}$. The entries of the Jacobian are computed as $\mathbf{d}f/\mathbf{dx}_{i,j} = \partial_{\boldsymbol{x}_j} f_i$. In the case of a function $f : \mathbb{R}^{n_i} \to \mathbb{R}$, this becomes a row vector and is called *gradient*. Many authors use the nabla symbol ∇f for the gradient, but we do not make the distinction and treat it as a Jacobian.

Parametric and nonparametric representation. For parametric functions $f : \boldsymbol{x} \in \mathbb{R}^{n_x} \to \mathbb{R}^{n_y}$ that depend on parameters $\boldsymbol{w} \in \mathbb{R}^{n_w}$, we use the notation $\boldsymbol{y} = f(\boldsymbol{x}; \boldsymbol{w})$ or $\boldsymbol{y} = f(\boldsymbol{x}|\boldsymbol{w})$. This distinction is important for differentiating between parametric and non-parametric models. For example, we can write a parabola $y = c_2 x^2 + c_1 x + c_0$ as a parametric function $y = f(x; \boldsymbol{c})$ with $\boldsymbol{c} = [c_1, c_2, c_3]$. This notation implies that we have a script that will require in inputs both x and $\boldsymbol{c}$; this is the essence of parametric modeling. However, we could put the focus on the data used for making those predictions. Assuming that the model parameters $\boldsymbol{c}$ were inferred from a training dataset $\boldsymbol{\Gamma}_* = (\boldsymbol{x}_*, \boldsymbol{y}_*)$, we might also write the parametric function as $y = f(x|\Gamma_*)$. This notation implies that our script will require input x and the training data $\boldsymbol{\Gamma}_*$; this is the essence of non-parametric modelling.

Preview Sample

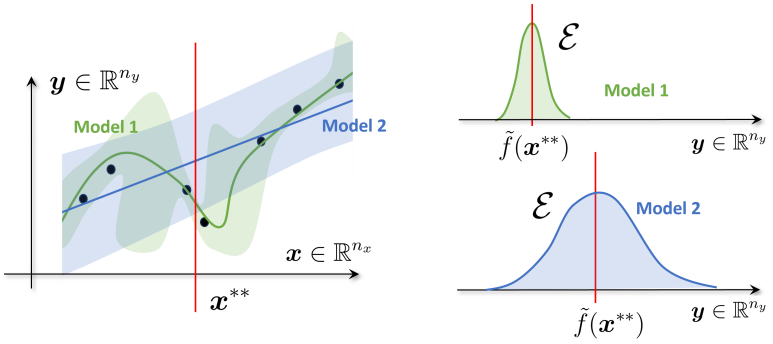


Figure 2.1: General overview of the regression framework, pictorially illustrated for a scalar problem. Left: Two possible models fit the training data (black dots). A prediction is requested and a stochastic process has to be fitted to the data. This can be described as in (2.2) with a deterministic model for the mean and a stochastic model for the local distribution.

Preview Sample

2.2 General Concepts

Regression is a subset of statistics and machine learning and, in particular, a subset of *supervised* (or predictive) *learning*. The goal is to learn a mapping from a continuous variable $\boldsymbol{x} \in \mathbb{R}^{n_x}$ to another continuous variable $\boldsymbol{y} \in \mathbb{R}^{n_y}$. This is in contrast to classification, where the output variable is categorical (i.e. it contains a finite set of classes/categories, such as "yes" or "no", "cat" or "dog", "laminar" or "turbulent"). Regression and classification share much of the general mathematical framework, and most of the algorithms used for one can be used for another with little to no modifications. The boundaries sometimes are so blurred that some regression methods (e.g. logistic regression) are de facto tools for classification.

In its goal of identifying continuous functions from data, regression methods share some common grounds with other curve-fitting methods, such as interpolation or smoothing. However, the key difference is in the statistical roots: the variables $\boldsymbol{x}, \boldsymbol{y}$ have a stochastic nature and thus generate a random process (that is, a distribution of possible functions). Our goal is to fit a random process to the data. In most cases, and indeed for all problems discussed in this lecture, we would be satisfied with a prediction of the mean function and a confidence interval around that mean.

Let us introduce the general formalism with the help of Figure 2.1, which

pictorially represents the problem for the case of a scalar-valued function to ease the graphical representation. Let us assume that a set of n_* training points (black dots) is available. In the most general high-dimensional scenario, these could be stored in two matrices:

$$\boldsymbol{X}_* := \begin{bmatrix} \boldsymbol{x}_{*1} \dots \\ \boldsymbol{x}_{**2} \dots \\ \vdots \\ \boldsymbol{x}_{n_p} \dots \end{bmatrix} \in \mathbb{R}^{n_* \times n_x} \quad \text{and} \quad \boldsymbol{Y}_* := \begin{bmatrix} \boldsymbol{y}_{*1} \dots \\ \boldsymbol{y}_{*2} \dots \\ \vdots \\ \boldsymbol{y}_{n_p} \dots \end{bmatrix} \in \mathbb{R}^{n_* \times n_y} . \tag{2.1}$$

To compress the notation, let us store this training data into a list $\boldsymbol{\Gamma}_* = (\boldsymbol{X}_*, \boldsymbol{Y}_*)$. The simplest model to picture a stochastic process is an additive model:

$$\boldsymbol{Y}(\boldsymbol{X}|\boldsymbol{\Gamma}_*) = \tilde{f}(\boldsymbol{X}|\boldsymbol{\Gamma}_*) + \mathcal{E}(\boldsymbol{X}|\boldsymbol{\Gamma}_*) , \tag{2.2}$$

with $\boldsymbol{X} \in \mathbb{R}^{n_* \times n_x}$ an arbitrary set of input points and $\boldsymbol{Y} \in \mathbb{R}^{n_* \times n_y}$ the associated predictions. The first term $\tilde{f}()$ is a deterministic function that provides one output given one input. The second term $\mathcal{E}()$ is a random variable to which we associate a probability density function $\mathcal{E}$. Therefore, the first term generates a surface in $\mathbb{R}^{n_y}$ (a curve in Figure 1), and the second term generates a distribution at each input, centred on the mean prediction. That is, the stochastic term has zero average everywhere: $\mathbb{E}\{\mathcal{E}(\boldsymbol{X})\} = 0$, with $\mathbb{E}$ the expectation operator. We usually use the deterministic model $\tilde{f}$ to make predictions and the probabilistic model $\mathcal{E}$ to provide uncertainties associated with these predictions.

To make the discussion more concrete, we consider two models, indicated in blue and green in Figure 1. A prediction is required for both of them in a new point $\boldsymbol{x}_{**}$. The figure on the right shows the distributions the two models generate at new locations. The width of the distribution at each location is linked to the width of the shaded areas on the right-hand side. Model 1 is much more complex and has a large variability in the distribution width compared to model 2. Which one of the two is most appropriate? We will never know, but we have the tools to make an educated guess.

2.2.1 A probabilistic perspective: The MLE

The *maximum likelihood estimation* (MLE) is an essential principle for fitting a stochastic process. We here introduce it in the simplest possible setting and

Preview Sample

refer the reader to Bishop et al. (2006); Deisenroth et al. (2020); Abu-Mostafa et al. (2012); Murphy (2012); Watt et al. (2020) for a more extensive treatment.

First, we consider the regression of a univariate (scalar) problem, that is $n_x = n_y = 1$ and $\tilde{f} : x \in \mathbb{R} \to y \in \mathbb{R}$. Second, we consider the simplest assumption for the stochastic contribution: we assume that all distributions we have seen in Figure 2.1 are Gaussian in every x and have all the same standard deviation $\sigma_y \in \mathbb{R}^+$. This assumption leads to the well-known *least square problem*. Different assumptions on the stochastic contributions leads to different minimization problems.

As a model for the mean prediction, we take a generic parametric function $y = \tilde{f}(x; \boldsymbol{w})$. This function could be a polynomial, a radial basis function expansion, an artificial neural network to mention some classic examples. Equipped with a model for the mean prediction and a model for the stochastic contribution in (2.2), we can now calculate the probability of observing a certain outcome $y = y_*$ for a specific input $x = x_*$. In the aforementioned setting, this reads

$$p(y = y_*) \propto \exp\left(-\frac{[y_* - \tilde{f}(x_*; \boldsymbol{w})]^2}{2\sigma_y^2}\right). \tag{2.3}$$

Given the n_* sample points in the training set $\mathit{\Gamma}_* = (\boldsymbol{x}_*, \boldsymbol{y}_*)$, we shall now ask ourselves: *What is the likelihood of observing the collected data if our model is valid?* The fact that this specific set of data has been collected (and not others) suggests that this specific set has a much higher probability of occurring than others*

According to the MLE principle, fitting a model means looking for the model that best explains the collected data, i.e. the model according to which the probability of observing that specific dataset is the highest. In other words, defining the *likelihood* as $p(\boldsymbol{\mathit{\Gamma}}_*|\boldsymbol{Y}(\boldsymbol{X}))$ the probability of observing the data $\boldsymbol{\mathit{\Gamma}}_*$ if the model $\boldsymbol{Y}(\boldsymbol{X})$ is "true", we seek the model that maximizes the likelihood. The assumption of uniform σ_y in each point of the domain implies that the outcome at each location is entirely independent of the outcome in other locations. The same must be true for the data we have collected, which thus are independent

* It is important to remember that in a probabilistic framework, no outcome can be deemed impossible. As illustrated in Figure 2.1, all curves within the shaded area have a high probability of accurately representing the data, indicating their validity. In contrast, curves consistently falling outside this area are less likely to be representative. However, no curve is entirely impossible.

Preview Sample

and identically distributed (i.i.d.). Therefore, the likelihood of observing the specific sequence we have collected, according to our current model, is

$$\begin{aligned} p(\boldsymbol{y} = \boldsymbol{y}_*|\boldsymbol{w}) &\propto \prod_{i=0}^{n_p-1} \exp\left(-\frac{(\boldsymbol{y}_{*i} - \tilde{f}(\boldsymbol{x}_{*i}; \boldsymbol{w}))^2}{2\sigma_y^2}\right) \\ &= \exp\left(-\sum_{i=0}^{n_p} \frac{(\boldsymbol{y}_{*i} - \tilde{f}(\boldsymbol{x}_{*i}; \boldsymbol{w}))^2}{2\sigma_y^2}\right) \\ &= \exp\left(-\frac{||\boldsymbol{y}_* - \tilde{f}(\boldsymbol{x}_*; \boldsymbol{w})||_2^2}{2\sigma_y^2}\right). \end{aligned} \tag{2.4}$$

having used basic properties of the exponential and having introduced the l_2 norm $||\bullet||_2$. Without necessarily involving logarithms and calculus,* it is intuitive that maximizing the exponential (hence the likelihood) requires minimizing its argument. Hence, maximizing the likelihood is equivalent, in this case, to minimizing the mean square error (often referred to as MSE), which reads:

$$\mathcal{J}(\boldsymbol{w}) = \frac{1}{n_*} \sum_{i=0}^{n_p-1} \left[\boldsymbol{y}_{*i} - \tilde{f}(\boldsymbol{x}_{*i}; \boldsymbol{w})\right]^2 = \frac{1}{n_*} ||\boldsymbol{y}_* - \tilde{f}(\boldsymbol{x}_*; \boldsymbol{w})||_2^2 . \tag{2.5}$$

It is also interesting to note that, in this case, the MSE corresponds to† the sample variance of the distribution $\mathcal{E}$. Hence, given $\boldsymbol{w}^* := \arg\min_w \mathcal{J}(\boldsymbol{w})$, one can estimate $\sigma_y^2 \approx \mathcal{J}(\boldsymbol{w}_*)$.

The best method to minimize (2.5) depends on the kind of parametric function. A closed-form solution is available for linear methods (e.g., radial basis function regression), while numerical optimization is required for nonlinear methods (e.g., artificial neural networks).

The generalization to higher dimensions is straightforward. The MSE is the most popular cost function for training machine learning algorithms, but many alternatives exist; we discuss these briefly in Section 2.2.3. First, let us return to the problem of fitting a random process to the data and evaluate its predictions.

* To find the maximum of (2.4), one usually takes the logarithm on both sides to obtain the log-likelihood and proceeds to show that the MLE requires the minimization of the MSE (e.g. Bishop et al. (2006)).

† See Taboga (2021) for a detailed derivation.

Preview Sample

2.2.2 Bootstrapping and cross-validation

Let us assume that the simplified stochastic model of Gaussian with uniform variance σ_y^2 is appropriate. We have identified a suitable parametric model $y = \tilde{f}(x; \boldsymbol{w})$ and the optimal set of parameters $\boldsymbol{w}_*$ that minimize the MSE in (2.5). Approximating $\sigma_y^2 \approx \text{MSE}$, we can now draw the mean prediction $f(x; \boldsymbol{w}_*)$ and identify a constant shaded area around it. For example, for the canonical confidence interval of 95%, we would draw the boundaries of the shaded area as $f(x; \boldsymbol{w}_*) \pm 1.96\sigma_y$. The random process we infer would be:

$$\boldsymbol{Y}(\boldsymbol{X}|\boldsymbol{w}_*) = \tilde{f}(x|\boldsymbol{w}_*) + \mathcal{N}(0, \mathcal{G}(\boldsymbol{w}_*)) \,, \tag{2.6}$$

where $\mathcal{N}(0, \mathcal{G}(\boldsymbol{w}_*)$ is a Gaussian with zero mean and covariance $\mathcal{G}(\boldsymbol{w}_*)$.

Are we done? Not quite. The model in (2.6) is valid only in the theoretical limit of infinitely many training data points. The parameters $\boldsymbol{w}*$ we have identified are optimal (in the sense of minimizing the MSE) only for the specific training dataset $\Gamma = (\boldsymbol{x}, \boldsymbol{y}_*)$. But what guarantees do we have that these parameters will perform well on new, unseen data? In general, we have no such guarantee. We can only hope that, if the training dataset is sufficiently large and representative, the resulting parameter estimate will generalize reasonably well.

A classic method for addressing the problem is statistical resampling. This usually takes the forms of *bootstrapping* or *cross-validation* (see Kim (2009)). Both methods seek to reproduce multiple ensembles of training data from one single dataset, to assess the performance of a predictive model in datasets that were not used during the training. Although the notion of Bootstrapping arises in a more general statistical context (see Efron & Tibshirani (1993); Davidson & Hinkley (2009) for an extensive overview), we here solely focus on the problem of model assessment.

In both cases, the available data is split into *training data* to evaluate the *in-sample error*[*] and *testing data* to evaluate the *out-of-sample error* and assess how well the model generalizes.

Bootstrapping and cross-validation differ in how they make use of the available data. In *bootstrapping*, the dataset is sampled randomly *with replacement*, meaning that data points may be selected multiple times or not at all. This approach is generally appropriate when a large dataset is available. For instance,

[*] which we can use to estimate σ_y, in the simplest stochastic model considered in the previous section.

Preview Sample

if $n_p = 1000$ data points are given, one may train the model $n_E = 100$ times, each time selecting $n_* = 700$ points for training and using the remaining $n_{**} = 300$ for testing. As n_* increases toward n_p, the likelihood that the same data points appear in multiple ensembles increases, which reduces variability across the ensemble. Since sampling is performed with replacement, duplicated entries may also occur within a single training set. Notably, one could even choose $n_* = n_p$ and still obtain slightly different training sets across ensembles.

In *cross-validation*, by contrast, the data is partitioned into K disjoint folds, and *no* data point appears in more than one fold. For $K = 10$, the model is trained $n_E = 10$ times, each time using $n_* = 900$ data points for training and $n_{**} = 100$ for testing: at each iteration, one fold is used for testing and the remaining folds for training. The limiting case is the *leave-one-out* strategy, where $K = n_p$, so the model is trained $n_E = n_p$ times, each time with $n_* = n_p - 1$ and $n_{**} = 1$.

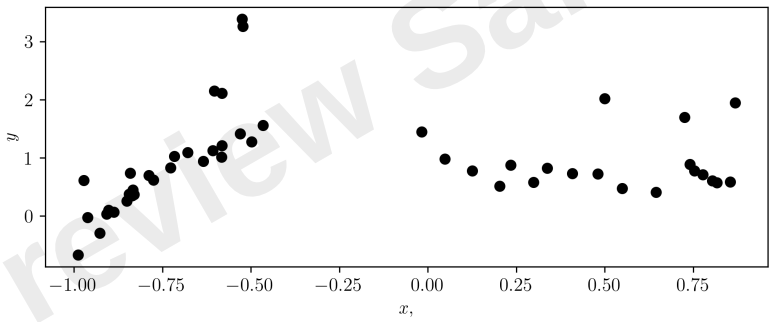


Figure 2.2: Dataset for tutorial 1, to illustrate the usage of cross-validation

Cross-validation is usually preferred for model validation, while bootstrapping is usually preferred for uncertainty quantification using a process called *bagging* (short for Bootstrap Aggregating, proposed by Breiman (1996)). We illustrate their usage with our first Python exercise. We consider the dataset in Figure 2.2. This consists of a small dataset ($n_p = 60$) with a significant noise level and several outliers. Moreover, data is lacking in a critical region. We test two polynomial models of different degrees and evaluate their performances.

The following Python function takes in input the available data (x,y), the order of the polynomial we want to test (n_O), the number of ensemble members we will re-sample (n_E) and the % of data we seek to keep as testing (tp) at each time.

Preview Sample

```
1  def Ensemble_Train_poly(x, y, n_0, n_E=100, tp=0.3):
2   '''
3   see python file for the documentation
4   '''
5   # in sample error of the population
6   J_i = np.zeros(n_E)
7   # out of sample error of the population
8   J_o = np.zeros(n_E)
9   # Distribution of weights
10  w_e = np.zeros((n_0+1, n_E))
11  for j in range(n_E):
12   # Split the dataset into training and testing
13   x_s, x_ss, y_s, y_ss = train_test_split(x, y,
14                                test_size=tp)
15   # Fit the polynomial on the training data
16   w_s = np.polyfit(x_s, y_s, n_0)
17   # store the model parameters
18   w_e[:,j] = w_s
19   #-------- in-sample performance ---------
20   # Make a prediction on the training data
21   y_tilde_s = np.polyval(w_s, x_s)
22   # In-sample error
23   J_i[j] = 1/len(x_s) * np.linalg.norm(y_tilde_s-y_s)**2
24   #-------- out-of-sample performance ---------
25   y_tilde_ss = np.poly(w_s, x_ss)
26   # Out of sample error
27   J_o[j] = 1/len(x_ss) * np.linalg.norm(y_tilde_ss-
28                           y_ss)**2
29  return J_i, J_o, w_e
```

Preview Sample

The code returns the in-sample and out-of-sample MSE (2.5) (J_i, J_o) in each of the resampled ensembles and the population of parameters as a matrix (w_e). The function repeats n_E times the following steps: (1) randomly split the data into training and testing portions (line 10), (2) fits the model minimizing the MSE (line 12), (3) makes predictions on the training data and evaluates in-sample performances (lines 17-19), (4) makes predictions on the testing data and evaluates out-of-sample performances (lines 21-23).

We do not explore the details of MSE minimization on line 12. It is enough to recall that a closed-form solution exists for a polynomial model, requiring only the solution of a linear system—handled by the polyfill function.

Since the process returns n_E models, we could test them all and use the results to make a plot like the one sketched in Figure 2.1 with the following Python function:

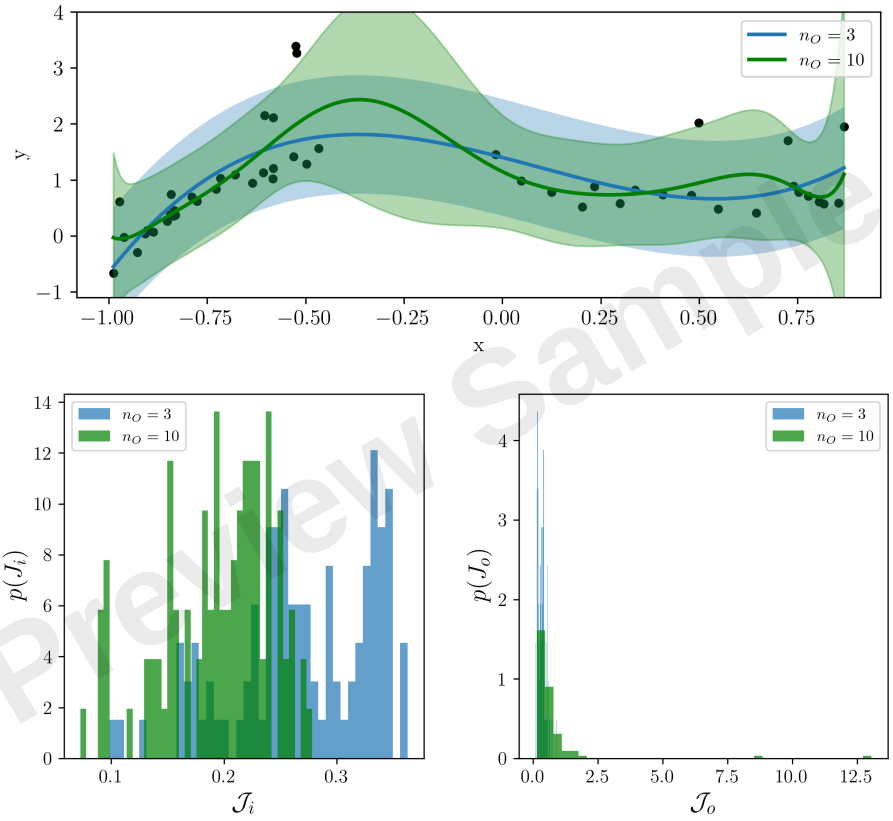


Figure 2.3: Tutorial to illustrate the usage of bootstrapping to estimate uncertainties via bootstrapping. Top: prediction of the two models versus data. Bottom: distribution of in-sample MSE (left) and out-of-sample MSE (right). The more complex model is more prone to overfitting.

Preview Sample

```
1  def Ensemble_Pred_poly(xg, w_e, J_i_mean):
2      '''see python file for the documentation'''
3      # Get n_p, n_O and n_e
4      n_p = len(xg); n_Op1, n_e=np.shape(w_e)
5      # prepare the population of predictions in y:
6      y_pop = np.zeros((n_p, n_e))
7      for j in range(n_e):   # loop over the ensemble
8          # predict for each set of w
9          y_pop[:,j] = Poly_model_Pred(xg, w_e[:,j])
10
11     # The mean prediction will be:
12     y_e = np.mean(y_pop, axis=1)
13     # the ensamble std:
14     Var_Y_model = np.std(y_pop, axis=1)**2
15     # Compute the final uncertainty:
16     Unc_y = np.sqrt(J_i_mean + Var_Y_model)
17     return y_e, Unc_y
```

Preview Sample

The output gives the mean prediction functions and the width of the probability density function sitting on each of these. Note that the final uncertainty in line 16 is the sum of two contributions, assuming that these are independent. The first is the estimated variance σ_y^2 from the data and is computed from the in-sample error. The second is the variance of the predictions due to the sensitivity of the model to small variations in the dataset. The first contribution is large when the model is *underfitting*. This means that the model is too simple to explain the data. The second contribution is large when the model is *overfitting*. This means that the model is too complex for the data at hand and thus becomes too sensitive to minor changes in the data. Increasing the model complexity generally reduces the first contribution but increases the second. The best model is the one that offers the best compromise. The less data we have and the more complex a model is, the higher the risk of overfitting.

Let us compare the performances of two models: the first with n_O=3 and the second with n_O=10. The results are shown in Figure 2.3, and the reader is referred to the provided Python files to reproduce the results.

The predictions of both models seem reasonable, with the model with higher order appearing 'wiggly'. The most complex model has a generally lower in-sample error (`J_i`) and higher out-of-sample error (`J_o`). The distribution of out-sample error is particularly skewed, with some ensemble producing a particularly high MSE. This is a classic footprint of overfitting. It is interesting to note that the width of the uncertainty region is almost constant for the simplest model but oscillates considerably for the most complex, especially in the regions

Not available in preview

"equally" by the training optimizer. Even when an appropriate balance is reached, the optimal solution usually seeks a compromise that does not ensure the fulfillment of the condition to machine precision. This can be problematic for problems extremely sensitive to, e.g., boundary conditions. A remedy is the use of methods for re-scaling* the gradient of the cost function during the training (Wang et al., 2021).

In summary, penalties are usually of great help and always worth considering, given how simple it is to set them up. However, one should not solely rely on these to fully enforce the physics-driven information unless fairly sophisticated methods are used.

3. **Lagrange Multipliers and Hard Constraints.** If the previous approach can be viewed as adding *soft* constraints, this framework enforces *hard* constraints. The training problem is formulated as a constrained optimization, where the data-driven cost function $\mathcal{J}$ is minimized subject to the physics-driven constraint $\mathcal{R}_{\mathcal{D}} = 0$. The literature on constrained optimization is extensive (see Nocedal & Wright (2006) and Martins & Ning (2021) for overviews), and many algorithmic strategies are available.

   The general idea is to introduce the augmented function $\mathcal{A} = \mathcal{J} + \boldsymbol{\lambda}^{\mathsf{T}} \mathcal{R}_{\mathcal{D}}$, where $\boldsymbol{\lambda} \in \mathbb{R}^{n_f}$ is the vector of Lagrange multipliers and n_f is the number of constraints. Unlike the soft-constraint approach, this formulation requires solving for both $\boldsymbol{f}$ and $\boldsymbol{\lambda}$, making the problem numerically more involved.

   For linear methods such as Radial Basis Functions (RBFs), this constrained formulation recovers well-known structures when addressing classical PDEs. When enforcing PDE constraints together with standard boundary conditions (e.g., Dirichlet or Neumann), the resulting system often reduces to a quadratic objective with linear constraints, leading to a large linear system (Sperotto et al., 2022). More broadly, the use of RBF expansions to solve PDEs without a computational mesh dates back to Kansa (1990a,b), and has since generated a substantial literature (see, e.g., Fornberg & Flyer (2015); Šarler (2005); Chen & Tanaka (2002); Chen (2003); Šarler (2007)). RBF-based meshless methods extend classical pseudo-spectral approaches (Fornberg, 1996), where Fourier or Chebyshev expansions are

* A comprehensive and didactic talk concerning this problem is provided by Paris Perdikaris and is available, at the time of writing, at `http://www.ipam.ucla.edu/abstract/?tid=15853&pcode=MLPWS3`.

Preview Sample

typically used, and can be interpreted as a class of collocation schemes. *Arguably*, one of the main reasons these methods have not achieved the same widespread adoption as FEM is that the resulting linear systems tend to be significantly less sparse, and therefore more memory-intensive.

For nonlinear methods, the constraining leads to less explored territory. To the authors' knowledge, no attempt has been made to combine a fully constrained formalism with Genetic Programming for solving PDEs, at least for fluid dynamic applications, and all known approaches in this direction rely on a penalization framework (see Tsoulos & Lagaris (2006); Sobester et al. (2008); Pratama et al. (2023); Oh et al. (2023). Concerning constrained ANNs, this is arguably the most promising and recent avenue. The first approach was recently proposed by Basir & Senocak (2022) (see also Basir & Senocak (2023) and Son et al. (2023)). Much development can be expected soon.

2.4 Summary and Conclusions

This chapter provided a broad overview of regression methods in machine learning and of strategies for incorporating physics-based information into the learning process. We began by framing regression as the task of fitting not just a single curve, but a stochastic process—a distribution of possible functions—to observed data. The simplest viewpoint treated the function as the sum of a deterministic component and a zero-mean stochastic term. We then moved to a probabilistic interpretation, showing how different assumptions on the stochastic term lead to different cost functions, which we referred to as *data-driven* cost functions. We concluded this part by introducing bootstrapping and cross-validation, fundamental tools for assessing generalization performance and understanding the impact of limited data.

We then contrasted the data-driven learning framework with the classical setting of *scientific computing*, where the target function is the solution of a physics-based model, typically expressed as a PDE. Drawing parallels between training parametric models and solving PDEs numerically allowed us to outline methods that combine these two perspectives: seeking functions that both match the available data *and* satisfy the governing physical laws.

With this foundation in place, we are now prepared to move to the next chapter, which presents three tutorial exercises illustrating practical approaches to such hybridization.

Preview Sample

3

Learning with Physical Constraints

Miguel A. Mendez,* Jan van Den Berghe, Manuel Ratz,
Matilde Fiore and Lorenzo Schena

von Karman Institute for Fluid Dynamics,
Sint-Genesius-Rode, Belgium

Preview Sample

This chapter complements the previous by providing three tutorial exercises on physics-constrained regression. These are implemented as "toy problems" that seek to mimic grand challenges in (1) the super-resolution and data assimilation of the velocity field in image velocimetry, (2) data-driven turbulence modeling, and (3) system identification and digital twinning for forecasting and control. The Python codes for all exercises are provided in the course repository.

3.1 Problem 1: "Fill the Gaps" and super-resolution

3.1.1 General Context

In experimental fluid mechanics, velocity fields are often measured using image velocimetry. This includes traditional Particle Image Velocimetry (PIV, Raffel et al. (2018)), which relies on cross-correlation to estimate particle displacement on a regular grid, and Particle Tracking Velocimetry (PTV, Schröder & Schanz (2023)), in which individual particles are localized and tracked to obtain velocity measurements at scattered spatial locations. Both approaches require post-processing to address missing or unreliable data: filtering and outlier removal in PIV can create gaps in the grid, while PTV data are inherently scattered. Historically, PIV gaps were filled using non-parametric local averaging methods (see Section 2.2.5). However, with the increasing adoption of three-dimensional velocimetry techniques (Elsinga et al., 2006; Schanz et al., 2016), the emphasis has shifted toward parametric and physics-informed regression methods for reconstructing complete flow fields.

The key advantage of a parametric model is its analytical representation, which allows for compact storage and facilitates the enforcement of constraints

* mendez@vki.ac.be.

such as exact divergence-free conditions in incompressible flows. The resulting analytic velocity field can be evaluated at *any* spatial location and provides symbolic (rather than numerical) derivatives, a capability often referred to as super-resolution. This removes the need to interpolate the data onto a specific grid or to compute derivatives using finite-difference approximations. The incorporation of physical priors or governing equations in the reconstruction process is commonly referred to as data assimilation, a field that has grown substantially in recent years (see Gesemann et al. (2016); Schneiders & Scarano (2016); Agarwal et al. (2021); Sperotto et al. (2022); Sciacchitano et al. (2022); Jeon et al. (2022)). The present "toy problem" therefore serves as an introductory exercise in data assimilation for image velocimetry.

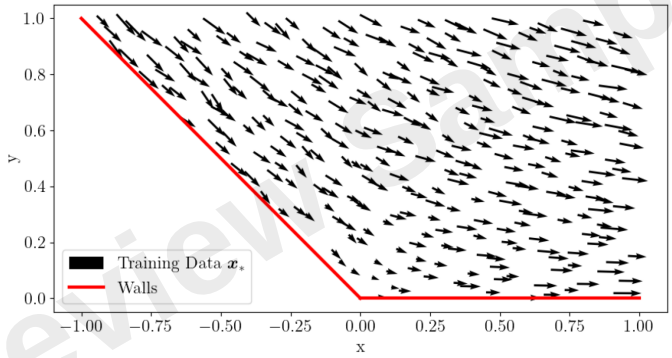


Figure 3.1: Quiver plot of the training data in the constrained RBF regression exercise of Section 3.1.

3.1.2 Proposed Exercise

In this exercise, we make use of the RBF-constrained framework proposed in Sperotto et al. (2022) and implemented in the open-source software SPICY* (meshlesS Pressure from Image veloCimetrY), developed at VKI (Sperotto et al., 2024). The version provided to participants includes recent extensions contributed by Manuel Ratz, who currently leads ongoing developments on meshless data assimilation for turbulence statistics (see Ratz & Mendez (2024)).

In the terminology of Section 2.3, the method belongs to the third category, in

* `https://github.com/mendezVKI/SPICY_VKI`

which *hard constraints* are used to enforce physical priors during regression. The task is to reconstruct a noisy velocity field measured in a two-dimensional corner flow while imposing (1) boundary conditions, (2) divergence-free constraints, and (3) curl-free conditions. In addition to these hard constraints, we also include a global penalty term promoting divergence-free behavior. As we will briefly discuss, enforcing constraints increases memory requirements and numerical difficulty, whereas penalty terms introduce no additional memory cost and can therefore assist the regression.

The flow considered here is shown in Figure 3.1. Training samples are depicted as velocity arrows, and the walls are shown in red. The reference flow is a 2D potential corner flow with potential $Ar^n \cos(n\Theta)$, where $A = 1$ and $n = 4/3$, corresponding to a corner angle of 135°. We sample the resulting velocity field at 298 quasi-random scattered points and add 30% uniform, uncorrelated noise to emulate measurement conditions. While this level of noise is higher than what is typically encountered in practice, it serves to clearly illustrate the robustness of constrained regression.

Preview Sample

3.1.3 Methodology and Results

The constrained RBF approach consists of minimizing the augmented cost function

$$\mathcal{A}(\boldsymbol{w}, \boldsymbol{\lambda}) = \mathcal{J}(\boldsymbol{w}) + \alpha \mathcal{P}(\boldsymbol{w}) + \boldsymbol{\lambda}^T \mathcal{H}(\boldsymbol{w}) \,, \tag{3.1}$$

where $\mathcal{P}$ is a quadratic penalty, $\mathcal{H}$ a linear constraint, and $\boldsymbol{w}$ are the weights for the RBF regression (as in (2.13)). The reason for taking the penalties as quadratic functions and the constraints as linear functions is that by doing so, setting the gradients $d\mathcal{A}/d\boldsymbol{w} = 0$ and $d\mathcal{A}/d\boldsymbol{\lambda} = 0$ leads to a linear system. Moreover, the minimization must treat both $\boldsymbol{w}$ and $\boldsymbol{\lambda}$ as unknown. Let us analyze each of the terms in (3.1) independently.

The first term is the data-driven cost function. We consider a standard l_2 norm at the scope. For the regression of a vector field with training data $\boldsymbol{w}_U^* = \boldsymbol{u}(\boldsymbol{x}^*) = (u(\boldsymbol{x}^*), v(\boldsymbol{x}^*))$ this reads:

$$\mathcal{J}(\boldsymbol{w}_U) = \left|\left| \begin{pmatrix} \boldsymbol{\Phi}(\boldsymbol{x}_*) & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Phi}(\boldsymbol{x}_*) \end{pmatrix} \begin{pmatrix} \boldsymbol{w}_u \\ \boldsymbol{w}_v \end{pmatrix} - \begin{pmatrix} \boldsymbol{u}_* \\ \boldsymbol{v}_* \end{pmatrix} \right|\right|_2^2 = ||\boldsymbol{\Phi}_U(\boldsymbol{x}_*)\boldsymbol{w}_U - \boldsymbol{U})||_2^2 \,, \tag{3.2}$$

where we have shortened the expression through the notation on the right-hand side.

As a quadratic penalty, we consider the divergence-free condition. This requires the spatial derivatives. For the RBF model, these are easily available in any point $\boldsymbol{x}$ by differentiating the basis functions: (2.13):

$$\partial_x u(\boldsymbol{x}) = \sum_{r=0}^{n_b-1} w_r \partial_x \phi_r(\boldsymbol{x}) = \boldsymbol{\Phi}_x(\boldsymbol{x})\boldsymbol{w}_u \tag{3.3}$$

$$\partial_y v(\boldsymbol{x}) = \sum_{r=0}^{n_b-1} w_r \partial_y \phi_r(\boldsymbol{x}) = \boldsymbol{\Phi}_y(\boldsymbol{x})\boldsymbol{w}_u \,. \tag{3.4}$$

Here, the columns of $\boldsymbol{\Phi}_x$ and $\boldsymbol{\Phi}_y$ collect the derivative of each RBF along the x and y direction, respectively. With these analytical derivatives, we can now define a quadratic penalty term as:

$$\mathcal{P} = \left|\left|\begin{pmatrix}\boldsymbol{\Phi}_x(\boldsymbol{x}_*) & \boldsymbol{\Phi}_y(\boldsymbol{x}_*)\end{pmatrix}\begin{pmatrix}\boldsymbol{w}_u \\ \boldsymbol{w}_v\end{pmatrix}\right|\right|_2^2 = ||\boldsymbol{D}_\nabla(\boldsymbol{x}_*)\boldsymbol{w}_U||_2^2 \,. \tag{3.5}$$

The penalty is only applied in the training points $\boldsymbol{x}_*$ but it could also be set in any point.

As linear constraints $\mathcal{H}$ in (3.1), we consider three terms. The first one further enforces the divergence-free condition in some specific points $\boldsymbol{x}_\nabla$; the associated linear constraint is similar to (3.5) and reads:

$$\mathcal{H}_\nabla = \begin{pmatrix}\boldsymbol{\Phi}_x(\boldsymbol{x}_\nabla) & \boldsymbol{\Phi}_y(\boldsymbol{x}_\nabla)\end{pmatrix}\begin{pmatrix}\boldsymbol{w}_u \\ \boldsymbol{w}_v\end{pmatrix} = \boldsymbol{D}_\nabla(\boldsymbol{x}_\nabla)\boldsymbol{w}_U \,. \tag{3.6}$$

The constraint points $\boldsymbol{x}_\nabla$ need not coincide with the data points. The second constraint is on the curl, which we impose on a set of points $\boldsymbol{x}_\omega$ and which reads as follows:

$$\mathcal{H}_\omega = \begin{pmatrix}-\boldsymbol{\Phi}_y(\boldsymbol{x}_c) & \boldsymbol{\Phi}_x(\boldsymbol{x}_c)\end{pmatrix}\begin{pmatrix}\boldsymbol{w}_u \\ \boldsymbol{w}_v\end{pmatrix} = \boldsymbol{D}_\omega(\boldsymbol{x}_\omega)\boldsymbol{w}_U \,. \tag{3.7}$$

Finally, the last constraint is on the boundary conditions. Both Dirichlet and Neuman conditions are possible (see Sperotto et al. (2022)), but for this exercise, we only use the former. The linear constraint for the Dirichlet conditions sets:

Preview Sample

$$\mathcal{H}_D = \begin{pmatrix} \boldsymbol{\Phi}(\boldsymbol{x_D}) & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Phi}(\boldsymbol{x_D}) \end{pmatrix} \begin{pmatrix} \boldsymbol{w}_u \\ \boldsymbol{w}_v \end{pmatrix} - \begin{pmatrix} \boldsymbol{c}_{Du} \\ \boldsymbol{c}_{Dv} \end{pmatrix} = \boldsymbol{D}(\boldsymbol{x}_D)\boldsymbol{w}_U - \boldsymbol{c}_D \,, \tag{3.8}$$

Assembling all terms in the augmented cost function (3.1) leads to:

$$\begin{aligned} \mathcal{A}(\boldsymbol{w}_U, \boldsymbol{\lambda}) =& \mathcal{J}(\boldsymbol{w}_U) + \alpha ||\boldsymbol{D}_\nabla(\boldsymbol{x}_*)\boldsymbol{w}_U|||_2^2 + \boldsymbol{\lambda}_\nabla^T[\boldsymbol{D}_\nabla(\boldsymbol{x}_\nabla)\boldsymbol{w}_U] + \\ & \boldsymbol{\lambda}_C^T[\boldsymbol{D}_C(\boldsymbol{x}_C)\boldsymbol{w}_U] + \boldsymbol{\lambda}_D^T[\boldsymbol{D}(\boldsymbol{x}_D)\boldsymbol{w}_U - \boldsymbol{c}_D] \end{aligned} \tag{3.9}$$

The careful choice of quadratic cost function, quadratic penalty, and linear constraints, together with the linearity of the RBF regression, allows for an analytic solution of the minimization of (3.9). Setting to zero the gradient with respect to both the weights and the Lagrange multipliers leads to the following linear system:

$$\begin{pmatrix} \boldsymbol{A}_U & \boldsymbol{B}_U \\ \boldsymbol{B}_U^T & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{w}_U \\ \boldsymbol{\lambda}_U \end{pmatrix} = \begin{pmatrix} \boldsymbol{b}_{U,1} \\ \boldsymbol{b}_{U,2} \end{pmatrix} \,, \tag{3.10}$$

where the individual terms are defined as:

$$\boldsymbol{A}_U = 2\boldsymbol{\Phi}_U^T(\boldsymbol{x}_*)\boldsymbol{\Phi}_U(\boldsymbol{x}_*) + 2\alpha \boldsymbol{D}_\nabla^T(\boldsymbol{x}_*)\boldsymbol{D}_\nabla(\boldsymbol{x}_*) \tag{3.11a}$$

$$\boldsymbol{B}_U = (\boldsymbol{D}_\nabla^T(\boldsymbol{x}_\nabla),\, \boldsymbol{D}_C^T(\boldsymbol{x}_C),\, \boldsymbol{D}^T(\boldsymbol{x}_D)) \tag{3.11b}$$

$$\boldsymbol{b}_{U,1} = 2\boldsymbol{\Phi}_U^T\boldsymbol{U} \tag{3.11c}$$

$$\boldsymbol{b}_{U,2} = (\mathbf{0},\, \mathbf{0},\, \boldsymbol{c}_D)^T \tag{3.11d}$$

$$\boldsymbol{\lambda}_U = (\boldsymbol{\lambda}_\nabla,\, \boldsymbol{\lambda}_C,\, \boldsymbol{\lambda}_D)\,. \tag{3.11e}$$

This linear system is solved with Cholesky factorizations and Schur complements, as described in Sperotto et al. (2022). In the provided SPICY toolbox, this is carried out in one command line. We use the noisy training data as input to our model and then use the super-resolution property to recompute the solution on a regular grid.

We include the previously mentioned physical priors at different points. The hard constraints of zero curl and zero divergence are *both* applied in the training points $\boldsymbol{x}_*$, and the prediction points $\boldsymbol{x}_{**}$. This serves illustrative purposes for this idealized potential flow test case since the zero curl conditions in real

Preview Sample

Not available in preview

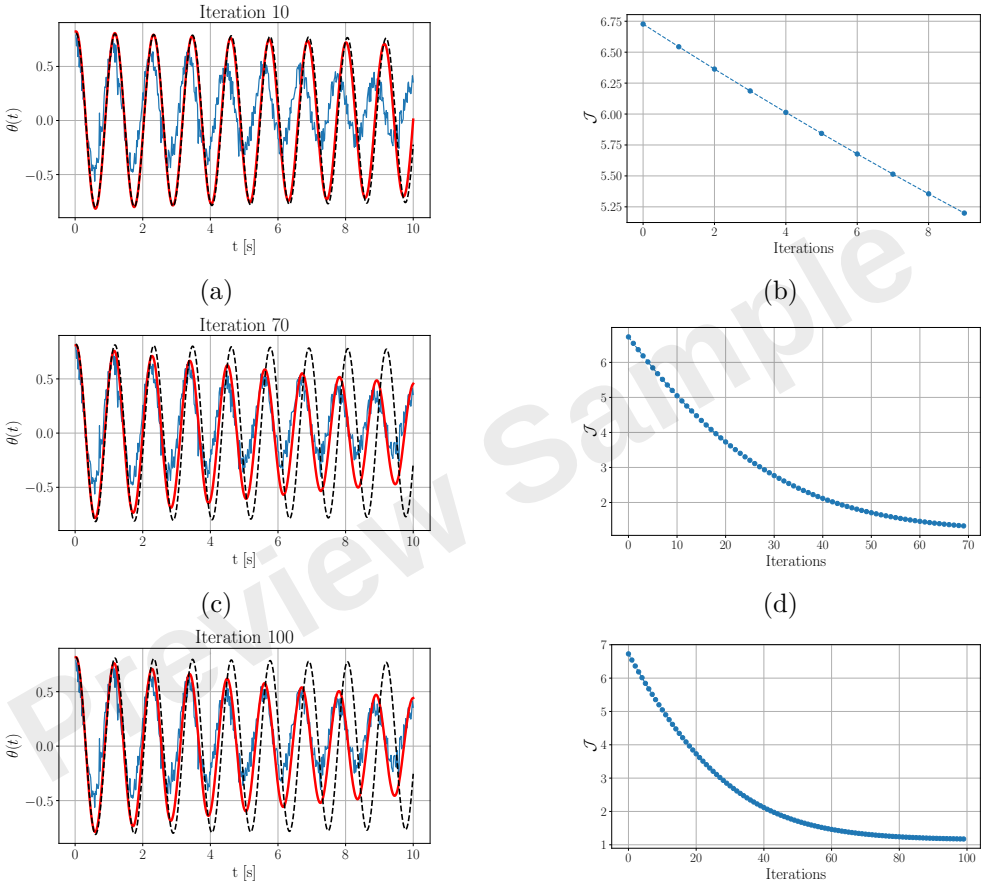


Figure 3.13: Pendulum assimilation performances. The left-hand side shows the time-series of (a) the real system (blue), the initial guess (dashed black line), and the current assimilated prediction (solid red line). The right-hand side depicts the cost function history at the $i-$th iteration.

Preview Sample

Preview Sample

4

Data Driven Turbulence Modeling

Paola Cinnella*

Institut Jean Le Rond D'Alembert, Sorbonne University

This chapter provides an introduction to data-driven techniques for the development and calibration of closure models for the Reynolds-Averaged Navier--Stokes (RANS) equations. RANS models are the workhorse for engineering applications of computational fluid dynamics (CFD) and are expected to play an important role for decades to come. However, RANS model inadequacies for complex, nonequilibrium flows, and uncertainties in modeling assumptions and calibration data are still major obstacles to the predictive capability of RANS simulations. In the following, we briefly recall the origin and limitations of RANS models and then review their shortcomings and uncertainties. Then, we provide an introduction to data-driven approaches to RANS turbulence modeling. The latter can range from simple model parameter inference to sophisticated machine learning techniques. We conclude with some perspectives on current and future research trends.

Preview Sample

4.1 Introduction

Accurate predictions of turbulent flows are of vital importance to natural and engineering systems, including climate, weather forecast, ocean dynamics, astrophysics, aerospace applications, energy conversion systems, civil engineering, and many others. The dynamics of fluid flows is described through conservation for mass, momentum, and energy. While many of the above-mentioned applications may involve compressible flows, reacting flows, or multi-phase flows, hereafter we restrict our attention to incompressible, single-phase, Newtonian fluids with constant properties, described by the celebrated Navier-Stokes (NS) equations,

* paola.connella@sorbonne-universite.fr.

written hereafter in non-dimensional form:

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{4.1a}$$

$$\frac{\partial u_i}{\partial t} + \frac{\partial \left(u_i u_j\right)}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{1}{Re}\frac{\partial^2 u_i}{\partial x_j \partial x_j}, \tag{4.1b}$$

where u_i, p, x_i, and t are, respectively, the non-dimensional flow velocity, pressure, and spatial and temporal coordinates. Although simpler in form than the partial differential equations governing the above-mentioned problems, incompressible NS equations cover a very wide variety of flow configurations and bear the key difficulty that leads to the turbulence modeling dilemma, i.e., the nonlinear convective term in Eq. (4.1b). Equation (4.1) is normalized with respect to a reference length L_{ref}, a reference velocity U_{ref}, and the density ρ and viscosity μ of the fluid. The parameter $Re = \rho U_{\text{ref}} L_{\text{ref}}/\mu$ is the Reynolds number, a measure of the relative importance of inertia to viscous forces. Because of the nonlinearity of the convection terms $\dfrac{\partial \left(u_i u_j\right)}{\partial x_j}$, the NS equations admit chaotic solutions when the Reynolds number is beyond some flow-dependent critical value. As the Reynolds number increases, eventually, the flow reaches a state of motion characterized by strong three-dimensional and unsteady chaotic fluctuations of the velocity and pressure fields, which is referred to as the turbulent regime. The nonlinear and multi-scale character of the governing equations for fluid flows makes the numerical solution of fluid problems particularly challenging, typically requiring an extremely high number of degrees of freedom (DOF). Precisely, the number of DOF required to resolve all spatial scales in turbulent flow grows as $Re^{9/4}$, making so-called Direct Numerical Simulation (DNS) of turbulent flow at all relevant scales impossible for most practical engineering problems, for which the Reynolds number is typically very large (see Pope 2000, for a detailed discussion). This has favored the development of so-called low-fidelity models, obtained by applying a coarse-graining operator to the NS equations. Denoting $\mathcal{N}(w) = 0$ the exact Navier-Stokes operator and $w = (u_1, u_2, u_3, p)$ the exact state vector, the coarse-grained equations are of the form:

$$\mathcal{N}_{\text{CG}}(w_{\text{CG}}) = 0, \quad \text{with} \quad \mathcal{N}_{CG} = \langle \mathcal{N} \rangle$$

$\langle \cdot \rangle$ being the coarse-graining operator, and $w_{\text{CG}} = \langle w \rangle$ the coarse-grained solution. Due to the nonlinearity of the NS equations, $\mathcal{N}_{\text{CG}}(w_{\text{CG}}) \neq \mathcal{N}(w_{\text{CG}})$ so that new, unclosed term, appear in the coarse-grained equations. To fix ideas, $\langle \cdot \rangle$ could be for instance:

Preview Sample

- A filtering operator: $\mathcal{H}_{\mathrm{CG}}$ corresponds to the filtered NS equations, and the corresponding simulations are referred to as Large Eddy Simulations (LES). The coarsening procedure generates unclosed terms, known as subgrid-scale (SGS) stresses, expressing the interactions between the unresolved subgrid scales and the resolved ones, which need to be modelled (Sagaut, 2006).
- The Reynolds averaging operator: $\mathcal{H}_{\mathrm{CG}}$ then corresponds to the Reynolds-Averaged Navier-Stokes (RANS) equations. Such equations are alike the exact equations, except for the appearance of unclosed terms expressing the contribution of unresolved turbulent scales (the main one being the Reynolds stress tensor), which need to be modeled to close the system.
- A combination of the preceding ones: $\mathcal{H}_{\mathrm{CG}}$ then corresponds to a hybrid RANS/LES operator, degenerating to an averaging operator in the limit where no scales are resolved. This strategy has been introduced to palliate the prohibitive computational cost of LES for wall-bounded flows at high Reynolds number due to the small yet energetic scales dominating the dynamics in the near-wall regions (Spalart, 2009). RANS/LES methods then combine LES in free shear regions with RANS models or other simplified models (e.g., boundary layer equation or law of the wall) in the under-resolved near-wall regions (Fröhlich & von Terzi, 2008; Chaouat, 2017). Wall-modeled LES is also used with similar purpose Cabot & Moin (2000); Piomelli & Balaras (2002); Kawai & Larsson (2012); Yang et al. (2015).

The hierarchy of turbulence modeling approaches is illustrated in Figure 4.1, with the top represented by the most physics-resolving and computationally expensive approach (DNS) and the bottom by the most empirical and computationally affordable approach (RANS). Lower-fidelity models toward the bottom of the pyramid involve more flow-dependent and uncertain closures than the higher-fidelity, scale-resolving approaches toward the top of the pyramid. On the other hand, high-fidelity, scale-resolving models are more susceptible to influences from numerical uncertainties and boundary conditions.

For flows dominated by the dynamics of large-scale turbulent structures, LES simulations are the reference high-fidelity approach for low to moderate Reynolds numbers and mildly complex geometries. Unfortunately, they become computationally unfeasible for high-Reynolds-number wall-bounded flows encountered in industrial configurations. Hybrid approaches combining RANS and

Preview Sample

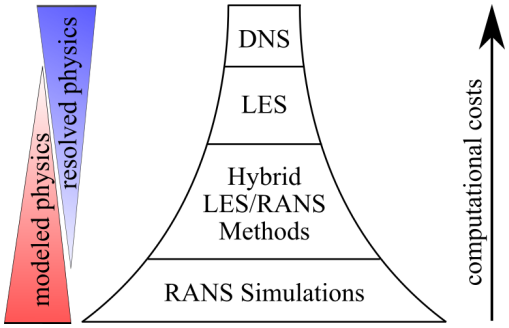


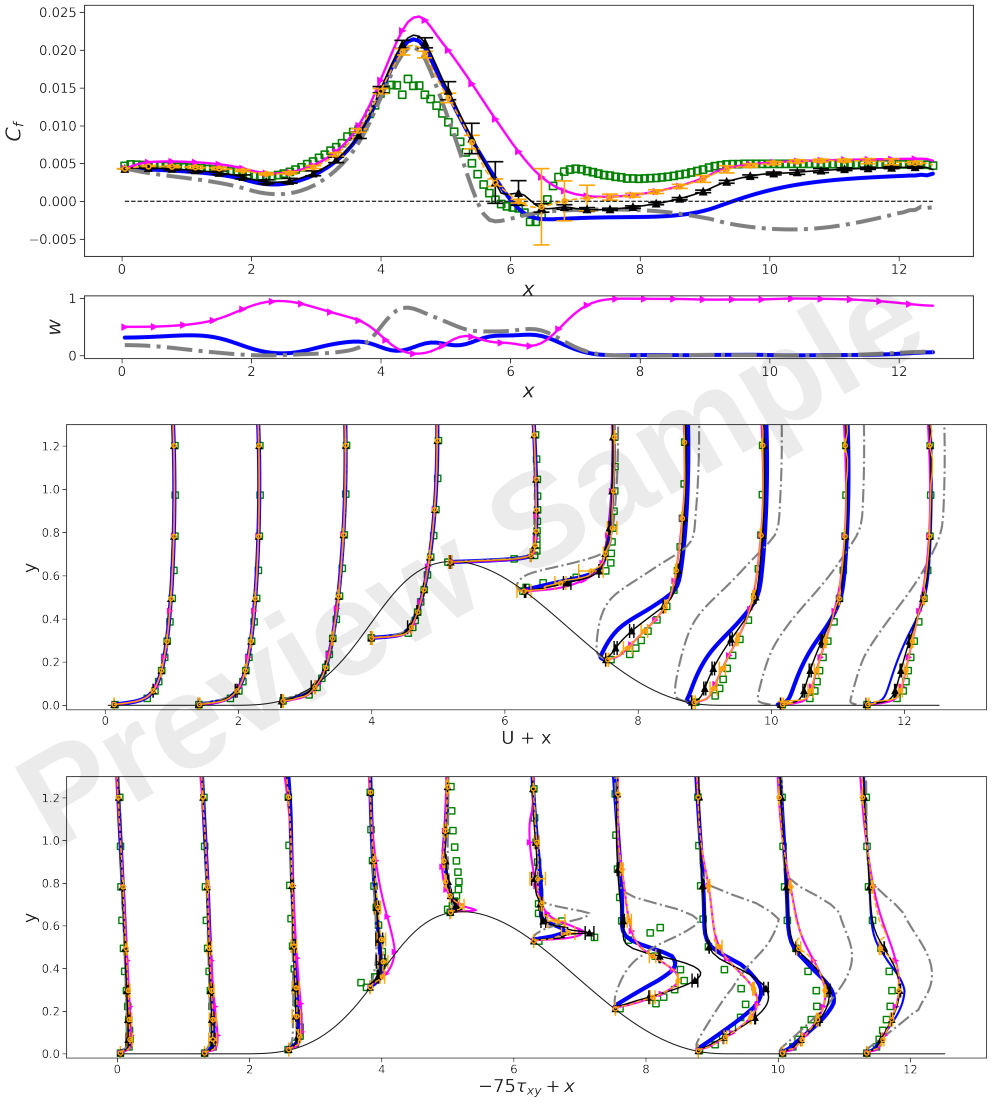
Figure 4.1: A schematic representation of the hierarchy of turbulence modeling approaches based on computational costs and the amounts of resolved versus modeled physics. From Xiao & Cinnella (2019).

Preview Sample

LES have been proposed to alleviate the computational burden, ranging from wall-modeled LES (WMLES) to zonal or global RANS/LES models (Sagaut et al., 2013). The latter are the most attractive for industrial applications due to their ability to switch automatically from RANS to LES based on the local mesh resolution and turbulence characteristics. However, they are sensitive to the numerical setup and computational grid, with the appearance of "grey zones" where the model behaves in between the RANS and LES, possibly leading to aberrant solutions (Spalart, 2009). Efforts to improve RANS/LES models often rely on empirical shielding functions or forcing terms. Also note that RANS/LES are inherently unsteady and 3D, hence much more costly than RANS. Methods for drastically reducing their computational cost are then highly desirable.

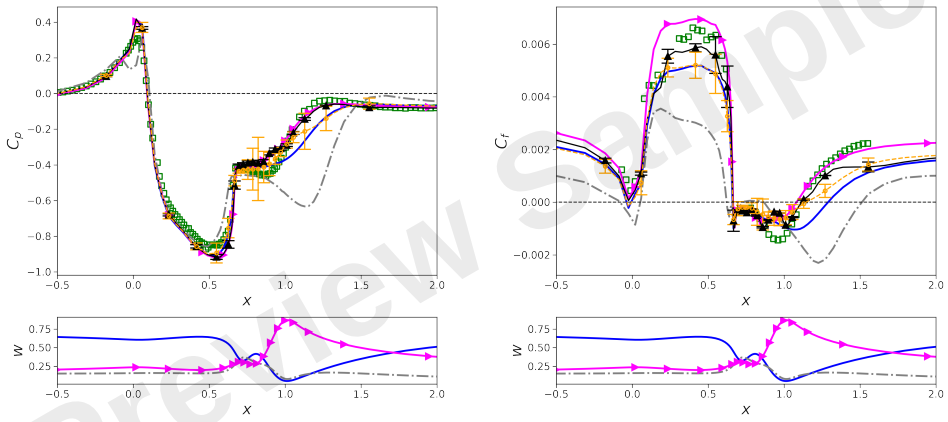
The RANS equations are the workhorse model in CFD, because of their robustness and relatively low computational cost for complex, high-Reynolds-number configurations of practical engineering interest compared to higher-fidelity models. The most widely used RANS models referred to as Linear-Eddy-Viscosity Models (LEVM), rely on the so-called Boussinesq assumption of alignment between the Reynolds stress and the mean strain rate tensor. Such a condition is, however, not verified even for relatively simple flows (see Wilcox (2006); Schmitt (2007)). Besides, the eddy viscosity coefficient is computed via auxiliary relations (e.g., transport equations for turbulent properties), introducing supplementary modeling hypotheses and closure coefficients. More advanced models than LEVM exist, but they also rely on modeling approximations and uncertain closure parameters (see Xiao & Cinnella (2019)). Whatever the closure

Not available in preview



Preview Sample

Figure 4.16: Horizontal velocity U and Reynolds shear stresses τ_{xy} at various x positions for the CD flow case. Baseline $k-\omega$ SST (—); $\boldsymbol{M}^{(ANSJ)}$ $(-\cdot-)$; $\boldsymbol{M}^{(SEP)}$ $(-\triangle-)$; High-fidelity data (□); Non-intrusive X-MA (orange with error bars); Intrusive X-MA (black with error bars).



Preview Sample

Figure 4.17: Pressure coefficient C_p and friction coefficient C_f along x axis for the WMH flow case. Baseline $k-\omega$ SST (—); $\boldsymbol{M}^{(ANSJ)}$ $(-\cdot-)$; $\boldsymbol{M}^{(SEP)}$ $(-\triangle-)$; High-fidelity data (□); Non-intrusive X-MA (orange with error bars); Intrusive X-MA (black with error bars).

5

Prediction of chaotic dynamics from data: An introduction

Luca Magri*[1,2,3], Andrea Nóvoa[1,4], Elise Özalp[1]

[1]Imperial College London
[2]The Alan Turing Institute
[3]Politecnico di Torino
[4]University of Cambridge

Preview Sample

This chapter offers a principled approach to the prediction of chaotic systems from data. First, we introduce some concepts from dynamical systems' theory and chaos theory. Second, we introduce machine learning approaches for time-forecasting chaotic dynamics, such as echo state networks and long short-term memory networks, whilst keeping a dynamical systems' perspective. Third, the lecture contains informal interpretations and pedagogical examples with prototypical chaotic systems (e.g., the Lorenz system), which elucidate the theory. The chapter is complemented by coding tutorials (online) at `https://github.com/MagriLab/Tutorials`.

5.1 Chaotic dynamical systems

In this lecture, we work with deterministic systems. Deterministic systems are noise-free systems, which means that there exists only one solution that corresponds to an initial condition. Chaos is a deterministic phenomenon, which is characterized by erratic behaviour that is difficult—yet possible, in principle—to predict. Chaotic dynamics are characterized by extreme sensitivity to small perturbations, such as changes in the initial conditions, parameters, or external forcing. Two nearby initial conditions, which can differ by a very small amount, will practically diverge in time from each other with an initial exponential rate (Figure 5.1). This makes the time accurate prediction of the solution difficult, which is sometimes informally referred to as *the butterfly effect* (Lorenz, 1969).

* l.magri@imperial.ac.uk.

But not all is lost. The long-term statistics of chaotic systems may be more predictable than the instantaneous time dynamics. The statistics, in fact, may not be significantly affected by tiny perturbations, whereas the instantaneous solution may be. For example, running the same code with the same initial conditions on a different number of processors should, in principle, provide two statistically equivalent solutions,* but with completely different instantaneous fields after a few time steps (Figure 5.1).

In this section, we present some basic concepts and nomenclature, which will be used throughout this lecture. Detailed references in the subject of chaos are Holmes & Guckenheimer (1983); Hilborn (2000); Pikovsky & Politi (2016); Boffetta et al. (2002), among many others.

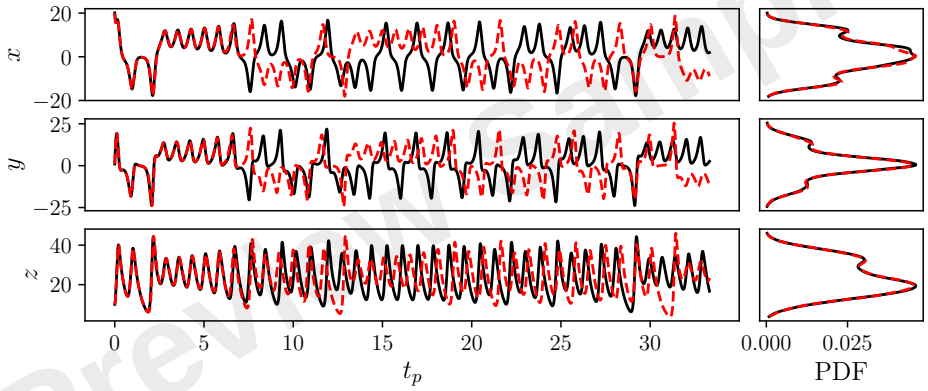


Figure 5.1: Solution of the Lorenz 63 in Lyapunov times (Section 5.1.5) solved for $\mathbf{x}_0 = [20.0, 1.0, 10.0]$ (black line) and for $\mathbf{x}_0 = [20.1, 1.0, 10.0]$ (red dashed line) with a fourth-order Runge-Kutta method.

5.1.1 Dynamical systems' equations

We work with chaotic systems that can be described as autonomous dynamical systems as

$$\dot{\mathbf{x}}(t) = \mathbf{F}(\mathbf{x}(t), \mathbf{p}), \qquad \mathbf{x}(0) = \mathbf{x}_0 \tag{5.1}$$

where the overdot $\dot{(\,)}$ is Newton's notation for time differentiation; $\mathbf{x} \in \mathbb{R}^{N_x}$ is the state vector, where the integer N_x denotes the degrees of freedom; the

* In this lecture, we work with ergodic systems, see Sec. 5.1.1.

subscript 0 denotes the initial condition; $\mathbf{F} : \mathbb{R}^{N_x} \to \mathbb{R}^{N_x}$ is a nonlinear smooth function*; and $\mathbf{p}$ is a vector containing the system's parameters, which will be dropped unless it is necessary for clarity. The solution is given by the trajectory $\mathbf{x}(t)$ that corresponds to the initial condition $\mathbf{x}_0$. Typically, analytical solutions for most problems are not available, which means that we need to resort to discretizing time for integration.

Attractors and ergodicity

We will often talk about "attractors" in this lecture. But what is an attractor? An attractor is the set of values that the solution takes asymptotically. In practice, if you integrate long enough so that the statistics of the solution do not change, you practically obtain (numerically) an approximation of the attractor. Once you are on the attractor, you stay on the attractor (technically, the attractor is an invariant set, which does not change under the dynamical system). This invariant set is a compact set of smaller dimension than the embedding phase space. Chaotic attractors are *strange* because they have a *fractal*, i.e. noninteger, dimension. Trajectories within a strange attractor appear to move around seemingly randomly. In this lecture, we assume that the system evolves on the same attractor for any initial conditions for simplicity. In other words, we work with *ergodic systems* (Birkhoff, 1931), in which the initial condition does not influence the attractor. Therefore, the time average is equal to the ensemble average.

Preview Sample

5.1.2 Linear analysis

We can say, informally, that if you sufficiently zoom in a plot of a nonlinear function, you will find a straight line. Likewise, we can say that if you sufficiently zoom in a nonlinear dynamical system, you will find a linear behaviour. This means that a nonlinear attractor can be "tessellated" or "patched" by infinitesimally small straight planes. Dynamically, we can say that the journey of any state starts from the tangent space (i.e., the small straight patches), which is why it is a key element to characterize.

In the tangent space, the evolution of a state is, of course, given by the dynamical system, $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x})$. However, *in the limit of infinitesimal perturbations*,

* The function $\mathbf{F}$ can represent ordinary differential equations, or partial differential equations that have been spatially discretized with their boundary conditions.

the dynamical system is identical to a linear system $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}) = \mathbf{J}\mathbf{x}$ ($\mathbf{J}$ is the Jacobian), which greatly simplifies our analysis. *The properties of the tangent (linear) space determine many properties of the nonlinear solution.* This is the objective of stability analysis.

In *stability analysis*, we are interested in computing the evolution of infinitesimal perturbations around a reference point of the attractor, $\bar{\mathbf{x}}(t)$. To do so, we split the solution as

$$\mathbf{x}(t) = \bar{\mathbf{x}}(t) + \mathbf{x}'(t), \tag{5.2}$$

where $\bar{\mathbf{x}}(t)$ is the unperturbed solution* of (5.1) such that $||\mathbf{x}'(t)|| \sim O(\epsilon)$, where $\epsilon \to 0$. The perturbation equation is found by truncating the Taylor expansion of the dynamical equations (5.1) to first order, which yields the tangent equation

$$\dot{\mathbf{x}}' = \mathbf{J}(t)\mathbf{x}', \qquad \mathbf{x}'(0) = \mathbf{x}'_0. \tag{5.3}$$

where $\mathbf{J}(t) \equiv \frac{d\mathbf{F}}{d\mathbf{x}}\big|_{\bar{\mathbf{x}}(t)}$ is the Jacobian.† The Jacobian is a key quantity in dynamical systems. On the one hand, the Jacobian around a fixed point is constant (a fixed point could be a steady solution of Navier-Stokes equations). The eigenvalues and eigenvectors of the Jacobian will establish the stability behaviour. If at least one eigenvalue has a positive growth rate, the fixed point is linearly unstable. Around a periodic flow, the time-evolution of the Jacobian matrix is periodic. (A periodic flow could be a periodic solution of Navier-Stokes equations.) If at least one eigenvalue (Floquet exponent) has a positive growth rate, the periodic solution is linearly unstable (e.g., for linear flow analysis, Magri, 2019; Magri et al., 2023). On the other hand, what happens when we perform linear analysis on chaotic solutions? The time-evolution of the Jacobian is chaotic, thus, we need to generalize stability analysis to chaotic Jacobians.

5.1.3 Largest (dominant) Lyapunov exponent

We analyze what makes a solution chaotic. We analyse how a small‡ perturbation evolves. For this, it is convenient to introduce the tangent propagator, which formally maps the perturbation, $\mathbf{x}'$, from time t to time $\tilde{t}$, as

$$\mathbf{x}'(t + \tilde{t}) = \mathbf{M}(t, \tilde{t})\mathbf{x}'(t). \tag{5.4}$$

* Note that we have made no assumption on the unperturbed solution that lies on the attractor, i.e., it may be time-dependent and aperiodic.

† In tensor notation, $J_{ij} \equiv dF_{ij}/dx_j\big|_{\bar{\mathbf{x}}(t)}$, with $i, j = 1, 2, \ldots, N_x$.

‡ We will use the word "small" to informally mean "infinitesimal", unless otherwise specified.

Preview Sample

Equation of tangent propagator

Show that

$$\begin{cases} \dfrac{d\mathbf{M}}{d\tilde{t}} = \mathbf{J}(\tilde{t})\mathbf{M}, \\ \mathbf{M}(t,0) = \mathbf{I}, \end{cases} \tag{5.5}$$

where $\mathbf{I}$ is the identity matrix. Show that

$$\mathbf{M}(t,\tilde{t}) = \mathscr{P}\left\{\exp\left(\int_t^{t+\tilde{t}} \mathbf{J}(\chi)d\chi\right)\right\}. \tag{5.6}$$

where $\mathscr{P}$ is the path-ordering operator.

Setting $t = 0$ without loss of generality, the norm of an infinitesimal perturbation, $\mathbf{x}'_0$, applied to the unperturbed solution, $\bar{\mathbf{x}}_0$, asymptotically grows as (Oseledets, 1968)

$$||\mathbf{x}'(\tilde{t})|| \cong ||\mathbf{x}'_0||e^{\lambda_1(\mathbf{x}'_0,\bar{\mathbf{x}}_0)\tilde{t}}, \tag{5.7}$$

where $\cong$ means "asymptotically equal to" and λ_1 is the largest Lyapunov exponent. This is a result of Oseledets' theorem (Oseledets, 1968), which shows the existence of Lyapunov exponents as constants of the attractor, and, in ergodic systems, they do not depend on the initial condition, $\mathbf{x}'_0$. Therefore, the largest (dominant) Lyapunov is given by

$$\lambda_1 = \lim_{\tilde{t}\to\infty} \frac{1}{\tilde{t}} \log \frac{||\mathbf{M}(0,\tilde{t})\mathbf{x}'_0||}{||\mathbf{x}'_0||}, \tag{5.8}$$

which is the time-averaged growth rate of infinitesimal perturbations. The largest Lyapunov exponent indicates the type of solution (i.e., attractor). If the largest Lyapunov exponent is $\lambda_1 < 0$, perturbations decay and the attractor is a fixed point. If $\lambda_1 = 0$, the attractor is a periodic orbit. If $\lambda_1 > 0$, the perturbation grows exponentially and, typically, the attractor is chaotic.* These criteria can be used to classify bifurcations in fluid systems (Huhn & Magri, 2020b). The largest Lyapunov exponent is a practical measure to compute the predictability of large-scale simulations because it (i) is easy to calculate and (ii) does not depend on the initial conditions in ergodic processes. In

* There are pathological examples of systems that do not fulfill this criterion. In practice, for engineering systems, it is safe to say that at least one positive Lyapunov exponent is associated with chaos, and vice versa.

Preview Sample

large-scale fluid-dynamics simulations, the Lyapunov exponent was calculated in channel and bluff-body flows (Blonigan et al., 2016), homogeneous isotropic turbulence (Nastac et al., 2017; Mohan et al., 2017), reacting and non-reacting turbulent jets (Nastac et al., 2017), a two-dimensional airfoil (Fernandez & Wang, 2017), backward-facing step (Ni & Wang, 2017), partially-premixed flames (Hassanaly & Raman, 2019), to name only a few.

Practical computation of the dominant Lyapunov exponent

We focus on the dominant Lyapunov exponent, λ_1. Obtaining accurate estimates of the Lyapunov exponent is straightforward even in large-scale simulations. A non-intrusive method is based on the calculation of the *separation* trajectory, also known as the *error* trajectory. The separation trajectory is the difference between two nearby trajectories (which can be Eulerian fields in computational fluid dynamics), which originate from two close initial conditions. Because it is almost sure* for the separation trajectory to have a component–even minuscule–in the direction that will grow with the dominant Lyapunov exponent, the separation trajectory will almost surely grow at an exponential divergence rate provided by the dominant Lyapunov exponent. This is why the dominant Lyapunov exponent is of paramount importance in chaotic flows. It can be calculated as described in the following practical and non-intrusive algorithm.

Pseudoalgorithm for dominant Lyapunov exponent

1. **Statistically converged solution.** Run a numerical simulation (5.1) until statistical convergence is reached, say, at t_0. The time solution $\bar{\mathbf{x}}(t)$ thereafter approximates the attractor.

2. **Reset time**, $t = t_0$.

3. **Perturb**. At $t = t_0$, impose the perturbed solution $\mathbf{x}' = \boldsymbol{\epsilon}$ as

$$\mathbf{x}'(t_0) = \bar{\mathbf{x}}(t_0) + \boldsymbol{\epsilon}, \tag{5.9}$$

   where $\boldsymbol{\epsilon}$ is a small random field, whose norm is typically in the range $10^{-9} - 10^{-3}$.

* "Almost sure" means that the Lebesgue measure of the set is 1, i.e., the probability of "being sure" is 1.

Preview Sample

4. **Separation trajectory.** Advance both solutions, $\bar{\mathbf{x}}(t_0)$ and $\mathbf{x}'(t_0)$, to some time t_f and evaluate the separation trajectory

$$\Delta\mathbf{x}(t) = \mathbf{x}'(t) - \bar{\mathbf{x}}(t) \qquad t_0 < t \leq t_f. \tag{5.10}$$

5. **Identification of the linear region** $t_1 < t \leq t_2$ where $\log\left(\|\Delta\mathbf{x}(t)\|\right)$ grows linearly; t_f in item 4 must be larger than t_2.

6. **Lyapunov exponent.** The Lyapunov exponent is the slope of the linear region, which can be obtained by linear regression

$$\lambda_1 \approx \frac{1}{t_2 - t_1} \log\left(\frac{\|\Delta\mathbf{x}(t_2)\|}{\|\Delta\mathbf{x}(t_1)\|}\right). \tag{5.11}$$

Practical tip: The evolution of the error trajectory can be quite erratic (loosely put, it might look "noisy", see Figure 5.2). In this case, you can run an ensemble of computations and take the average to estimate the Lyapunov exponent.

Preview Sample

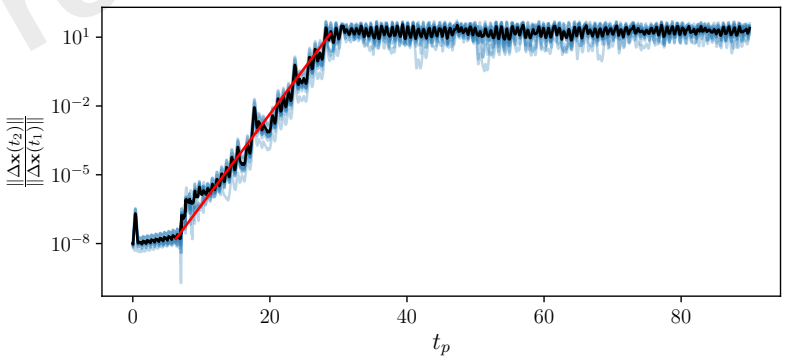


Figure 5.2: Separation trajectory in the Lorenz system over 10 random perturbations (blue lines) and their mean (black line). The dominant Lyapunov exponent is $\lambda \approx 0.929$, which corresponds to the slope (red line). The time unit, t_p is normalized by the dominant Lyapunov exponent.

5.1.4 Lyapunov spectrum

So far, we have defined the largest Lyapunov exponents. What about the "other" Lyapunov exponents in the spectrum? Oseledets theorem shows that, for $\mathbf{x} \in \mathbb{R}^N$ and non-degenerate systems, there exist N Lyapunov exponents, $\lambda_1 \geq \cdots \geq \lambda_N$. The Lyapunov spectrum is provided, theoretically, by the eigenvalues of the Oseledets matrix (Oseledets, 1968)

$$\mathbf{\Xi}^{\pm}(t) = \lim_{t' \to \pm\infty} \frac{1}{2t'} \log \left[\mathbf{M}(t, t')^{\mathrm{T}} \mathbf{M}(t, t') \right]. \tag{5.12}$$

This matrix is called "forward" if $t' \to +\infty$ or "backward" if $t' \to -\infty$. Insight into the Oseledets' matrix can be obtained from Singular Value Decomposition (SVD) of matrix $\mathbf{M}$, such that $\mathbf{M}(t, t') = \mathbf{U}\mathbf{S}\mathbf{V}^{\mathrm{T}}$, where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices and $\mathbf{S}$ is a diagonal matrix with non-negative real entries (the singular values). Substituting the decomposition in the argument of the logarithm of (5.12), an eigenvalue decomposition is obtained, $\mathbf{M}^{\mathrm{T}}\mathbf{M} = \mathbf{V}(\mathbf{S}^{\mathrm{T}}\mathbf{S})\mathbf{V}^{\mathrm{T}} = \mathbf{V}\mathbf{S}^2\mathbf{V}^{\mathrm{T}}$, which, after applying the logarithm, becomes $\mathbf{V}\log(\mathbf{S}^2)\mathbf{V}^{\mathrm{T}} = 2\mathbf{V}\log(\mathbf{S})\mathbf{V}^{\mathrm{T}}$. Thus, (5.12) can be rewritten as

$$\mathbf{\Xi}^{\pm}(t) = \lim_{t' \to \pm\infty} \mathbf{V} \frac{\log\left[\mathbf{S}(t, t')\right]}{t'} \mathbf{V}^{\mathrm{T}}, \tag{5.13}$$

which shows that the Lyapunov exponents, which are the eigenvalues of $\mathbf{\Xi}^{\pm}$, are equal to the average logarithms of the singular values of $\mathbf{M}(t, t')$. However, the numerical computation of the Lyapunov spectrum from (5.13) is unstable: You will almost never get a good prediction of the Lyapunov spectrum (for a discussion refer to Huhn & Magri, 2020b; Huhn, 2022). This is because we need a long-term averaging (in principle, a limit to infinity) to obtain the Lyapunov spectrum. Because of this long integration, vectors almost surely align to the dominant direction corresponding to the largest Lyapunov exponent, λ_1. To overcome this numerical overflow, we employ the Gram-Schmidt orthonormalisation (Schmidt, 1907; Bennetin et al., 1980; Sandri, 1996).

Practical computation of the Lyapunov spectrum

We know how to define and compute the largest Lyapunov exponents, from (5.8) and (5.11). However, we know that there exist N Lyapunov exponents in chaotic systems, under mild assumptions. We need first to define these. Let us imagine a Kafkaesque situation: You are sitting on a chaotic attractor at point, $\bar{\mathbf{x}}_0$. You

Preview Sample

and your friends are infinitesimal. Therefore, you can only walk on straight paths. Now you ask your friends to surround you, and you arrange everybody to fill a perfect parallelepiped centered at $\bar{\mathbf{x}}_0$. Now, the dynamical system, $\mathbf{F}$ (or, equivalently, $\mathbf{J}$ because we are infinitesimal) will take you from $\bar{\mathbf{x}}_0$ to a close location $\bar{\mathbf{x}}_1$. In this trip, some of you will stretch at different rates, some others will not change, and others will be compressed at different rates. Let us start with the lucky ones: those who do not change. These are you and the people close to you in the direction of motion $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x})$. Let us move to the unlucky ones, i.e., those who are stretched or compressed by $\mathbf{F}$ (or, equivalently, $\mathbf{J}$ because we are infinitesimal). Some will get stretched (or compressed) more than others: It depends on which direction they sit around you. So, let us formalize this absurd example with some more rigor.

First, let us consider an infinitesimal p-volume centered *in the tangent space*. A p-volume is a parallelepiped that might have as many sides, $\mathbf{t}_i$, as the phase space, or less, hence, $i = 1, 2, \ldots, p \leq n$

$$\mathrm{Vol}^{(p)}(\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_p) \equiv \mathbf{t}_1 \wedge \mathbf{t}_2 \wedge \ldots \wedge \mathbf{t}_p, \tag{5.14}$$

where $\wedge$ is the wedge symbol (which is the cross-product operation in three-dimensional space). Second, on average, the volume will expand with a rate given by the average divergence, $\lambda^{(p)}$

$$\lambda^{(p)} \equiv \lim_{t\to\infty} \frac{1}{t} \log \left[\frac{\mathrm{Vol}^{(p)}(\mathbf{M}(t)\mathbf{t}_1, \mathbf{M}(t)\mathbf{t}_2, \ldots, \mathbf{M}(t)\mathbf{t}_p)}{\mathrm{Vol}^{(p)}(\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_p)} \right]. \tag{5.15}$$

Because the initial volume is arbitrary, we take it to be unitary. "Expansion" becomes "compression" if the divergence sign is negative. Third, we observe that the vectors $\mathbf{M}\mathbf{t}_i$ are, in general, non-orthogonal. Therefore, they form a parallelepiped. This parallelepiped will numerically collapse along the direction associated with the dominant Lyapunov exponent, which is what we want to avoid when computing the Lyapunov spectrum. However, we know something about geometry: *We know that the volume of a parallelepiped is equal to the volume of the equivalent rectangular parallelepiped.* Therefore, we will compute the volume divergence of the equivalent rectangular parallelepiped, and this will be it. Fourth, how do we make an oblique parallelepiped rectangular? From a linear algebra point of view, this question can be phrased as "Given a non-orthogonal basis, how do we make it orthogonal?". The answer lies in the Gram-Schmidt orthonormalisation. So we rectangularize* the initial volume

* In all likelihood, this is a neologism.

Preview Sample

Not available in preview

Lorenz 63			
λ_i	target	ESN	LSTM
1	0.9050	0.9067	0.873
2	9×10^{-5}	-8×10^{-5}	-8×10^{-3}
3	-14.572	-14.664	-14.0959

Table 5.3: Lyapunov Exponents from LSTM and ESN.

Appendix A: Ridge regression for ESN training

The weights of the output matrix $\mathbf{W}_{\text{out}}$ are obtained by solving

$$\underset{\mathbf{W}_{\text{out}}}{\text{argmin}} \quad \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{W}_{\text{out}}\mathbf{r}(t_i) - \mathbf{x}(t_i)\|_2^2 + \frac{\gamma}{N_x}\sum_{j=1}^{N_x} \|\mathbf{w}_{out,j}\|_2^2 \equiv \mathcal{J}(\mathbf{W}_{\text{out}}) \tag{5.61}$$

$$\text{s.t.} \quad \mathbf{r}(t_i) = \tanh\left(\sigma_{\text{in}}\mathbf{W}_{\text{in}}\mathbf{x}(t_{i-1}) + \rho\mathbf{W}\mathbf{r}(t_{i-1})\right),$$
$$\mathbf{r}(t_0) = \mathbf{0},$$

where γ is the Tikhonov regularisation parameter, σ_{in} is the input scaling factor, and ρ is the spectral radius. The minimization problem (5.61) has an analytical solution, which is obtained by minimizing the cost function $\mathcal{J}$ with respect to the output matrix $\mathbf{W}_{\text{out}}$, and setting the result to zero, such that

$$\frac{\mathrm{d}\mathcal{J}}{\mathrm{d}\mathbf{W}_{\text{out}}} = \frac{1}{NN_x}\sum_{i=1}^{N}\left\{2\left(\mathbf{W}_{\text{out}}\mathbf{r}(t_i) - \mathbf{x}(t_i)\right)\mathbf{r}(t_i)^{\mathrm{T}} + 2\gamma\mathbf{W}_{out}\right\}$$
$$= \frac{1}{NN_x}\sum_{i=1}^{N} 2\left\{\left(\mathbf{W}_{\text{out}}\mathbf{r}(t_i)\mathbf{r}(t_i)^{\mathrm{T}} + \gamma\mathbf{W}_{\text{out}}\right) - \mathbf{x}(t_i)\mathbf{r}(t_i)^{\mathrm{T}}\right\} = \mathbf{0}.$$

Rearranging the terms in (5.62) we find that

$$\sum_{i=1}^{N}\mathbf{W}_{\text{out}}\left(\mathbf{r}(t_i)\mathbf{r}(t_i)^{\mathrm{T}} + \gamma\mathbf{I}\right) = \sum_{i=1}^{N}\mathbf{x}(t_i)\mathbf{r}(t_i)^{\mathrm{T}} \tag{5.62}$$

$$\Rightarrow \quad \sum_{i=1}^{N}\left(\mathbf{r}(t_i)^{\mathrm{T}}\mathbf{r}(t_i) + \gamma\mathbf{I}\right)\mathbf{W}_{\text{out}}^{\mathrm{T}} = \sum_{i=1}^{N}\mathbf{r}(t_i)\mathbf{x}(t_i)^{\mathrm{T}}, \tag{5.63}$$

which can be written in the compact form

$$\left(\mathbf{R}\mathbf{R}^{\mathrm{T}} + \gamma\mathbf{I}\right)\mathbf{W}_{\text{out}}^{\mathrm{T}} = \mathbf{R}\mathbf{X}^{\mathrm{T}}, \tag{5.64}$$

Preview Sample

where $\mathbf{R} = [\mathbf{r}(t_1) \,|\, \ldots \,|\, \mathbf{r}(t_N)]$ and $\mathbf{X} = [\mathbf{x}(t_1) \,|\, \ldots \,|\, \mathbf{x}(t_N)]$ are the horizontal time-concatenation of the output augmented reservoir state and training data. The hyperparameters σ_{in}, ρ and γ can be optimized during training through Recycle Validation (Racca & Magri, 2021).

Preview Sample

6

Reinforcement Learning for Fluid Mechanics: an overview on Fundamentals from a Control Perspective

Onofrio Semeraro*

Laboratoire Interdisciplinaire des Science du Numérique (LISN), CNRS, Université Paris-Saclay, Orsay, France

Preview Sample

This chapter introduces some theoretical foundations of reinforcement learning and its applications to fluid mechanics from a control theory perspective. This choice is intended to provide the reader accustomed to flow control the key ideas in a more familiar vocabulary and a perspective on a vast literature blooming with tremendous momentum. First, we set the stage by introducing flow control and re-framing motivations and goals using reinforcement learning. Next, we shift our focus on some basic concepts by discussing the Hamilton-Jacobi-Bellman equation, the dynamic programming for nonlinear optimal control, and the iterative schemes used for approximate solution. Finally, the chapter closes by reconciling these elements with the terminology of reinforcement learning practice.

6.1 Flow control and reinforcement learning

Control applications in fluid mechanics have attracted the attention of numerous research efforts as it is nowadays recognized that the optimization of aerodynamic flows in aircraft and vehicles design may have a deep impact on the reduction of pollutant emissions, mitigation of acoustic noise or control of highly complex conditions such as separation and stall (Abergel & Temam, 1990; Gad-el Hak, 2000; Kim & Bewley, 2007; Brunton & Noack, 2015; Duriez et al., 2016; Rowley

* onofrio.semeraro@universite-paris-saclay.fr.

& Dawson, 2017). Several methodologies have been applied to control fluids, ranging from passive to active strategies. Here, we consider active flow control (AFC): the dynamics is modified by injecting energy into the system using actuators, acting as transducers for the flow manipulation (Cattafesta III & Sheplak, 2011). The action of the actuators is modulated by policies aimed at optimizing the performance and the dynamic response in a prescribed manner (open-loop) or as a function of some observations of the system at hand (closed-loop). The identification of these policies based on measurements and models of the physics is the objective of the control design.

In principle, AFC strategies optimize the flow in real time; in practice, these techniques are mostly used in limited numerical and experimental test cases. Indeed, approximations based on reduced-order models of the physical system can critically lose accuracy when control is applied, resulting in poor performance and lack of robustness. Addressing these challenges using machine learning (ML) tools (Brunton et al., 2020) and, more specifically, reinforcement learning (RL) can be a key factor in extending flow control to realistic cases by circumventing models limitations or lack of robustness due to their data-driven nature. A possible definition based on the main goals of RL is the following:

> *"RL studies how to use past data to enhance the future manipulation of a dynamical system"* (Recht, 2019).

The description applies equivalently to standard control theory. This suggestive similarity in definition and purpose is not coincidental: the common roots of RL and modern control theory can be found in dynamic programming (DP), a nonlinear optimization protocol based on the Bellman equation (Bellman, 1958). Starting from DP, these two disciplines evolved in parallel in the last decades, leading to the co-development of different approaches to similar problems (Bertsekas, 1995; Sutton & Barto, 2018). The solution of the Bellman equation is the value function, a nonlinear function, which is related to a score associated with a given action or controlled trajectory. Once the value function is known, one can determine the optimal policy. However, the Bellman equation results to be computationally impractical in most cases, even when models are available or direct methods applied. For this reason, a large body of literature is dedicated to the numerical approximation of the Bellman equation and the iterative schemes used for its solution. Iterative methods are particularly interesting as they can be applied with and without a prior model at hand: from a theoretical viewpoint these are the premises on the top of which RL algorithms stand (Sutton &

Preview Sample

Barto, 2018).

RL algorithms are iterative, data-driven and solely relying on limited measurements; models are completely replaced or updated during the iterative process by exploration: the state space of the system is learnt by using past data extracted from the measurements and the interactions of the system or agent with the environment. The set of all the actions the agent can act out in an environment is called action space and a score is assigned to each action for the value function evaluation. In the limit of full knowledge of this space, the resulting policy is optimal if the Bellman equation associated with the value function is fulfilled. In that sense, RL is inspired by nature as it tries to mimic the process of learning of living beings.

A further ingredient is represented by artificial neural networks (ANN). The success of ML applications in very diverse fields, ranging from computer vision and natural language processing to medical diagnosis, is mainly due to the versatility of ANN and their effectiveness in supervised and unsupervised learning (Goodfellow et al., 2016). From a mathematical viewpoint, their versatility is motivated by their properties of universal approximators of nonlinear functions: the combination of ANN, for the approximation of the policy and the value function, with RL led to the Deep RL (DRL) framework. The first application of ANN in RL is often credited to the work by Tesauro (1994), who developed a program – TD-Gammon – combining temporal difference and ANN to play backgammon. In the same years, the application of ANN in combination with dynamic programming was discussed in seminal works on the subject by Bertsekas (1995), under the name of neuro-dynamic programming; the recent developments in the field of deep learning and the super–human performance achieved by DRL in solving games such as `go` and `shogi` (Silver et al., 2017) boosted the popularity of the approach. Together with the vast availability of open-source packages, this is also one of the reasons why DRL is often seen "only" as one of the main subfields of ML and used as a black-box tool. However, this limited perspective risks being rather simplistic: RL is well grounded in optimal control theory, and the interaction between these two disciplines could play a key role in future technological challenges such as the development of driverless cars, self-supervised learning or flow control.

Preview Sample

6.1.1 Standard approaches in flow control: a brief overview

In the following, we introduce a brief, non-exhaustive overview of active flow control. From a physical point of view, the range of applications is as broad as the cases in which the presence of a fluid impacts the efficiency or performance of the dynamical system under investigation; among the examples we can cite, control mechanisms range from quenching the instabilities responsible for the transition to turbulence at relatively low Reynolds numbers (Sipp & Schmid, 2016) to the modification of the mean-flow or of the large scale structures for turbulent cases (Kühnen et al., 2018).

The DP framework provides the theoretical ground for generalizing the optimal control problem from linear to nonlinear cases (Bertsekas, 1995, 2019). A special case is the linear quadratic regulator (LQR), a standard solution of optimal control which can be derived directly from the Bellman equation and reduces to the algebraic Riccati equation in the linear, steady limit (Lewis et al., 2012); when the LQR is combined with optimal estimators, we obtain the linear quadratic Gaussian (LQG). In these hypotheses, LQR/LQG controllers are an ideal benchmark for assessing the optimality of the policies. In flow control, examples can be found in Högberg & Henningson (2002); Högberg et al. (2003); Chevalier et al. (2007). In alternative to the direct methods, one can resort to the adjoint-based formulation in the same linear/linearized limit (Luchini & Bottaro, 2014); the latter can be extended to nonlinear cases and model predictive controllers (MPC) (Glad & Ljung, 2000; Bewley et al., 2001; Xiao & Papadakis, 2019) or adaptive controllers (Åström & Wittenmark, 2008). These techniques have been widely used in fluid mechanics and require for the control design a physical model, describing the behavior of the system.*

When a physical model is available, as in the case of fluid mechanics, solving the governing equations can be too slow with respect to the dynamics at play to be useful, if not even unfeasible: for instance, direct computations of LQR controllers are limited by the degrees of freedom n that cannot exceed $n \approx 10^4$, unless resorting to iterative methods (Semeraro et al., 2013). Alternatively, one can reduce the problem's dimensionality by identifying suitable low-order models that preserve the system's dynamics for control design while meeting computational and real-time constraints. In fluid mechanics, model reduction

* Note: the term *model* will refer only to *physical* models, including low-order ones. In contrast, in the ML community, the term defines approximations or parameterized functions, such as input-output relations mediated by ANN.

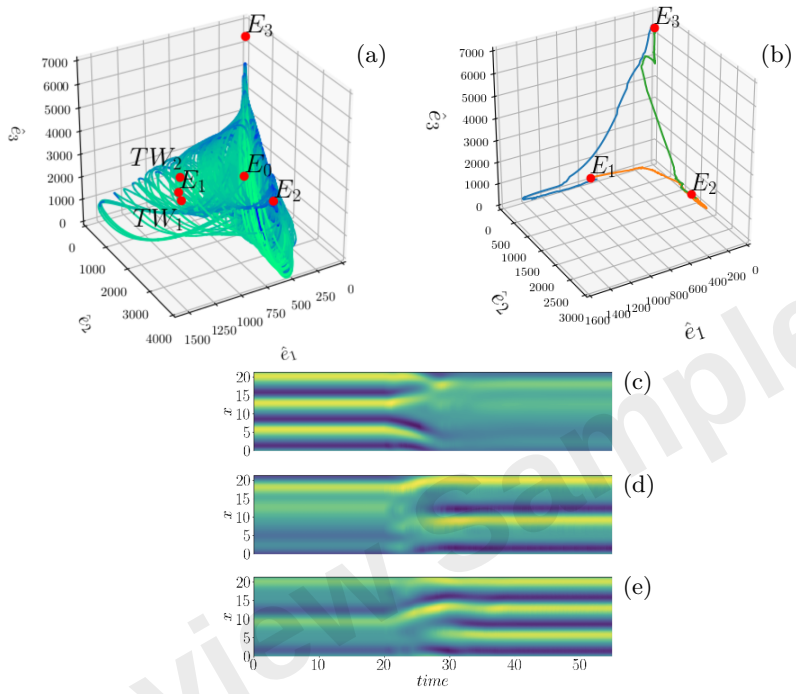


Figure 6.1: Example of a nonlinear dynamical system controlled using localized actuators, localized sensors, and an actor-critic algorithm (see Sec. 6.3); the figure is adapted from Bucci et al. (2019). In (a), the uncontrolled dynamics governed by the Kuramoto-Sivashinsky equation is shown in phase-space, by projecting on the first three Fourier modes ($\hat{e}_i$). Red dots indicate the 4 unstable equilibria (E) and 2 travelling waves (TW) characterizing the dynamics when the domain length $L = 22$ (Cvitanović et al., 2010). Chaotic behavior is observed. In (b), the system controlled by RL is shown: three policies are computed, driving the system towards each nontrivial equilibrium. The controlled trajectories are shown in the spatio-temporal plots (c)-(d)-(e) for $E_3 \to E_1$, $E_1 \to E_2$, and $E_2 \to E_3$, respectively.

and system identification enjoyed widespread popularity in the last two decades, with applications ranging from balance truncation (Rowley, 2005; Ma et al., 2011) to system identification (Ljung, 1999; Noack et al., 2011; Hervé et al.,

Preview Sample

2012) or subspace iteration methods (Van Overschee & De Moor, 2012; Juillet et al., 2014). Review works can be found in Kim & Bewley (2007); Bagheri et al. (2009b); Brunton & Noack (2015); Sipp & Schmid (2016); Rowley & Dawson (2017). Despite this large amount of applications, low-order models are prone to decay of performance and lack of robustness with respect to the modeling uncertainties or change of flow parameters; also, estimation and low-observability deteriorate the performance of model-based controllers when more realistic, experimental setups characterized by noise in the measurements or localized/geometrically-constrained sensors are considered. For example, we can consider the delay of laminar–turbulent transition, generally tested within the limit of small amplitudes or in highly controlled wind-tunnel tests (Semeraro et al., 2013; Fabbiane et al., 2015). Higher amplitudes are characterized by nonlinearities limiting model-based AFC, although some improvements can be achieved by using adaptive filters on relatively slow time-scale (Sturzebecher & Nitsche, 2003; Fabbiane et al., 2014), robust control (Bewley et al., 2000; Leclercq et al., 2019) or model-predictive control (Losse et al., 2011; Goldin et al., 2013; Arbabi et al., 2018; Marra et al., 2024).

6.1.2 Reinforcement learning in flow control

Some of the limitations of model-based controllers can be circumvented by data-driven tools, being inherently based on measurements and past data taken from local sensors to identify control policy without relying on *a-priori* physical model. Among the early works on flow control using RL, those by Guéniat et al. (2016); Rabault et al. (2019, 2020); Fan et al. (2020) are worth mentioning. These works are representative of very different approaches within RL and allow us to quickly glance at the large variety of available techniques. In Rabault et al. (2019), an actor-based method is applied for the control of a flow developing past a cylinder using Proximal Policy Optimization (PPO). In PPO, parametric policies are evaluated by registering the system for a long period of time and calculating the cumulative reward; optimization is performed by stochastic gradient-descent algorithms to update the policy. PPO is thus an on-policy, gradient-descent policy method. A prototype of this class of strategies is the `reinforce` algorithm; the resulting solution in these cases does not satisfy Bellman's optimality principle. A dual approach is to approximate the state-value-action function or $\mathcal{Q}$-function (Watkins & Dayan, 1992). In these approaches, usually termed value-based, a critic is introduced such that

Preview Sample

Bellman optimality is guaranteed if the system analyzed is Markovian and completely known from observables. An early application based on $\mathcal{Q}$-learning was presented for fluid control by Guéniat et al. (2016). A disadvantage of the critic algorithms is the discrete representation of the action, in case a continuous system is considered. For this reason, these two philosophies are often found combined, resulting in a number of different solutions. One example is the previously mentioned PPO algorithm, which incorporates a critic that estimates the value function of the policy. The critic is trained using the cumulative rewards computed along the trajectories, and introduces a baseline, such as the value function. The difference between the observed reward and this baseline defines the so-called advantage function. The resulting critic helps refine the policy by providing feedback on its performance. On the other hand, algorithms where the critic is based on $\mathcal{Q}$-learning can be coupled with an actor part, in order to learn a continuous policy. A recent experimental example is provided by the work of Fan et al. (2020). With a similar strategy, in Bucci et al. (2019) an actor-critic RL algorithm, the Deep Deterministic Policy Gradient (DDPG) (Silver et al., 2014; Lillicrap et al., 2015) is used for the control of the nonlinear dynamics governed by the Kuramoto-Sivashinsky equation, without prior knowledge of the system. The work was among the first demonstrations of control of chaotic dynamical systems by RL. An overview of the results is shown in Fig. 6.1: the highly nonlinear dynamics (a) is stabilized (b) around unstable fixed points (c-e) only by relying on limited measurements and localized equispaced actuators. In principle, the combination of actor and critic allows an approximation of the policy and ensures that the function is a solution of the Bellman equation when Markiovanity conditions are met.

Besides optimal control of fluid flows, RL has also attracted attention in a variety of applications ranging from bio-mimetic optimization to numerical analysis. Among the numerous contributions available in the literature, it is worth mentioning the early work by Vergassola et al. (2007) on infotaxis as a navigation strategy without gradients; this landmark work gave the impulse to numerous works inspired by bio-mimetics where RL is used for the control of gliding or perching (Reddy et al., 2018; Novati et al., 2019), optimal swimming (Verma et al., 2018; Borra et al., 2022) and point-to-point navigation (Biferale et al., 2019). Shape optimization has been discussed in the works by Viquerat et al. (2021); Keramati et al. (2022); Dussauge et al. (2023). From the numerical analysis viewpoint, a recent trend in turbulence modeling consists of automating the closure term by multi-agent RL (Novati et al., 2021), including wall models

Preview Sample

applications (Bae & Koumoutsakos, 2022). Adaptive mesh applications through RL have been recently proposed by Yang et al. (2023) and Foucart et al. (2023).

The interested reader can find extensive reviews on data-driven control in fluid mechanics in Brunton & Noack (2015); Brunton et al. (2020). A recent review of RL in fluid mechanics was authored by Vignon et al. (2023). Finally, it is worth to mention the work by Pino et al. (2023), where a comparative analysis of ML methods for active flow control is proposed, focusing in particular on genetic programming and RL, benchmarked against global optimization techniques such as Bayesian and Lipschitz optimizations.

6.1.3 Organization of the chapter

The remainder of the chapter is organized as follows. In Sec. 6.2, we focus on the Hamilton-Jacobi-Bellman (HJB) equations and DP for nonlinear optimal control. The HJB equation is a classical result that extends the linear optimal control to nonlinear cases (Sec. 6.2.3). The discrete counterpart of the HJB equation – the Bellman equation – is at the heart of DP (Sec. 6.2.5) and RL. Iterative solutions to the Bellman equation – namely, the value iteration and policy iteration – are introduced in Sec. 6.2.6. To avoid an abstract discussion of these methods, we introduce the linear limits as an illustration. The chapter closes with a section where the introduced elements are reconciled with the terminology of RL practice in Sec. 6.3 and a short bibliography in Sec. 6.4.

Preview Sample

6.2 From nonlinear control to reinforcement learning

In this section, we state the optimal control problem for linear and nonlinear systems, showing how the Hamilton-Jacobi-Bellman equation can be used for generalizing the results obtained in the linear framework. The discrete counterpart – the Bellman equation – is analyzed in the second step. The solution of the Bellman equation is at the heart of dynamic programming. Direct and basic iterative solutions are quickly described. Finally, reinforcement learning is introduced.

6.2.1 Nomenclature and state space representation

Due to numerous commonalities characterizing the two frameworks, it is interesting to introduce the basic nomenclature of control theory and RL in parallel. We start by considering the simplified block diagram in Fig. 6.2; this sketch is

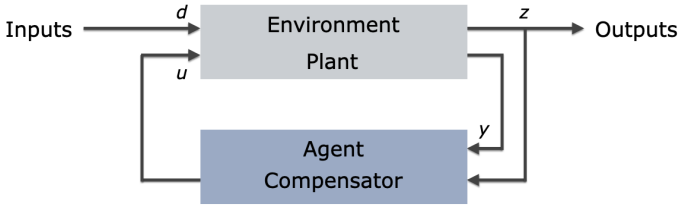


Figure 6.2: Simplified block diagram showing a closed loop. The system to be controlled is defined as plant or environment following control or RL terminology, respectively. The system is equipped with inputs and outputs. The outputs are $y(t)$ and $z(t)$. A feedback law is computed starting from $y(t)$ in order to minimize or maximize the measurements $z(t)$: in optimal control, we define it as a cost function; in RL is mostly found as a reward or value function. A compensator is designed to compute a control input $u(t)$ based on outputs. In RL, the compensator is called agent. Finally, the disturbance input $d(t)$ drives the system's dynamics as input noise or external forcing.

Preview Sample

adapted from a block-diagram in the book by Sutton & Barto (2018), but it is analogous to many other diagrams that it is possible to find in control books. The sketch depicts a closed-loop system where a feedback is introduced for the control of the plant; by definition, the plant is the system to be controlled equipped with a set of actuators and sensors. Sensors detect the system and, in the case of flow control, velocity or pressure probes can be used for this purpose. Actuators introduce energy into the system through volume forcing (such as plasma actuators), blowing and suction jets or active surfaces (Cattafesta III & Sheplak, 2011).

From an analytical viewpoint, a plant corresponds to the open-loop input-output system. We consider here a time-continuous, space-discrete representation and introduce the state vector $\mathbf{x}(t) \in \mathbb{R}^n$, with n the number of spatial degrees of freedom of the system. We thus introduce a nonlinear system denoted by $\mathcal{F} : \mathbb{R}^n \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_d} \to \mathbb{R}^n$; the state-space representation can be written as

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathcal{F}(\mathbf{x}, u, d, t) \\ z(t) &= \mathbf{C}_1\mathbf{x}(t) + \nu_z \\ y(t) &= \mathbf{C}_2\mathbf{x}(t) + \nu_y. \end{aligned} \tag{6.1}$$

The input $u(t) \in \mathbb{R}^{n_u}$ is the control signal and it is the unknown of the control

problem, with n_u entries (multi-input). The noise process or disturbance $d(t) \in \mathbb{R}^{n_d}$ is regarded as an input or forcing on the system. The outputs are $y(t) \in \mathbb{R}^{n_y}$ and $z(t) \in \mathbb{R}^{n_z}$. The sensors are modeled using the operators by $\mathbf{C}_1 \in \mathbb{R}^{n_z \times n}$ and $\mathbf{C}_2 \in \mathbb{R}^{n_y \times n}$, assuming both relations to be linear, and in both cases an additive noise process is added for modeling errors or noise in the measurements.

So far, we have assumed the most general situation where the governing equations and the interactions with control inputs are nonlinear. However, special cases are also possible. A first case consists of a linear description $\mathcal{B}_u(\mathbf{x})$ of the actuation as a function of $\mathbf{x}(t)$, written in the form

$$\dot{\mathbf{x}}(t) \quad = \quad \mathcal{F}(\mathbf{x}, d, t) + \mathcal{B}_u(\mathbf{x})u(t).$$

When the system can be modeled by means of linearization around the equilibrium points of the state space (also denoted as singular or stationary points), or approximated by a linear time-invariant model, the linear mapping is computed as $\mathbf{A} = \nabla_{\mathbf{x}}\mathcal{F}|_{x=\bar{x}}$, where $\bar{\mathbf{x}}$ denotes one of these points and $\mathbf{A} \in \mathbb{R}^{n \times n}$, assuming $u = 0$. Similarly, we obtain a representation for the actuator as $\mathbf{B}_u = \nabla_u \mathcal{F}|_{u=\bar{u}}$ at $\mathbf{x} = \bar{\mathbf{x}}$, where $\mathbf{B}_u \in \mathbb{R}^{n \times n_u}$. The resulting state equation reads as

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}_u u(t) + \mathbf{B}_d d(t), \tag{6.2}$$

where we introduce $\mathbf{B}_d \in \mathbb{R}^{n \times n_d}$ as a model mimicking the external disturbances. The plant can be a simplified model of the environment or coinciding with that. In flow control, it is the case when Linearized Navier-Stokes equations (the plant) are used to control a nonlinear flow (the environment). The main aim of control design and RL is maximizing or minimizing quantities of interest described by a performance index. According to the context, the quantities of interest of the optimization are denoted as cost function when minimized, reward when maximize or value function. Classic quantities of interest in fluid mechanics correspond to measurements chosen for maximizing lift, reducing draft, or mitigating noise emissions. The optimization is achieved by inferring efficient and robust control strategies or policies, modulating the action of the actuators on the system. In control, the compensator provides this action; the compensator consists typically of an estimator, used for reconstructing the state based on the estimation measurements and/or augmenting its expressivity, and the controller indicated by the feedback control law $\mathbf{K} \in \mathbb{R}^{n_u \times n}$, providing the input signal based on the reconstructed state approximating the full state such

Preview Sample

that $\hat{\mathbf{x}}(t) \approx \mathbf{x}(t) \in \mathbb{R}^n$, and $u(t) = \mathbf{K}\hat{\mathbf{x}}(t)$. Note that it is not necessary to consider a full-order estimated state, if the input-output behaviour of the system is well represented by a reduced-order model and its associated state. In this case the reconstructed state can be $n_r \ll n$ and $u(t) = \mathbf{K}_r\hat{\mathbf{x}}_r(t)$, with $\mathbf{K}_r \in \mathbb{R}^{n_u \times n_r}$. A further reduction is obtained when an output feedback control is considered: in this case the control law reduces to the transfer function $u(t) = \mathbf{K}_y y(t)$, with $\mathbf{K}_y \in \mathbb{R}^{n_u \times n_y}$ (Lewis et al., 2012). In RL, the compensator roughly corresponds to the agent, while the policy corresponds to the controller. Indeed, from a technical viewpoint, the policy is a nonlinear function $u(t) = \mathbf{ß}(y(t))$ based on the measurements $y(t)$.

In summary, one can say that the agent is the compensator, while the pairs environment-plant and policy-controller can be used alternatively according to the contexts but defined similarly.

6.2.2 Optimal control in linear system: the Riccati equation

We can now introduce the optimal control problem by considering the time-invariant, continuous linear system

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}_u u, \tag{6.3}$$

already introduced in Sec. 6.2.1. For improved readability, the time dependence of the variables is omitted except where necessary for clarity. We aim at identifying a control law as a function of time $u(t) \in \mathbb{R}^{n_u}$ such that an associated performance index or cost function is minimized. The cost function is chosen to be quadratic and reads

$$\mathcal{J}(\mathbf{x}(t_0), t_0) = \frac{1}{2}\mathbf{x}(T)^T\mathbf{P}_T\mathbf{x}(T) + \frac{1}{2}\int_{t_0}^{T}\left(\mathbf{x}^T\mathbf{Q}\mathbf{x} + u^T\mathbf{R}u\right)dt. \tag{6.4}$$

The terminal cost at time T is defined by the quadratic form $\mathbf{x}^T\mathbf{P}_T\mathbf{x}$, with $\mathbf{P}_T \geq 0$ positive semi-definite. The second term is an integral in time where the integrand is a sum of two terms: an energy associated with the state vector $\mathbf{x}(t)$ and a norm of the control signal $u(t)$, with $\mathbf{Q} \geq 0$ and $\mathbf{R} > 0$ the respective weights determining the relative penalties. To ease the discussion, without loss of generality, we choose a null terminal state, *i.e.* $\mathbf{x}(T) = 0$. We aim at determining the optimal signal $u^*(t)$ defined in the interval $[t_0, T]$. An optimal quantity will be indicated with $(^*)$ in the following whenever needed.

Preview Sample

The solution of the control problem can be obtained by introducing the following Hamiltonian function

$$\mathcal{H} = \frac{1}{2}\left(\mathbf{x}^T\mathbf{Q}\mathbf{x} + u^T\mathbf{R}u\right) + \boldsymbol{\lambda}^T\left(\mathbf{A}\mathbf{x} + \mathbf{B}_u u\right), \tag{6.5}$$

where the first term is associated with the cost function, while the second term is the constraint represented by the dynamical system under consideration, and the state $\boldsymbol{\lambda} \in \mathbb{R}^n$ is the co-state or adjoint state. Minimization is performed by considering the gradients of $\mathcal{H}$ with respect to the parameters of the problem

$$\dot{\mathbf{x}} = \frac{\partial \mathcal{H}}{\partial \boldsymbol{\lambda}} \quad \rightarrow \quad \dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}_u u \qquad \textit{with} \quad \mathbf{x}(0) = \mathbf{x}_0, \tag{6.6}$$

$$-\dot{\boldsymbol{\lambda}} = \frac{\partial \mathcal{H}}{\partial \mathbf{x}} \quad \rightarrow \quad \dot{\boldsymbol{\lambda}} = -\mathbf{A}^T\boldsymbol{\lambda} - \mathbf{Q}\mathbf{x} \quad \textit{with} \quad \boldsymbol{\lambda}(T) = 0, \tag{6.7}$$

$$0 = \frac{\partial \mathcal{H}}{\partial u} \quad \rightarrow \quad 0 = \mathbf{R}u + \mathbf{B}_u^T\boldsymbol{\lambda}. \tag{6.8}$$

By zeroing the last relation, we obtain the optimal control law

$$u^* = -\mathbf{R}^{-1}\mathbf{B}_u^T\boldsymbol{\lambda}.$$

Note that Eq. 6.7 is the equation associated with the adjoint state and is marched backward in time. The iterative solution of the resulting system of equations provides the optimal solution $u(t)$ in Eq. 6.8, which is minimizing since $\partial^2\mathcal{H}/\partial u^2 = \mathbf{R} > 0$.

In alternative, one can cast the equations in Eq. 6.6-6.8 into an algebraic equation: the control algebraic Riccati equation (CARE). To this end, we assume that $\boldsymbol{\lambda}(t) = \mathbf{P}\mathbf{x}(t)$, with $\mathbf{P} \geq 0$; this procedure is called sweep method (Lewis et al., 2012). The last relation can be derived in time and the relations Eq. 6.6-6.7 plugged in

$$\begin{aligned} \dot{\boldsymbol{\lambda}} &= \dot{\mathbf{P}}\mathbf{x} + \mathbf{P}\dot{\mathbf{x}} \\ -\mathbf{A}^T\boldsymbol{\lambda} - \mathbf{Q}\mathbf{x} &= \dot{\mathbf{P}}\mathbf{x} + \mathbf{P}\left(\mathbf{A}\mathbf{x} + \mathbf{B}_u u\right). \end{aligned}$$

The relation can be further manipulated by introducing the optimal solution Eq. 6.8 and the relation $\boldsymbol{\lambda}(t) = \mathbf{P}\mathbf{x}(t)$

$$-\dot{\mathbf{P}}\mathbf{x} = \mathbf{A}^T\mathbf{P}\mathbf{x} + \mathbf{P}\mathbf{A}\mathbf{x} - \mathbf{P}\mathbf{B}_u\mathbf{R}^{-1}\mathbf{B}_u^T\mathbf{P}\mathbf{x} + \mathbf{Q}\mathbf{x}. \tag{6.9}$$

Preview Sample

In the stationary case, $\dot{\mathbf{P}} = 0$; since the solution is valid $\forall \mathbf{x}$, the resulting CARE reads

$$0 = \mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} - \mathbf{P}\mathbf{B}_u\mathbf{R}^{-1}\mathbf{B}_u^T\mathbf{P} + \mathbf{Q}. \tag{6.10}$$

The solution $\mathbf{P}$ enables to compute the control gain $\mathbf{K}$: it is sufficient to consider

$$u^* = -\mathbf{R}^{-1}\mathbf{B}_u^T\boldsymbol{\lambda} = -\mathbf{R}^{-1}\mathbf{B}_u^T\mathbf{P}\mathbf{x}, \tag{6.11}$$

and define the feedback gain as $\mathbf{K} = \mathbf{R}^{-1}\mathbf{B}_u^T\mathbf{P}$. This is the solution to the linear quadratic regulator (LQR) problem.

In perfect analogy, it is possible to define an optimal estimation problem, where we aim at minimizing the following error norm

$$\min \|y(t) - \hat{y}(t)\|, \tag{6.12}$$

where the measurement $\hat{y}(t)$ is associated with the estimated state $\hat{\mathbf{x}}(t)$. The combination of an optimal estimator and the LQR leads to the linear quadratic Gaussian compensator (LQG). For the separation principle, the LQG compensator is optimal (Glad & Ljung, 2000).

Further observations can be made; first of all, since the Riccati equation is not directly solvable for systems characterized by a large number of degrees of freedom, say $N > 10^4$, the direct-adjoint system is often solved instead, see for instance Bewley et al. (2016), Luchini & Bottaro (2014), Semeraro et al. (2013) and citations therein. Additionally, optimization based on a finite sliding temporal window leads to the development of MPC (see e.g. Bewley et al., 2001). Another aspect that can be noted is on the modeling of disturbances. The LQR problem is defined based on the linear time-invariant in Eq. 6.3 and solved using the CARE in Eq. 6.10; the CARE is written using the state matrix $\mathbf{A}$, the model for the actuation $\mathbf{B}_u$, and the metric related to the cost function. In principle, a full-order, perfect model does not require any assumption on the disturbances driving or affecting the system. In practice, this becomes a critical part of the estimation process, particularly in accounting for model uncertainties. Within control theory, robust control is a significant field dedicated to addressing these limitations. Interestingly, Riccati equations can be also cast for robust control synthesis (Glad & Ljung, 2000; Zhou et al., 2002). As previously mentioned, our objective here is to tackle these challenges through data-driven solutions.

6.2.3 The Hamilton-Jacobi-Bellman equation

We can now introduce informally the Hamilton-Jacobi-Bellman (HJB) equation, a nonlinear partial differential equation that is at the heart of the optimal control

Preview Sample

problem in nonlinear systems and provides necessary and sufficient conditions for the optimality. Its solution is the optimal cost function. Once the cost function is known, the optimal control or policy can be computed by maximization or minimization. In this sense, we shift the focus from determining the optimal policy to the computation of the optimal cost function. This dual viewpoint characterizes also the RL applications, where the cost function is typically referred to as value function.

We consider the plant in Eq. 6.1 and assume the state to be fully known and not recorded from the measurements. We define the cost function through the following performance index

$$\mathcal{J}(\mathbf{x}(t_0), t_0) = h(T, \mathbf{x}(T)) + \int_t^T r(\mathbf{x}, u) d\tau, \tag{6.13}$$

where we are interested in determining the optimal control $u^*(t)$ in the interval $t \in [t_0, T]$. As t is the current time, a dummy variable τ is introduced in the integral. The integrand of the second term denoted $r(\mathbf{x}, u)$ is the reward function. The first term is a function associated with the terminal condition at final time T. The cost-to-go $\mathcal{J}(\mathbf{x}, t)$ can be rewritten as

$$\begin{aligned} \mathcal{J}(\mathbf{x}, t) &= \int_t^{t+\Delta t} r(\mathbf{x}, u) d\tau + \int_{t+\Delta t}^T r(\mathbf{x}, u) d\tau + h(T, \mathbf{x}(T)) \\ &= \int_t^{t+\Delta t} r(\mathbf{x}, u) d\tau + \mathcal{J}(\mathbf{x} + \Delta\mathbf{x}, t + \Delta t), \end{aligned} \tag{6.14}$$

where $t + \Delta t$ is a future time close to t. For optimizing the cost-to-go $\mathcal{J}(\mathbf{x}, t)$, the integral has been broken into two terms: the first associated with the reward computed in the interval $[t, t + \Delta t]$, and a second term associated with the future rewards, appropriately replaced by the future cost function. The last expression describes all the possible values for the cost from time t to the time horizon T; however, if we know the optimal cost for all the possible states $\mathbf{x} + \Delta\mathbf{x}$ in the time interval, we can cast the following minimization

$$\mathcal{J}^*(\mathbf{x}, t) = \min_{\substack{u(\tau) \\ t \le \tau \le t+\Delta t}} \left[\int_t^{t+\Delta t} r(\mathbf{x}, u, t) d\tau + \mathcal{J}^*(\mathbf{x} + \Delta\mathbf{x}, t + \Delta t) \right]. \tag{6.15}$$

The expression can be manipulated by performing the integral corresponding to the instantaneous reward and introducing a Taylor series for the $\mathcal{J}^*(\mathbf{x} + \Delta\mathbf{x}, t +$

Preview Sample

Not available in preview

physics or the guidance of a risk metric (Garcıa & Fernández, 2015).

Acknowledgments The author would like to thank Lionel Mathelin (CNRS-LISN) for sharing ideas on numerous data-driven subjects and providing comments on the chapter; Miguel A. Mendez (VKI-ULB) and Alessandro Parente (ULB) for suggestions on the final version of this manuscript; Rémy Hosseinkhan-Boucher, Amine Saibi, and past collaborators Michele Alessandro Bucci (Safran-Tech) and Nicoló Fabbiane (Onera) for their stimulating viewpoints. Part of this research was funded under grants ANR-DGA FlowCon (project-ANR-17-ASTR-0022) and ANR-JCJC REASON (ANR-21-CE46-0008).

Preview Sample

Preview Sample

7

Model Predictive Control

Stefano Discetti,* Luigi Marra, Andrea Meilán-Vila

Universidad Carlos III de Madrid, 28911, Leganés, Spain

This chapter provides an overview of Model Predictive Control (MPC). The concept of receding horizon control is introduced, and the main strengths and limitations of the technique for controlling linear and nonlinear dynamical systems are discussed. The implementation of MPC for linear discrete time-invariant systems is presented as a scenario to familiarize the reader with the main concepts and on-time implementation challenges. The solution via batch approach is included as an interesting exercise for the reader to dive into the optimization process.

The formulation for nonlinear dynamical systems is then presented, including a discussion of the control's feasibility, stability, and hyperparameter tuning. Finally, a test case of the application of data-driven MPC for controlling the wake of a fluidic pinball is presented, in which system identification, hyperparameter tuning, and control optimization are entirely data-driven.

Preview Sample

7.1 What is Model Predictive Control?

The term Model Predictive Control (MPC), also referred to as Receding Horizon Control (RHC), designates a large class of control methods based on the use of a model of the system to predict future behavior and identify optimal control inputs over a recurrently shifting horizon (hence the term *receding horizon*). The flexible framework of MPC fostered research efforts in the last decades, with a recent renaissance thanks to the new opportunities brought by the advances of machine-learning techniques.

The literature on MPC is very vast and varied. Providing a complete overview here would be a daunting task. These notes aim to provide a walkthrough of the main implementation steps, challenges, and caveats without the pretense of being exhaustive. The interested reader can dive into more profound waters

* sdiscett@ing.uc3m.es.

by referring to books, covering a broad spectrum of topics within MPC, from foundational theories to practical applications, and from linear to nonlinear systems (Maciejowski, 2001; Camacho & Alba, 2013; Wang, 2009; Kouvaritakis & Cannon, 2016; Borrelli et al., 2017; Grüne et al., 2017; Rawlings et al., 2017; Rossiter, 2017). Several excellent review papers also cover fundamentals and industrial applications of the method, see e.g. Qin & Badgwell (1997); Lee (2011); Mayne (2014); Hewing et al. (2020); Schwenzer et al. (2021) among others.

This section describes the ingredients of MPC and briefly outlines its main advantages and limitations over other classical control strategies. Finally, it closes with a short historical perspective on the development of MPC and a brief overview of recent successful applications in fluid flows.

7.1.1 The rational behind MPC

The main strategy of MPC is sketched in Figure 7.1. The following steps are generally followed:

- The state of the system is observed at time t_j.
- The system's behavior under different input sequences is predicted with a plant model. The model incorporates previous inputs and states to output the expected dynamics based on a sequence of future control actions. A control and a prediction window/horizon are set. If the control window is shorter than the prediction one, the input is most often left unchanged after the last instant of the control window.
- The input sequence is optimized to achieve a certain control objective. This process typically involves minimizing a cost function containing penalties for the input and the discrepancy with the objective. The optimization process can include constraints, e.g. bounds for the acceptable values of states and inputs, rate of change of inputs/outputs, total energy expenditure, etc. The setpoint, i.e., the target to be achieved for each of the state coordinates, can be time-dependent. Future changes to the setpoint can easily be integrated or even be anticipated to account for time delays in the practical implementation.
- The optimal control sequence is applied only for a short time (in discrete systems, ideally, down to a single step).

Preview Sample

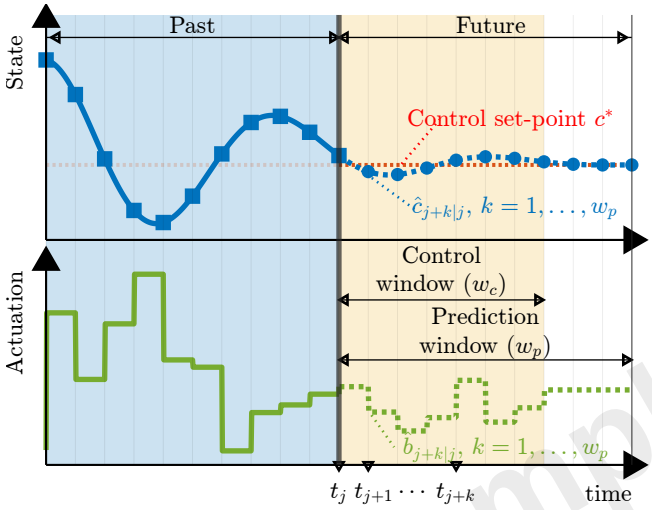


Figure 7.1: Graphical representation of the MPC strategy for stabilizing around a setpoint (horizontal dashed line). Past measurements (light-blue-shaded region) depict system state (blue lines with squares) and actuation (green line). The control window, of length w_c, is shown in orange. Dashed lines indicate future state and actuation predictions. Blue circles represent a discrete sampling of the system state prediction. The continuous formulation allows non-mandatory discrete sampling and allows step-like actuation requirements to be relaxed. Figure reproduced from Marra et al. (2024), licensed under CC BY 4.0.

Preview Sample

- The state is observed after applying the control, and the prediction process is repeated to update the control action.

The open approach of MPC is particularly well-suited for controlling systems with multiple inputs and outputs, allowing for the easy incorporation of hard constraints and flexible implementation of control objectives. The cornerstone of MPC is the plant model, which should be sufficiently accurate and robust to ensure accurate prediction of future states under control actions but, at the same time, simple enough to allow real-time computation.

7.1.2 Main advantages and limitations

MPC is a powerful control strategy that brings forth several desirable features:

- The framework is highly flexible, allowing the application to linear, nonlinear, continuous, and discrete dynamical systems, as long as the plant is modeled with sufficient accuracy.

- The implementation of constraints is straightforward. This is particularly relevant for applications where safety concerns are critical. Furthermore, control feasibility studies can be performed offline if a plant model is available. For instance, technological constraints such as intrinsic time delays, rate of change/update of the actuation, and upper/lower bounds of control actions can easily be tested and stressed.

- It incorporates a model of the system, implemented in a predictive framework, which allows anticipating future events and taking corrective actions in advance. This provides a significant advantage with respect to reactive controllers, which react to current and past states/errors.

- The prediction framework handles multiple inputs and outputs, thus allowing the development of coordinated control strategies.

- Online updates of the model enable adaptive control under changes in operating conditions, parameters of the dynamics, etc. In contrast, classical control strategies often require manual re-tuning.

- The update of the initial state of the prediction implicitly introduces a degree of robustness. Deviations from the expected state are indeed directly observed when updating subsequent control actions, and the control input is updated according to such discrepancy.

On the downside, online implementation is still a grand challenge. Solving the optimization problem is computationally intensive. The computational cost increases with the model complexity. Long prediction horizons can easily lead to large latency times, thus jeopardizing real-time control. Short prediction horizons reduce the computational cost, but on the other hand, might lead to *short-sighted* control policies which might be affected by stability issues and poor performance. Fast-paced advances in hardware, however, are paving the way toward more affordable MPC implementation, thus further expansion of this control strategy is to be expected shortly.

Furthermore, the model is a cornerstone of MPC implementation. Models should be parsimonious for efficient implementation but, at the same time,

Preview Sample

accurate over sufficiently long time horizons. While for some systems, analytical models might be available from applications of first principles and governing equations, in general, plant models must be distilled from data of the system behavior. Reliable system identification is the key to an effective implementation of MPC. Recent advances in data-driven system identification and modeling are progressively eroding this barrier, enlarging the portfolio of possible applications of MPC.

7.1.3 Towards application to chaotic high-dimensional systems and fluid flows

The concept of formulating a control problem as a moving-horizon controller, i.e., a sequence of optimizations of open-loop sequences with a recursive update of the state, roots back to the work of Propoi (1963). The first step towards practical applications was taken only a decade later, with the works of Richalet et al. (1978) and Cutler & Ramaker (1980). Richalet et al. (1978) refers to the method as Model Predictive Heuristic Control (MPHC), stressing the heuristic nature of the control strategy. A few years later, Cutler & Ramaker (1980) developed the Dynamic Matrix Control (DMC), primarily targeted to the oil refining industry. The DMC is based on a piecewise linear model, progressively updated using the discrepancy between the predicted output and the measured states. The work was later expanded by Garcia & Morshedi (1986), with the Quadratic Dynamic Matrix Control (QDMC). The method integrates quadratic programming into DMC, allowing it to handle directly process constraints.

During the 80s, the work on MPC was mostly directed at establishing theoretical grounds for its application in industry. In the following decade, strong efforts were targeted at addressing the robustness and stability of the procedure, with the introduction of Robust Model Predictive Control (Campo & Morari, 1987). For an excellent survey on the topic, the reader is referred to the review by Bemporad & Morari (2007).

Since then, MPC has been widely used in numerous industrial applications. MPC is particularly popular in the process industry, with the petrochemical sector as its main historical promoter. It found also extensive application in power electronics (Vazquez et al., 2014), building climate control (Oldewurtel et al., 2012), agriculture (Ding et al., 2018), robotics (Shi & Zhang, 2021), automotive (Hrovat et al., 2012), among others (see Schwenzer et al., 2021 for a detailed review).

Preview Sample

Applying MPC to turbulent flow control is extremely challenging since the system to be controlled is often strongly nonlinear and high-dimensional and evolves at timescales that are difficult to tackle with current hardware. Nonetheless, the potential to disclose unforeseen optimal control strategies fueled the interest in MPC for flow control. Early efforts have targeted using simulation as an exact plant model (Bewley et al., 2001) to relaminarize a channel flow using unsteady blowing/suction with zero net mass flux. Exploiting receding horizon control strategies undisclosed the interesting result that formulations targeted to drag minimization are less effective than formulations aimed to control the terminal (i.e., at the end of the optimization horizon) turbulent kinetic energy if relaminarization is sought. The results highlight that strategies that allow exploration or deviation from the desired state in the short term can still effectively achieve the final goal at the end of the horizon.

A remarkable evaluation of MPC against other model-based and model-free techniques for the delay of the laminar/turbulent transition has been conducted by Fabbiane et al. (2014). More recently, Morton et al. (2018) proposed a framework called DeepMPC, which combines the receding-horizon approach with a deep neural network for the plant model (in this case to approximate the mapping of the Koopman operator), and applied it to the control of the flow past a circular cylinder. A data-driven Koopman-based MPC framework was also used by Arbabi et al. (2018) to stabilize a lid-driven cavity. The basic idea is to lift the nonlinear dynamics in a higher dimensional space where the dynamics leverage the Koopman operator and delay embeddings.

More recently, Bieker et al. (2020) applied DeepMPC to the wake of a fluidic pinball and extended the capability of the framework to account for state observation from a limited number of sensors. The low-order model is approximated using a Recurrent Neural Network (RNN). To reduce the computational cost associated with online optimization, Sasaki & Tsubakino (2020) proposed a method to design a control law using regression analysis. In this approach, data from offline simulations was used to approximate the behavior of MPC. Additionally, Krishna et al. (2022) investigated the use of MPC to optimize the trajectory planning of an active mobile sensor in an unsteady fluid flow field. In this case, the plant model is imposed as a double-gyre flow field, and the focus is on the effect of actuation and state penalty parameters on the strategy adopted by MPC to exploit unsteady coherent structures. Recently, Déda et al. (2023) compared MPC with a neural-network controller for several systems, including the compressible flow past a cylinder.

Preview Cambridge

Neural network controllers show promising results when sufficient data are available for training; however, in the low-data limit, parsimonious descriptions of the dynamics of the system (see Kaiser et al., 2018b) would provide an advantage to MPC.

7.2 A glimpse of MPC of linear systems

This section focuses on the formulation and implementation of MPC for the control of linear systems. While exploring a simpler scenario can certainly help the reader grasp the fundamental concepts behind MPC implementation, it is also important to note that the history of fluid mechanics is rich with successful applications of linear model-based flow control, such as transition delay (Bewley & Liu, 1998; Bagheri et al., 2009a; Semeraro et al., 2013) or skin friction drag reduction (Cortelezzi et al., 1998; Kim, 2003). The problem formulation and a strategy to solve it via batch approach are presented here.

7.2.1 Problem formulation

Consider the following linear discrete time-invariant (DTI) system:

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k \tag{7.1}$$

$$\mathbf{y}_k = \mathbf{C}\mathbf{x}_k + \mathbf{D}\mathbf{u}_k \tag{7.2}$$

Here $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{u} \in \mathbb{R}^m$ and $\mathbf{y} \in \mathbb{R}^p$ are the state, input and output vectors, respectively. The state-transition matrix $\mathbf{A}$ contains $n \times n$ elements (assumed here being real). The eigenvalues of $\mathbf{A}$ define the stability of the system. In particular, the system is stable if all eigenvalues have a norm smaller than 1. The matrices $\mathbf{B}$, $\mathbf{C}$, $\mathbf{D}$ are referred to as input (or control) matrix, output (or measurement) matrix, and feedthrough matrix. For a single input system, $\mathbf{B}$ is simply a column vector of size $n \times 1$, weighing the effect of the input on each of the state variables. The output matrix $\mathbf{C}$ contains $p \times n$ elements if p linear combinations of the n state variables are measured; in the case of full-state measurement $\mathbf{C}$ is the identity matrix $\mathbb{I}_n$ of size $n \times n$. The feedthrough matrix $\mathbf{D}$, of size $p \times n$ represents the effect of the input directly on the measured output variables, without any interaction with the system dynamics.

We will assume in the remainder of this section full-state measurement (i.e., $\mathbf{C} = \mathbb{I}_n$) with no feedthrough ($\mathbf{D} = \mathbf{0}$).

Preview Sample

Assume that the objective is to regulate the system to the origin, i.e., $\mathbf{x} = 0$, with constrained states $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^n$ and output $\mathbf{u} \in \mathcal{U} \subset \mathbb{R}^m$. The sets of admissible states $\mathcal{X}$ and inputs $\mathcal{U}$ are generally defined by polyhedra. We will thus assume that:

$$\begin{aligned} \mathbf{x} \in \mathcal{X} &= \{\mathbf{x} \in \mathbb{R}^n : \mathbf{F_X x} \leq \mathbf{g_x}\} \\ \mathbf{u} \in \mathcal{U} &= \{\mathbf{u} \in \mathbb{R}^m : \mathbf{F_U u} \leq \mathbf{g_u}\} \end{aligned} \tag{7.3}$$

where $\mathbf{F_X}$ and $\mathbf{F_U}$ are the matrices containing the coefficients of the linear inequalities. Assuming a quadratic cost function, the optimization problem is formulated as:

$$\begin{aligned} \min_{\mathbf{z}} \quad & \sum_{k=0}^{N-1} (\mathbf{x}_k^T \mathbf{Q} \mathbf{x}_k + \mathbf{u}_k^T \mathbf{R} \mathbf{u}_k) + \mathbf{x}_N^T \mathbf{P} \mathbf{x}_N \\ \text{s.t.} \quad & \mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k, \quad k = 0, \ldots, N-1 \\ & \mathbf{x}_k \in \mathcal{X}, \quad k = 0, \ldots, N-1 \\ & \mathbf{u}_k \in \mathcal{U}, \quad k = 0, \ldots, N-1 \\ & \mathbf{x}_N \in \mathcal{X}_f \\ & \mathbf{x}_0 \text{ is given} \end{aligned} \tag{7.4}$$

where $\mathbf{z} = \mathbf{u}_0, \ldots, \mathbf{u}_{N-1}$ is the set of inputs, N is the number of time steps of the prediction window, $\mathbf{Q} \in \mathbb{R}^{n \times n}$, $\mathbf{P} \in \mathbb{R}^{n \times n}$, $\mathbf{R} \in \mathbb{R}^{m \times m}$ are respectively the cost matrices for state, terminal state (i.e., the state at the end of the prediction horizon) and control inputs. $\mathcal{X}_f$ denotes the terminal set. The elements of $\mathbf{Q}$ determine the penalty for deviations of the states from the desired values. The matrix $\mathbf{Q}$ should be symmetric and positive semi-definite (i.e., $\mathbf{x}^T \mathbf{Q} \mathbf{x} \geq 0$ for all $\mathbf{x} \neq 0$). The elements of $\mathbf{R}$ define the penalty applied to the use of control inputs. Large values discourage strong control actions. Thus, the elements of $\mathbf{R}$ are often tuned, considering energy expenditure or actuators' wear. The matrix $\mathbf{R}$ should be symmetric and positive definite (i.e., $\mathbf{u}^T \mathbf{R} \mathbf{u} > 0$ for all $u \neq 0$), to ensure that control actions are always penalized. The penalties in $\mathbf{P}$ are set to ensure desirable features at the end of the prediction horizon (for instance, stability or convergence to a desired state). The value of the elements $\mathbf{P}$ can be determined, for instance, by solving a Riccati equation in the context of linear quadratic regulator (LQR) control, as discussed by Mayne et al. (2000).

Preview Sample

Not available in preview

the nonlinear behavior of turbulent flows. Variants of MPC, such as Robust MPC and Stochastic MPC, are designed to take into account the parametric uncertainty of the model, external disturbances, etc. Nonetheless, modeling these effects is often a complicated task, and implementation of Robust and Stochastic MPC entails a substantial increase in the computational cost of the procedure, which would be very challenging to afford for real-time control. Furthermore, latency and time delays of turbulent flows introduce the additional complication of potential shift between actuation and effect on the flow. Longer prediction horizons are required to account for this issue. This further pushes towards models capable of delivering reliable long-term predictions of the flow, at least for those directions object of the control. The rewards for addressing these challenges are significant; Bewley et al. (2001) identified already more than 20 years ago that efficient actions for flow control should be long-sighted, at least in the framework of relaminarization and drag reduction of wall-bounded flows.

- **Dependence on the expertise of the user**: the performances of MPC depend on the selection of hyperparameter and, consequently, on user expertise. Learning-based MPC offers solutions to learn parameters based on simpler performance indicators. It is foreseeable that there will be a more intense blending in the future of MPC with deep-learning methods to leverage this principle.

In conclusion, while the limitations underscore the current challenges, they also unveil interesting opportunities for refining Model Predictive Control in fluid flow applications.

Preview Sample

Preview Sample

8

Fundamentals of Dimensionality Reduction

Miguel A. Mendez,* Romain Poletti and Lorenzo Schena

von Karman Institute for Fluid Dynamics,
Sint-Genesius-Rode, Belgium

This chapter reviews the fundamentals of dimensionality reduction, a subset of machine learning that seeks to identify low-dimensional representation of high-dimensional data. This is the essence of many data processing tasks, from filtering to pattern identification. The lecture begins with a brief review of the general concepts and the two main families of methods, namely autoencoders and manifold learning. We then move to *linear autoencoders*, which encompasses various popular decompositions in fluid dynamics, such as the Proper Orthogonal Decomposition, Dynamic Mode Decomposition and their variants (e.g. Spectral POD (sPOD) and Multiscale POD (mPOD)). Finally, we complete the tour with a brief overview of nonlinear methods for manifold learning, including kernel PCA, Isometric Mapping (ISOMAP), Locally Linear Embedding (LLE), and t-SNE (t-Distributed Stochastic Neighbor Embedding). To promote hands-on experience, this lecture provides two tutorial sessions. The Python codes for all exercises are provided in the course repository.

Preview Sample

8.1 A note on notation and style

We stick to the notation introduced in 2.1, with the following additional elements:

Sampling The sampling of a continuous function is stored in a vector or a matrix. We assume uniform sampling both in space and time. For vector $\boldsymbol{d}(t)$ sampled the time domain t, considering a discretization $t_k = k\Delta t$, with $f_s = 1/\Delta t$ the sampling frequency and $k = [0, 1, \dots n_t - 1]$, we could write $\boldsymbol{d}[k]$ or $\boldsymbol{d}_k$ or $\boldsymbol{d}(t_k)$. The same is true for the space domain, although a matrix linear index must be introduced. This is important when we transform a matrix (for

* mendez@vki.ac.be.

example, a spatial realization of a quantity) into a vector.

For example, let $p[i, j]$ be the 2D discretization of a pressure field $p(x, y)$, where the axes were discretized as $i \in [0, n_x - 1]$ and $j \in [0, n_y - 1]$. For the purpose of this lecture, that field would be written as a single "snapshot" vector $\boldsymbol{p} \in \mathbb{R}^{n_s}$, with $n_s = n_x n_y$. The entries in this vector would be accessed with a matrix linear index, denoted in bold, i.e. $\boldsymbol{p}[\boldsymbol{i}]$. The way this accesses the data in the matrix depends on whether the flattening is performed column-wise or row-wise. For example, for a matrix $\boldsymbol{A} \in \mathbb{R}^{3\times3}$, the column-wise and the row-wise matrix indices are*

$$\text{column-wise } \boldsymbol{i}: \ \boldsymbol{A} = \begin{bmatrix} 0 & 3 & 6 \\ 1 & 4 & 7 \\ 2 & 5 & 8 \end{bmatrix} \quad \text{row-wise } \boldsymbol{i}: \ \boldsymbol{A} = \begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{bmatrix}, .$$

In a vector quantity, e.g. a velocity field $\boldsymbol{U}[u(\boldsymbol{x_i}), v(\boldsymbol{x_i})] \in \mathbb{R}^{n_s\times2}$, we consider that the reshaping stacks all the components one below the other producing a state vector of size $n_s = n_C\, n_x\, n_y$ where $n_C = 2$ is the number of velocity components.

Preview Sample

8.2 General Concepts

Dimensionality reduction aims to represent high-dimensional data in a lower-dimensional space while retaining the key features and patterns of variability. The underlying assumption is that, although the data may appear high-dimensional, its meaningful structure often lies on a much lower-dimensional manifold. For example, in face recognition, the variability in images of different individuals can often be explained by a few dominant factors such as pose, lighting, or expression. By identifying these dominant patterns, one can construct a reduced set of basis images that retains the information necessary for recognition (Swets & Weng, 1996; Turk & Pentland, 1991).

In its most general form, the process of dimensionality reduction is an "information bottleneck" (Vladimir Cherkassky, 2008) composed of an encoder mapping and a decoder mapping, as shown in the schematic of Figure 8.1. A high dimensional vector $\boldsymbol{x} \in \mathbb{R}^{n_x}$ is mapped onto a low dimensional one $\boldsymbol{z} \in \mathbb{R}^{n_z}$ (with $n_z \ll n_x$) by a mapping $\boldsymbol{z} = \mathcal{E}(\boldsymbol{x}|\boldsymbol{X}^*)$, called **encoder**, which must be

* Recall that here use Python-like indexing. Hence, the first entry is 0 and not 1

inferred from a set of training data $\boldsymbol{X}^* \in \mathbb{R}^{n_x \times n_t}$. The opposite process is carried out by a mapping $\tilde{\boldsymbol{x}} = \mathcal{D}(\boldsymbol{z}|\boldsymbol{Z}^*)$, called **decoder**, which seeks to bring the mapping back to the original dimension and is trained on a set of training data $\boldsymbol{Z}^* \in \mathbb{R}^{n_z \times n_t}$. The composition of the encoder and the decoder is referred to as **autoencoder** and the space $\mathbb{R}^{n_z}$ is referred to as **latent space**. Both the encoder and the decoder functions can take the form of a parametric or nonparametric function (see Chapter 2).

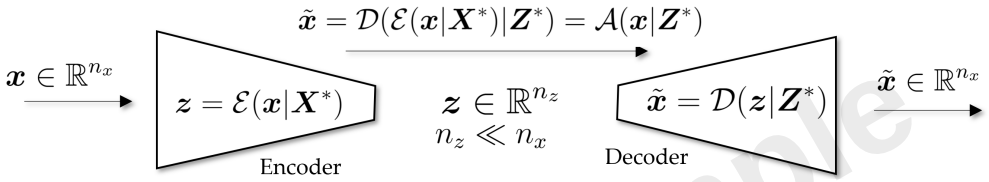


Figure 8.1: Schematic illustration of the general process of dimensionality reduction, viewed as an "information bottleneck". The encoder maps a high-dimensional input to a low-dimensional one, often called latent space. The decoder does the opposite.

Autoencoders provide a general framework for dimensionality reduction. The goal is to learn a compressed representation $\boldsymbol{z}$ of the data $\boldsymbol{x}$ such that an approximate reconstruction $\tilde{\boldsymbol{x}}$ can still be obtained and the essential features of the original data are preserved. This is useful for at least three reasons. First, it offers interpretability and economy: if the relevant variability in the data is contained in $n_z \ll n_x$ dimensions, then the remaining $n_x - n_z$ dimensions can be ignored. Second, the reduced representation $\boldsymbol{z}$ may serve as input to simpler models or supervised learning methods, lowering computational cost. Third, by focusing on salient features, the autoencoder can learn to discount irrelevant variability, such as noise or outliers—indeed, most denoising strategies can be viewed as autoencoding.

A related but distinct class of techniques is **manifold learning**. These methods do not aim to reconstruct the original data and therefore do not require a decoder. Instead, their objective is to embed the data in a lower-dimensional space while preserving some notion of similarity or neighborhood structure. The central assumption (Zheng & Xue, 2009) is that the high-dimensional data lie on (or near) a low-dimensional manifold, and the task is to uncover coordinates on this manifold that faithfully represent the relationships between data points. Manifold learning is widely used for visualizing high-dimensional

Preview Sample

datasets, detecting clusters or anomalies, and exploring geometric structure.

The distinction between autoencoders and manifold learning is subtle but important. While certain autoencoder architectures can serve as manifold learners and vice versa, most manifold learning algorithms produce compressed representations that are *not* designed to be decoded back to the original high-dimensional space.

8.3 Linear autoencoding and manifold learning: PCA

We consider first a simplified toy problem. Consider the dataset in Figure 8.2. This set consists of $n_p = 150$ datapoints $\boldsymbol{x} \in \mathbb{R}^{3\times 1}$ that have been sampled from three distributions in a three-dimensional space. The color in the markers encodes the different distributions (clusters). The dataset can be compactly represented by a dataset matrix $\boldsymbol{X} \in \mathbb{R}^{3\times 150}$ with each point (vector) along its columns. The data has been mean-centered (i.e. the mean vector over the rows of $\boldsymbol{X}$ is zero).

Preview Sample

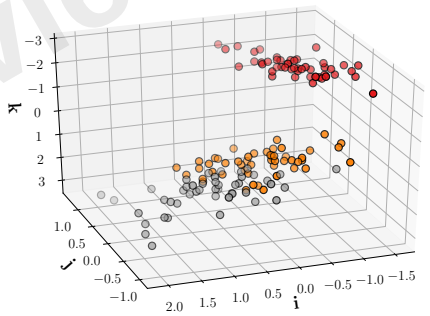


Figure 8.2: Plot of the dataset considered in this section: $n_p = 150$ belonging to three probability density functions are sampled in a three-dimensional space.

To illustrate the main ideas behind dimensionality reduction, we here seek to identify the best 2D representation of the data. That is, we seek a compression ratio 3:2, from $\mathbb{R}^3$ to $\mathbb{R}^2$. A simple approach would consist in projecting along any of the planes defined by the Cartesian coordinate frame. The projection requires defining a basis matrix, here denoted by $\boldsymbol{B}$, whose columns collect the basis vectors in the projected space. For instance, the projection onto the $(\boldsymbol{i}, \boldsymbol{j})$

plane is

$$\boldsymbol{Z}_B = \boldsymbol{B}^T\boldsymbol{X} \quad \text{with} \quad \boldsymbol{B} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \qquad \boldsymbol{Z}_B \in \mathbb{R}^{2\times 150}\,. \tag{8.1}$$

We evaluate the quality of a projection by the amount of information it preserves. The operation in (8.1) is a linear **encoder**. The matrix $\boldsymbol{B}$ has no inverse. Nevertheless, it is possible to show* that the "best possible" inversion of (8.1), in an l_2 sense, is

$$\tilde{\boldsymbol{X}}_B = \boldsymbol{B}\boldsymbol{Z}_B = \boldsymbol{B}\boldsymbol{B}^T\boldsymbol{X} = \mathcal{A}_B\boldsymbol{X}, \tag{8.2}$$

where $\mathcal{A}_B = \boldsymbol{B}\boldsymbol{B}^T$ is the linear **autoencoder** associated with the basis $\boldsymbol{B}$. In this simple example, the autoencoder removes the last row of $\boldsymbol{X}$, and this information cannot be recovered.

We measure the performance of this linear autoencoder using the l_2 norm:

$$J(\boldsymbol{B}) = \|\boldsymbol{X} - \tilde{\boldsymbol{X}}_B\|_2 = \|\boldsymbol{X} - \mathcal{A}_B\boldsymbol{X}\|_2\,. \tag{8.3}$$

A natural question is that of identifying the basis vectors that allows to minimize (8.3). This leads to the **Principal Component Analysis** (PCA). Without entering into the details of the derivation,† the basis that leads to this optimal reconstruction is the solution of the following eigenvalue problem:

$$(\boldsymbol{X}\boldsymbol{X}^T)\boldsymbol{u}_j = \lambda_j\boldsymbol{u}_j \Longleftrightarrow (\boldsymbol{X}\boldsymbol{X}^T)\boldsymbol{U} = \boldsymbol{\Lambda}\boldsymbol{U}, \tag{8.4}$$

where the eigenvectors $\boldsymbol{u}_j$ are the **principal components**, collected as columns of the matrix $\boldsymbol{U} \in \mathbb{R}^{3\times 150}$ and the eigenvalues λ_j (collected in the diagonal of $\boldsymbol{\Lambda} \in \mathbb{R}^{3\times 3}$) controls their relative importance. It is easy to show that these are non-negative positive numbers and (8.4) is the first step to compute the **Singular Value Decomposition** (SVD) of $\boldsymbol{X}$. Keeping the focus on the definition of the best linear autoencoder, we have

$$\tilde{\boldsymbol{X}}_U = \boldsymbol{U}\boldsymbol{Z}_U = \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{X} \Longleftrightarrow \tilde{\boldsymbol{x}}_k = \sum_{i=1}^{2} c_{kj}\boldsymbol{u}_j \text{ with } c_{kj} = \boldsymbol{u}_j^T\boldsymbol{x}_k, \tag{8.5}$$

* This follows from minimizing $\|\boldsymbol{X} - \boldsymbol{B}\boldsymbol{Z}\|_2$ with respect to $\boldsymbol{Z}$: the optimal solution is $\boldsymbol{Z} = \boldsymbol{B}^T\boldsymbol{X}$, which is equivalent to using the Moore–Penrose pseudoinverse of $\boldsymbol{B}$.

† An extensive presentation is proposed by Jolliffe (2002) and Bishop et al. (2006).

Preview Sample

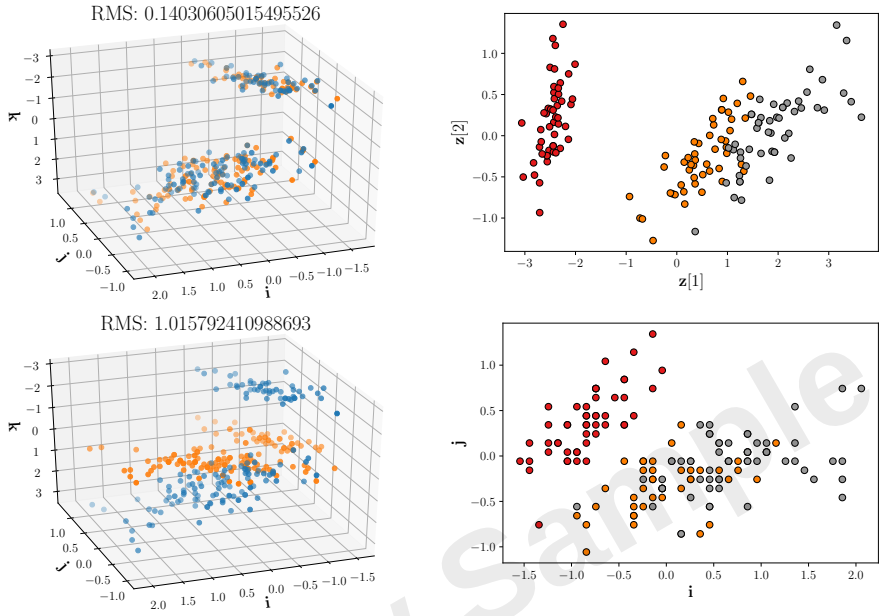


Figure 8.3: Comparison of the data reconstruction via the autoencoder $\mathcal{A}_V$ (top left) versus the autoencoder $\mathcal{A}_B$ (bottom left). On the right, the data is shown in the 2D representations provided by each base, keeping the cluster colour code.

Preview Sample

with $\mathcal{A}_U = \boldsymbol{U}\boldsymbol{U}^T$ the PCA autoencoder. The summation on the right recalls that each of the approximation ($\tilde{\boldsymbol{x}}_k$) of a data point ($\boldsymbol{x}_k$) is a linear combination of the principal components $\boldsymbol{u}_j$. The coefficients $c_{kj} = \widehat{\boldsymbol{x}}_U[j] \in \mathbb{R}^{n_z}$, written as a vector over the index j, is the PCA-transformed data: that is the projection of the k-th data point $\boldsymbol{x} \in \mathbb{R}^{n_x}$ onto the principal components.

This is the encoding function $\mathcal{E}$ introduced in Figure 8.1; the entries of the encoded representation are inner products:

$$\boldsymbol{z}_i = \mathcal{E}(\boldsymbol{x}_i) \rightarrow \boldsymbol{z}_i[j] = \boldsymbol{x}_i^T \boldsymbol{u}_j = \boldsymbol{u}_j^T \boldsymbol{x}_i \,. \tag{8.6}$$

The decoding function for the PCA is simply $\tilde{\boldsymbol{x}}_i = \mathcal{D}(\boldsymbol{z}_i) = \boldsymbol{U}\boldsymbol{z}_i$.

Figure 8.3 shows a comparison between the PCA and the trivial projection onto $\boldsymbol{B}$. The figures on the left show the reconstructions $\tilde{\boldsymbol{X}}_U$ (top) and $\tilde{\boldsymbol{X}}_B$ (bottom) while the figures on the right show the data in the reduced space: the plane $(\boldsymbol{u}_1, \boldsymbol{u}_2)$ for the PCA and the plane $(\boldsymbol{i}, \boldsymbol{j})$ for the trivial projection. In the reconstruction plots, the blue markers indicate the original data, while the

orange ones indicate the reconstructions. In the 2D plots, the color code of the clusters is maintained. The normalized error (i.e. (8.3) divided by $\sqrt{3n_p}$) for the PCA is $E = 0.14$ while it is $E = 1.05$ for the trivial projection. The 2D representation from the PCA preserves more information and keeps the distant cluster (red markers) far from the other two as in the original 3D.

We just introduced PCA as a linear autoencoder, obtained by minimizing the l_2 reconstruction error in (8.2). However, PCA can also be derived a manifold learning perspective. Suppose that our goal in projecting the data from $\mathbb{R}^3$ to $\mathbb{R}^2$ is to preserve the linear correlations between data points. That is, we aim to maintain the pairwise inner products before and after projection:

$$J(\boldsymbol{B}) = \sum_{i=1}^{n_p}\sum_{j=1}^{n_p}\left(\boldsymbol{x}_i^T\boldsymbol{x}_j \;-\; \boldsymbol{z}_i^T\boldsymbol{z}_j\right)^2 = \left\|\boldsymbol{X}^T\boldsymbol{X} - \boldsymbol{Z}^T\boldsymbol{Z}\right\|_F^2, \qquad \boldsymbol{Z} = \boldsymbol{B}^T\boldsymbol{X}, \tag{8.7}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. This formulation depends only on the geometry of the data cloud (i.e., how points relate to each other), and not on reconstructing the original data. Therefore, (8.7) expresses dimensionality reduction as a *manifold learning* problem: we seek a low-dimensional embedding that preserves similarities. Minimizing (8.7) leads to the classical Multidimensional Scaling (MDS) algorithm, originally introduced by Torgerson (1952). An extended derivation is provided in Ghojogh & Crowley (2019). A useful observation arises by substituting the encoding into the inner product:

$$\boldsymbol{Z}^T\boldsymbol{Z} = \boldsymbol{X}^T\boldsymbol{B}\boldsymbol{B}^T\boldsymbol{X}. \tag{8.8}$$

To preserve correlations, we therefore seek a basis $\boldsymbol{B}$ such that $\boldsymbol{B}\boldsymbol{B}^T$ is as close as possible to the identity on the subspace spanned by the data. This is precisely the same condition encountered when minimizing the reconstruction loss in the linear autoencoder formulation. Thus, PCA appears in two complementary guises: (i) as the optimal linear autoencoder minimizing reconstruction error, and (ii) as a manifold learning method minimizing the loss of correlation structure under projection.

8.4 Data Driven Modal Analysis

In fluid dynamics, the datasets to be compressed often arise from numerical simulations or complex experiments that provide time-resolved information about

Preview Sample

the velocity field in a flow. Such flows are typically turbulent and therefore characterized by a wide range of spatial and temporal scales. Dimensionality reduction methods have thus been introduced to extract coherent patterns and flow structures from this apparent complexity.

One of the earliest and most influential tools developed for this purpose is the *Proper Orthogonal Decomposition* (POD), introduced by Lumley (1970); Lumley & Poje (1997). As we will show shortly, POD can be viewed as a direct generalization of the PCA framework discussed above.

The idea that turbulent flows can be interpreted as superpositions of coherent building blocks has motivated the "dynamical systems" perspective on turbulence (see (Berkooz et al., 1993; Holmes et al., 2012)) and the development of model order reduction techniques based on Galerkin projection (see (Cordier & Bergmann, 2013; Benner et al., 2015; Ahmed et al., 2021a; Benner, 2020; Dawson, 2023) for comprehensive reviews). Additionally, POD has been widely used as a filtering tool—for instance, for outlier removal (Raiola et al., 2015; Higham et al., 2016) and image processing (Mendez et al., 2017)—and as a basis for efficient regression and interpolation schemes (Ratz et al., 2022; Casa & Krueger, 2013; Bouhoubeiny & Druault, 2009).

Over the past decade, numerous extensions and variants of POD have been proposed, along with new modal decomposition techniques. More broadly, the effort to represent flows as linear combinations of "coherent patterns" (or *modes*) has grown into a well-established field commonly referred to as *data-driven modal analysis* (see (Taira et al., 2017, 2020)). In this section, we review several of the most widely used decompositions.

Preview Graphic

8.4.1 From Autoencoders to Modal Analysis

Our starting point is a dataset rearranged in the form of a *snapshot matrix* $\boldsymbol{D} \in \mathbb{R}^{n_s \times n_t}$. This contains n_t snapshots (columns) collecting n_s data points each. Using a Python-like notation, we denote as $\boldsymbol{d}_k = \boldsymbol{D}[:, k] \in \mathbb{R}^{n_s}$ the k-th column (snapshot) in $\boldsymbol{D}$. The reshaping into a column vector is carried out regardless of the nature of the dataset. Considering for example a 2D field acquired via planar Particle Image Velocimetry (PIV), providing a velocity field $\boldsymbol{u}(x_i, y_j, t_k) = (u(x_i, y_j, t_k), v(x_i, y_j, t_k))^T$ over a regular grid $\boldsymbol{x} = (x_i, y_j)$ with $i = 1, \dots n_x$ and $j = 1, \dots n_y$. A snapshot vector can be obtained by flattening the two scalar fields u, v into vectors and stacking one below the other to have $n_s = 2n_x n_y$. If the dataset contains a video sequence, then each snapshot is

obtained by reshaping the images.

We assume in this lecture that data is available in a Cartesian and uniform grid with spacing Δx in both directions. Then, in a first approximation, one can assume that the data is constant within an area Δx^2 (piece-wise constant interpolation). Denoting as Ω the domain (area) over which the velocity was sampled, the L_2 norm in Ω can then be linked to the l_2 norm in $\mathbb{R}^{n_s}$ (see Mendez et al. (2019)). Similarly, assuming that the snapshots have been collected at uniform time intervals Δt, so that $t_k = k\Delta t$, with $k = 0, \dots n_t - 1$ and $f_s = 1/\Delta t$ is the sampling frequency, the L_2 norm in the time interval of duration $T = n_t \Delta t$ can be linked to the l_2 norm in $\mathbb{R}^{n_t}$.

We here focus on dimensionality reduction technique in the *space* domain, i.e. along the columns of $\boldsymbol{D}$. Therefore, we seek to map our dataset $\boldsymbol{D} \in \mathbb{R}^{n_s \times n_t}$ into a compressed representation $\boldsymbol{Z} \in \mathbb{R}^{n_r \times n_t}$. The column $\boldsymbol{z}_k = \boldsymbol{Z}[:, k] \in \mathbb{R}^{n_r}$ is thus the reduced representation of the snapshot $\boldsymbol{d}_k \in \mathbb{R}^{n_s}$, with $n_r \ll n_s$. In modal analysis, it is particularly interesting to follow how $\boldsymbol{z}$ changes from snapshot to snapshot (thus in time).

From the example in Section 8.2, we know that the encoded representation corresponds to a projection onto a basis (see (8.1) and (8.6)). In this setting, we denote this basis by $\boldsymbol{\Phi}(\boldsymbol{x}) \in \mathbb{R}^{n_s \times n_r}$, whose columns contain the spatial modes. Once reshaped back into their physical dimensions, each column of $\boldsymbol{\Phi}$ corresponds to a flow field. The matrix of temporal coefficients is denoted by $\boldsymbol{Z}(t_k) \in \mathbb{R}^{n_r}$.

In a general linear autoencoder formulation, the encoding and decoding operations are written as

$$\boldsymbol{Z}(t_k) = \mathcal{E}\big(\boldsymbol{D}(\boldsymbol{x}, t_k)\big) = \boldsymbol{\Phi}^T \boldsymbol{D}(\boldsymbol{x}, t_k), \tag{8.9}$$

$$\tilde{\boldsymbol{D}}(\boldsymbol{x}, t_k) = \mathcal{D}\big(\boldsymbol{Z}(t_k)\big) = \boldsymbol{\Phi}\, \boldsymbol{Z}(t_k). \tag{8.10}$$

If the columns of $\boldsymbol{\Phi}$ are not orthonormal, the least-squares decoder* is used instead:

$$\tilde{\boldsymbol{D}}(\boldsymbol{x}, t_k) = \big(\boldsymbol{\Phi}^T \boldsymbol{\Phi}\big)^{-1} \boldsymbol{\Phi}^T \boldsymbol{D}(\boldsymbol{x}, t_k). \tag{8.11}$$

In the case of a PCA of the snapshot matrix $\boldsymbol{D}$, the spatial structures have a special property and we denote them as $\boldsymbol{\Phi}_{\mathscr{P}}$. Following (8.4), these are solution of the following eigenvalue problem:

* This least-squares reconstruction is the same expression encountered in Chapter 2 when training linear parametric models. A detailed discussion of encoding in space vs. time is provided in Mendez (2023b).

Preview Sample

Not available in preview

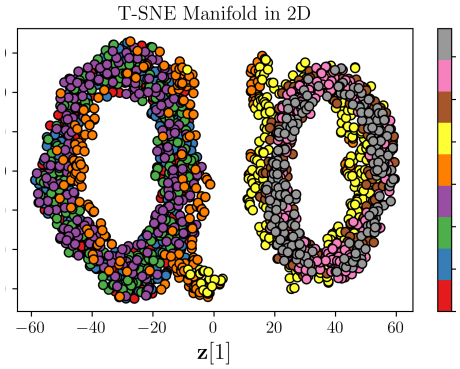


Figure 8.16: Mapping of the cylinder dataset onto $\mathbb{R}^2$ using t-SNE.

8.8 Summary and Conclusions

This chapter surveyed linear and nonlinear dimensionality reduction methods, emphasizing the distinction between autoencoders (which focus on reconstruction) and manifold learning (which focuses on preserving relational geometry). We began with PCA—a foundational linear method interpretable both as an optimal linear autoencoder and as a manifold-learning procedure preserving pairwise correlations—and then branched into two directions: (i) linear methods for data-driven modal analysis and (ii) nonlinear manifold learning, which can uncover low-dimensional embeddings even when the underlying structure is curved. While the latter remains relatively unexplored in fluid mechanics, the tutorials presented here aim to provide intuition and encourage further use.

Regarding data-driven modal analysis, the main conclusions from the four investigated decompositions (POD, DMD, SPODs, mPOD) are as follows:

1. **POD** offers the strongest autoencoding and therefore the most effective dimensionality reduction. This makes it ideal for data compression, noise reduction, and statistical filtering. However, its modes typically contain a broad range of frequencies, which can make physical interpretation less transparent.

2. **DMD** provides the cleanest frequency separation and, in principle, can identify the normal modes of the underlying dynamics. It is the method of choice for data-driven stability analysis and for flows dominated by

Preview Sample

coherent oscillations at fixed frequencies (e.g., vortex shedding, cavity tones). However, DMD relies on representing the temporal dynamics of each mode as a single complex exponential. When the dataset contains intermittent phenomena, time-varying frequencies, or broadband structures (as is common in experimental and turbulent flows), the DMD modes either decay rapidly (modulus < 1) or distribute across many weak modes. In such cases, the decomposition does not converge.

3. **Spectral POD (Towne et al. 2018)** provides frequency-resolved modes with strong robustness to noise, thanks to its Welch-type averaging. This makes it particularly well-suited for stationary datasets with long time histories. It is primarily an *analysis* tool rather than a compression method (it generates far too many modes!), and it is not directly applicable to transient flows.

4. **Filtered SPOD (Sieber et al. 2016)** provides a simple and tunable bridge between POD and harmonic decompositions by applying a diagonal temporal filter to the correlation matrix. It is robust to noise and practical for exploratory analysis of leading frequency ranges. However, the relationship between the filter parameter and the resulting spectral selectivity is indirect, and the method assumes stationarity.

5. **mPOD** allows explicit partitioning of the data into prescribed frequency bands while retaining energy optimality within each band. It is well suited for multiscale and transient flows and provides useful time–frequency localization. However, it involves more hyperparameters and requires user experience to tune. Moreover, when the prescribed bands become too narrow, the underlying filter design becomes poorly localized, making mPOD unsuited for identifying purely harmonic modes.

Finally, the nonlinear manifold learning examples demonstrated that different similarity metrics (Euclidean, geodesic, or local linear) lead to different low-dimensional embeddings, yet all revealed qualitatively consistent low-dimensional structure. In our case, a dataset in $\mathbb{R}^{4260}$ could be meaningfully represented in just 2 or 3 dimensions. Whether such reduced spaces can be used to formulate predictive dynamical models—and then be reliably lifted back to physical space—remains one of the most exciting open frontiers in physics-informed machine learning and reduced-order modeling.

Preview Sample

9

Data-driven post-processing and reduced order modelling

Ashton Ian Hetherington, Adrián Corrochano,
Soledad Le Clainche*
School of Aerospace Engineering,
Universidad Politécnica de Madrid, Spain

Preview Sample

This chapter delves into data-driven equation-free models, focusing on integrating physical insights through reduced order models (ROMs). It emphasizes the transformative potential of these approaches in advancing our understanding of complex non-linear dynamical systems. The chapter reviews cutting-edge techniques and case studies across various industries, demonstrating the power of data-driven modeling. The chapter introduces data analysis and post-processing methods, resulting in precise and entirely data-driven hybrid ROMs. By combining modal decomposition and deep learning, these methods unveil fundamental patterns in dynamic systems, enhancing our grasp of underlying physics. They also enable database reconstruction from limited measurements and forecasting of system dynamics. These hybrid methods, combining experimental and numerical data, offer accurate alternatives to resource-intensive numerical simulations, reducing computational costs. They prove versatile for optimization and control, particularly in fluid mechanics, providing valuable insights into complex nonlinear dynamical systems. The chapter elucidates the mathematical foundations of these hybrid methods and provides practical application examples, offering a comprehensive guide to their implementation and potential in both scientific and practical contexts.

9.1 Introduction

In recent years, the proliferation of high-quality data has ushered in a transformative era in the realms of machine learning and reduced order modeling. This surge in data availability has paved the way for novel approaches to un-

* soledad.leclainche@upm.es.

raveling the intricacies of complex non-linear dynamical systems without the need for a priori knowledge of governing equations. In this exciting landscape, data-driven equation-free models have emerged as a beacon of promise, offering an alternative path to comprehending and manipulating these intricate systems. Using the power of machine learning, these models have the capacity to extract knowledge directly from data, bypassing the traditional reliance on theoretical principles. This paradigm shift encourages the exploration of data-driven avenues to unveil new hypotheses and construct models that hitherto may have remained concealed.

The implications of this transformation extend far and wide, with applications spanning diverse industries such as aerospace, automotive, construction, pharmaceuticals, chemicals, manufacturing, and more. The ability to harness data to decipher complex systems has the potential to revolutionize problem-solving and decision-making across these domains.

Broadly categorized, there are two primary approaches to data-driven modeling, each offering unique advantages and insights. The first approach centers on data forecasting models, which are adept at predicting future data trajectories through the utilization of machine learning techniques, prominently including deep neural networks. Notably, these models do not incorporate explicit physical principles into their construction. In contrast, the second approach is exemplified by reduced order models (ROMs) enriched with physical insights. These hybrid models skillfully integrate an understanding of the underlying physics with advanced pattern identification techniques such as proper orthogonal decomposition Sirovich (1987) and dynamic mode decomposition Schmid (2010). This amalgamation of physical principles and data-driven methodologies enables the extraction of pertinent spatio-temporal information from the data, providing a deeper understanding of the system's behavior.

The integration of hybrid data-driven ROMs into the study of non-linear dynamical systems brings forth a multitude of advantages over relying solely on deep neural networks. These ROMs empower researchers to identify key instabilities and mechanisms within the studied systems, shedding light on critical aspects of the underlying physics. Furthermore, they pave the way for the development of potent tools for optimization and control. Armed with a profound understanding of the physical intricacies at play, these models facilitate more accurate system phase predictions, the adoption of controlled and robust strategies, a reduction in computational costs for numerical simulations, and the streamlining of information collection in experimental setups.

Preview Sample

In this Chapter, we explore data-driven equation-free models, emphasizing the integration of physical insights through ROMs. We delve into the opportunities and challenges presented by these innovative approaches, while highlighting their potential to reshape our understanding of complex nonlinear dynamical systems. Through a comprehensive review of current state-of-the-art techniques and case studies spanning multiple industries, we aim to demonstrate the power of data-driven modeling in advancing scientific knowledge and practical applications.

More specifically, this chapter introduces two distinctive data-driven methodologies for the development of ROMs: modal decomposition tools and deep learning architectures. Modal decomposition techniques, notably prevalent in fluid dynamics, serve as invaluable instruments for extracting the physical patterns driving the dynamical systems in question. These patterns are often intimately linked to flow instabilities that trigger profound changes within the flow itself. By distilling the dimensionality of the system to a select few relevant patterns that encapsulate the flow dynamics, these methodologies lay the foundation for more concise and interpretable models. It is within this reduced dimensionality framework that deep learning architectures are employed, enabling the creation of diverse ROMs grounded in the principles of flow physics, often referred to as hybrid physics-aware ROMs, with broad-ranging applications.

The synergy between modal decomposition and deep learning techniques offers a clear advantage, fostering the development of hybrid physics-aware ROMs, also known as hybrid machine learning tools. In the expansive realm of fluid dynamics, where datasets are characterized by immense dimensions, often reaching tens of millions of grid points in typical computational fluid dynamics (CFD) problems, the integration of these methodologies becomes not only advantageous but also essential. Such an approach alleviates computational bottlenecks associated with purely deep learning models, streamlining the calibration of neural network schemes (NN) and reducing the overall computational cost. This reduction in dimensionality facilitates the implementation of more straightforward schemes, such as one-dimensional designs, as opposed to the computationally intensive three-dimensional architectures. In doing so, it opens new horizons for the efficient modeling of complex systems within fluid dynamics and beyond.

The models presented in this Chapter, whether they are solely based on modal decomposition or hybrid models based on deep learning and physical principles,

Preview Sample

can be used to develop three different applications: (1) patterns identification, suitable to study the physics behind the data analyzed; (2) data reconstruction, capable of reconstructing two- or three- dimensional databases from a set of selected points, using data from sensors, or repairing missing data; (3) data forecasting, which builds ROMs to predict the spatio-temporal evolution of the signal analyzed. This chapter reviews the mathematical background behind each method and will illustrate some of their most relevant applications.

This summary reveals the fusion of modal decomposition and deep learning methodologies, promising to illuminate the path toward enhanced understanding, more efficient modeling, and transformative applications in the realm of reduced-order modeling. As we embark on this exploration, we invite the reader to delve into the depths of these innovative techniques, unlocking the potential to bridge the gap between data-driven and physics-informed models and advance the frontiers of knowledge and practical implementation.

Preview Sample

9.2 Methodology

This section introduces the methodology encompassing modal decomposition and deep learning architectures, with a focus on developing diverse ROMs. These ROMs, refined through complex database treatments, become versatile tools for addressing a wide array of complex applications. As previously introduced, the deep learning models are actually hybrid since modal decomposition algorithms are integrated into these in the data pre-processing stage: as mentioned before, these models are known as hybrid physic-aware ROMs. Compressing the input data down to its main features and then training a neural network with this data increases the model's performance whilst reducing its computational cost. The Python codes of the hybrid methods presented in this chapter can be found in ModelFLOWs-app.*

These fully data-driven methods inputs datasets in matrix form, consisting of a set of K snapshots $\mathbf{v}_k = \mathbf{v}(t_k)$, where t_k is the time value at instant k, which are conveniently collected into a *snapshot matrix* as

$$\boldsymbol{V}_1^K = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k, \mathbf{v}_{k+1}, ..., \mathbf{v}_{K-1}, \mathbf{v}_K]. \tag{9.1}$$

Depending on the dataset's complexity, it might be more advantageous to restructure the data into a "snapshot tensor." This involves segregating and

* `modelflows.github.io/modelflowsapp/`

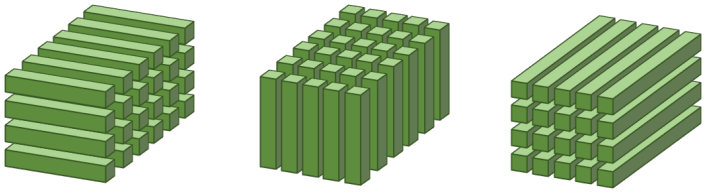


Figure 9.1: The fibers of a third-order tensor.

arranging the various components of the dataset into distinct tensor components while also separating the spatial coordinates for improved organization. A fluid dynamics dataset is usually formed by velocity components, but other components can also be included as part of the database analyzed (depending on the problem under study). For example, in atmospheric boundary layer flows, such as those encountered in meteorology, we typically consider both velocity and pressure fields; when addressing turbulent combustion problems, our dataset expands to encompass temperature, pressure fields, and various chemical species; in the case of flutter analysis during flight tests, an array of accelerometers strategically placed on the aircraft's surface gathers crucial system data, forming a signal that undergoes analysis to predict the evolution of this instability, among other scenarios, et cetera.

Within a snapshot tensor, the snapshot matrix is accommodated within a multidimensional array, contingent on several indices. The *fibers* of the snapshot tensor are formed by the corresponding rows and columns of the matrix. An illustrative example of a third-order tensor showcasing these tensor fibers is presented in Figure 9.1.

The methods outlined in this chapter typically employ fourth and fifth-order tensors for analyzing two-dimensional and three-dimensional datasets, respectively. To illustrate this concept more clearly, let us consider a two-dimensional dataset (a plane) with three velocity components: stream-wise velocity u_x, normal velocity u_y, both part of the in-plane velocity $\mathbf{u}$, in a Cartesian coordinate system with dimensions $J_2 \times J_3$, as follows:

$$\mathbf{u}(x_{j2}, y_{j3}, t_k) \quad for\ j_2 = 1, \ldots, J_2, \quad j_3 = 1, \ldots, J_3, \quad k = 1, \ldots, K. \tag{9.2}$$

The snapshot data can then be re-arranged into a fourth-order $J_1 \times J_2 \times J_3 \times K$-tensor $\boldsymbol{V}$, and now the components $V_{j_1,j_2,j_3,k}$ are defined as

$$V_{1,j_2,j_3,k} = u_x(x_{j2}, y_{j3}, t_k), \quad V_{2,j_2,j_3,k} = u_y(x_{j2}, y_{j3}, t_k). \tag{9.3}$$

Preview Sample

The index J_1 is used to organize the velocity components (in this case, $j_1 = 1, 2$ where $J_1 = 2$), J_2 and J_3 index the discrete values of the two spatial coordinates x and y, and K is utilized to index the time values. In the case of three-dimensional time-dependent datasets, the data is organized into a fifth-order $J_1 \times J_2 \times J_3 \times J_4 \times K$-tensor $\boldsymbol{V}$, and its components $V_{j_1,j_2,j_3,j_4,k}$ are defined as

$$\begin{aligned} V_{1j_2j_3k} &= v_1(x_{j_2}, y_{j_3}, z_{j_4}, t_k), \\ V_{2j_2j_3k} &= v_2(x_{j_2}, y_{j_3}, z_{j_4}, t_k), \\ &\cdots \\ V_{j_1j_2j_3k} &= v_{j_1}(x_{j_2}, y_{j_3}, z_{j_4}, t_k), \\ &\cdots \\ V_{J_1j_2j_3k} &= v_{J_1}(x_{j_2}, y_{j_3}, z_{j_4}, t_k). \end{aligned} \tag{9.4}$$

Similar to the fourth-order tensor, j_1 describes the velocity (or corresponding) components, j_2, j_3 and j_4 are used to index the discrete values of the three spatial coordinates x, y, and z, which correspond to the streamwise, normal and spanwise directions, and k indexes the time instant.

The snapshot matrix presented in eq. (9.1) and snapshot tensor from eq. (9.4) are related since the tensor indexes j_1, j_2 and j_3 (and j_4 for three-dimensional datasets) can be folded together into a single index j. So $\boldsymbol{V}_1^K \in \mathbb{R}^{J \times K}$, where $J = J_1 \times J_2 \times J_3 (\times J_4)$ forming the first dimension of the snapshot matrix.

In what follows, the quality of the ROMs developed by each one of the methods presented is measured via the relative root mean square error (RRMSE), which is computed as

$$RRMSE = \sqrt{\frac{\sum_{k=1}^{K} ||\mathbf{v}_k - \mathbf{v}_k^{approx.}||^2}{\sum_{k=1}^{K} ||\mathbf{v}_k||^2}}, \tag{9.5}$$

where $||\cdot||$ is the usual Euclidean norm and vectors $\mathbf{v}_k$ and $\mathbf{v}_k^{approx.}$ correspond to the real and approximated solutions.

9.2.1 Modal decomposition

The most popular modal decomposition methods generally used in fluid dynamics are Proper Orthogonal Decomposition (POD) Lumley (1967), Singular Value Decomposition (SVD) Sirovich (1987) and Dynamic Mode Decomposition (DMD)

Preview Sample

Schmid (2010). On the one hand, SVD and POD decompose a database as a group of SVD or POD modes orthogonal in space. Each mode is associated with a temporal coefficient, containing information about the system dynamics. Generally, SVD and POD are terms used interchangeably in the literature, although, as explained below, SVD is only one of the two possible methods that can be used to calculate the POD modes. On the other hand, DMD decomposes a database as a group of DMD modes. These modes are orthogonal in time (but not in space), so each mode is associated with a single frequency. Both POD (or SVD) and DMD are techniques suitable for identifying patterns representing flow physics and providing complementary results. While POD offers the advantage of significantly reducing data dimensionality in complex signals by capturing information across multiple frequencies within each mode, the characteristics of DMD differ. DMD modes are designed to identify individual and isolated frequencies associated with distinct dynamical events that govern the flow dynamics. In some cases, DMD modes can be connected to flow instabilities, triggering changes in the flow.

The good performance of these fully data-driven techniques to analyze data in complex non-linear dynamical systems encouraged several authors to develop extensions of these methos, to increase their robustness and widen their field of application to several research fields (different from fluid dynamics). In the literature, it is possible to find a bunch of examples (e.g., see some of these methods in Refs. Mendez et al. (2020b); Towne et al. (2018); Kutz et al. (2016b); Jovanović & Bamieh (2005), more examples and details can be found in the review article Ref. Begiashvili et al. (2023a)). This section will only focus on two methods, high order SVD (HOSVD) Tucker (1966) and higher order DMD (HODMD) Le Clainche & Vega (2017a), introducing some of their main applications.As elaborated upon below, these methods have been chosen for their demonstrated high-performance track record in analyzing and developing ROMs across a broad spectrum of fluid dynamics applications.

Preview Sample

Singular value decomposition (SVD) and proper orthogonal decomposition (POD)

Proper Orthogonal Decomposition (POD) is a mathematical approach initially proposed by Lumley Lumley (1967) to extract coherent structures from turbulent flows. The primary aim of POD is to decompose data into modes that optimize the mean square of the field variable under study. The dimension of the state

vector, contingent upon the spatial degrees of freedom within the dataset, is employed in assessing the covariance within the conventional approach utilized for computing the POD modal expansion. In dealing with large two-dimensional or three-dimensional problems, the computational demands of this approach can become notably onerous. Under such circumstances, Sirovich Sirovich (1987) pioneered the utilization of the singular value decomposition (SVD) method, which serves as an effective means to extract the POD modes.

SVD is a commonly employed factorization technique in the realm of fluid dynamics, particularly in the examination of turbulent flows. SVD decomposes a matrix into orthogonal directions represented by singular vectors, with corresponding singular values indicating the significance or magnitude of each direction in the transformation of the matrix. SVD finds applications beyond fluid dynamics, notably in the realm of low-rank matrix approximations. This method is advantageous since it reduces the amount of data to be examined, eliminates noise, and filters spatial redundancies Le Clainche et al. (2020a). Since, as mentioned, SVD is one of the two possible techniques that can be used to calculate POD modes, so both, POD and SVD modes can be considered the same (these terms are used indistinctly in the literature).

The SVD method starts by decomposing a collection of spatio-temporal data $v(x, y, z, t)$ into a set of proper orthogonal spatial modes, which are usually referred to as SVD or POD modes, and are represented by $\boldsymbol{\Phi}_n(x, y, z)$, which are weighted by the temporal coefficients $\mathbf{c}_n(t)$, as

$$\mathbf{v}(x, y, z, t) \simeq \sum_{n=1}^{N} \mathbf{c}_n(t)\, \boldsymbol{\Phi}_n(x, y, z). \tag{9.6}$$

The SVD method then factorizes the snapshot matrix $\boldsymbol{V}_1^k$, eq. (9.1), decomposing it into its spatial orthogonal modes $\mathbf{W}$, SVD or POD modes $\mathbf{T}$, and singular values $\boldsymbol{\Sigma}$ as

$$\boldsymbol{V}_1^K \simeq \mathbf{W}\, \boldsymbol{\Sigma}\, \mathbf{T}^\top. \tag{9.7}$$

where $^\top$ denotes the matrix transpose. The diagonal of matrix $\boldsymbol{\Sigma}$ contains the singular values $\sigma_1, \cdots, \sigma_K$, and $\mathbf{W}^\top\mathbf{W} = \mathbf{T}^\top\mathbf{T} =$ the $N \times N-$unit matrix, with N being the number of retained SVD modes. This parameter is also known as *spatial complexity* and will be referred to as so in the following sections when HODMD methodology is explained. It is important to differentiate between

Preview Content

J, the *spatial dimension* of the dataset, and the spatial complexity N, where $N \leq min(J, K)$.

SVD modes are organized in descending order based on their singular values. Typically, modes associated with the highest singular values depict the overarching system dynamics and embody coherent structures or patterns, while those linked to the lowest singular values might be omitted from the approximation, accounting for a certain level of uncertainty. These modes can be linked to noise, especially in experimental datasets featuring spatial redundancy. In the context of fluid dynamics, they are commonly (though not invariably) connected to small-scale flow features.

The number of N retained SVD modes for approximation to construct the expansion eq. (9.6), are gauged for a given tunable tolerance, ε_{svd}, which may be comparable to the amount of noise (in the case of experimental results), may be related to the size of the flow structures (in turbulent flows), etcetera, defined as

$$\sigma_{N+1}/\sigma_1 \leq \varepsilon_{svd}. \tag{9.8}$$

Note that the number of modes retained can be determined by applying different criteria, as discussed by Parente & Sutherland (2013a), where the POD or SVD approaches are also referred to as *Principal Component Analysis* (PCA).

An important facet of POD mode computation involves solving the eigenvalue problem associated with the covariance matrix. $\mathbf{S} = \boldsymbol{V}_1^K(\boldsymbol{V}_1^K)^\top$. This procedure is inherent to the SVD algorithm, which is employed to derive the POD modes. For the snapshot matrix given in Eq. (9.1), which comprises velocity vectors, this covariance matrix portrays the fluctuations in the squared magnitude of the velocity vector. This property corresponds to the concept of kinetic energy. Consequently, in fluid dynamics, the arrangement of POD modes is determined by their kinetic energy levels, as manifested in the singular values.

High order singular value decomposition (HOSVD)

The HOSVD algorithm was first introduced by Tucker in 1966 Tucker (1966) and with its implementation by de Lathauwer et al. De Lathauwer et al. (2000); De Lathawer et al. (2000), it has grown in popularity in recent years thanks to its high efficiency when applied in a diversity of fields.

Tensor-structured databases can undergo decomposition via HOSVD, a technique that applies SVD individually to each of the tensor fibers. For

Preview Sample

example, the HOSVD of a fifth-order tensor, as defined in Eq. (9.4), can be expressed as follows:

$$V_{j_1 j_2 j_3 j_4 k} \simeq \sum_{p_1=1}^{P_1} \sum_{p_2=1}^{P_2} \sum_{p_3=1}^{P_3} \sum_{p_4=1}^{P_4} \sum_{n=1}^{N} S_{p_1 p_2 p_3 p_4 n} W^{(1)}_{j_1 p_1} W^{(2)}_{j_2 p_2} W^{(3)}_{j_3 p_3} W^{(4)}_{j_4 p_4} T_{kn}, \quad (9.9)$$

where $\mathbf{S}_{p_1 p_2 p_3 p_4 n}$ is the *core tensor*, which is a fifth-order tensor, and the columns of the matrices $\mathbf{W}^{(1)}$, $\mathbf{W}^{(2)}$, $\mathbf{W}^{(3)}$, $\mathbf{W}^{(4)}$ and $\mathbf{T}$ are commonly known as the *modes* of the decomposition.

The initial group of modes, represented by the columns of matrices $\mathbf{W}^{(l)}$ for $l = 1, 2, 3$, and 4, are associated with both database components and spatial variables. Consequently, these modes are termed as the spatial HOSVD modes. On the other hand, the columns of the matrix $\mathbf{T}$ are linked to the time variable, earning them the designation of temporal HOSVD modes. The decomposition of singular values is now formed by five sets of values, $\sigma^{(1)}_{p_1}$, $\sigma^{(2)}_{p_2}$, $\sigma^{(3)}_{p_3}$, $\sigma^{(4)}_{p_4}$, and σ^{t}_{n}, which are arranged in decreasing order.

Similar to SVD, the HOSVD (9.9) is accurate without truncation. Nevertheless, truncation can be beneficial to filter noise and false artifacts or reduce the data dimensionality. Similar to SVD, each case's number of retained modes often depends on a tunable tolerance as $\sigma_{P_1+1}/\sigma_1 \leq \varepsilon_{svd_1}$, $\sigma_{P_2+1}/\sigma_1 \leq \varepsilon_{svd_2}$, $\cdots$, $\sigma_{N+1}/\sigma_1 \leq \varepsilon_{svd_5}$.

The tolerance is usually set to the same value, denoted as $\varepsilon_{svd_l} = \varepsilon_{svd}$, where $l = 1, 2, 3, 4, 5$. However, for highly intricate dynamics, it becomes crucial to adjust these tolerances according to the specific database under investigation. This flexibility represents one of the primary advantages of the HOSVD algorithm when compared to SVD, where the dimensionality reduction is uniform across all directions and components of the database. With HOSVD, it becomes feasible to differentiate between various levels of noise or component magnitudes.

Nevertheless, for highly intricate dynamics, it is imperative to customize these tolerances according to the specific database under scrutiny. One of the principal advantages of HOSVD over SVD lies in its uniform reduction of dimensionality across all directions and components within the database. This uniformity enables the discrimination between various magnitudes of noise levels or components when employing HOSVD.

After truncation, HOSVD expansion eq. (9.9) is written as

$$V_{j_1 j_2 j_3 j_4 k} \simeq \sum_{n=1}^{N} W_{j_1 j_2 j_3 j_4 n} \hat{V}_{kn}, \quad (9.10)$$

Preview Sample

Not available in preview

Two different architectures of neural networks are used, RNN and CNN. The model collects data from the transient stage of the numerical simulation and predicts its evolution in time. In Fig. 9.15, we present a representative snapshot of the CO_2 concentration along with the predictions and the temporal evolution of two characteristic points. Both architectures correctly predict the snapshots. The point situated in the far field is accurately predicted, as is the point located within the mixing layer between the fuel and the oxidizer. The ROM is more than 100 times faster than the CFD simulation and the RRMSE in the predictions is less than $3 - \%$ for the RNN architecture and 4% for the CNN.

9.4 Conclusions

Our exploration of data-driven equation-free models, with a strong emphasis on the integration of physical insights through ROMs, reveals a landscape brimming with potential for understanding and manipulating complex non-linear dynamical systems. This chapter has elucidated the transformative power of data-driven modeling, showcasing its capacity to advance both scientific knowledge and practical applications across diverse industries.

We introduced two pivotal data-driven methodologies: modal decomposition tools and deep learning architectures, each with its unique advantages and applications. Modal decomposition, rooted in fluid dynamics, provides a means to extract physical patterns governing dynamic systems. These patterns are intimately linked to flow instabilities, offering a deeper understanding of complex phenomena. Deep learning, on the other hand, leverages the prowess of neural networks to create versatile ROMs grounded in flow physics, forming the basis of hybrid physics-aware ROMs.

The synergy between these methodologies, demonstrated through hybrid physics-aware ROMs or hybrid machine learning tools, presents an opportunity to overcome computational bottlenecks, reduce costs, and streamline model development. This is especially crucial in domains with vast datasets, such as computational fluid dynamics, where dimensional reduction enables more efficient modeling.

We also highlighted the diverse applications of these models, including pattern identification, data reconstruction, and data forecasting, underscoring their adaptability and versatility. These applications provide insights into underlying physics, identifying modes, in some cases, connected to flow instabilities and

Preview Sample

triggering changes in the flow. This knowledge opens new avenues for flow control applications and reveal physical phenomena hitherto unknown in complex problems.

Our results presented focusing into the fusion of modal decomposition and deep learning methodologies serves as a summary to the potential of data-driven modeling. By combining modal decomposition methods, to reduce data dimensionality and extract physical patterns of the flow, with neural networks, we unlock new avenues for enhanced understanding and efficient modeling of complex systems. As we look ahead, these innovative techniques are poised to push the boundaries of knowledge and practical implementation, making data-driven equation-free models a powerful tool for tackling the challenges of the modern scientific landscape.

Acknowledgements The authors would like to express their gratitude to the research group ModelFLOWs, for their valuable discussions and assistance. Also, the authors would like to acknowledge the collaboration of Prof. R. Vinuesa (KTH) and Prof. A. Parente (ULB), who have contributed by sharing databases and engaging in fruitful discussions. The authors acknowledge the grants TED2021-129774B-C21 and PLEC2022-009235 funded by MCIN/AEI/ 10.13039/501100011033 and by the European Union "NextGenerationEU"/PRTR. The authors also acknowledge the grant PID2023-147790OB-I00 funded by MCIU/AEI/10.13039/501100011033/FEDER, EU.

Preview Sample

10

Data-Driven Modeling for Enhanced Aerodynamic

Philipp Bekemeyer,* Jigar Parekh, Simon Wassing and Derrick Armando Chaves Hines

Institute of Aerodynamics and Flow Technology, Braunschweig, Germany

Preview Sample

Machine learning and artificial intelligence techniques have transformed our everyday lives within the past few years. In areas where vast amounts of data are available, the aforementioned techniques have had tremendous success, especially when mathematical models are lacking. Instead, engineering tools in general and computational fluid dynamics tools in particular rely on first-order principles that directly enable the description and investigation of system behavior. Based on these principles, derived tools significantly contribute to the green transformation of the aviation sector. However, such tools are far from perfect and suffer several shortcomings, e.g., computational bottlenecks once a massive amount of simulations is required or the problem of deriving accurate turbulence models to describe small-scale turbulent behavior. Machine learning techniques are generally regarded as a possibility to enhance and complement first-order based numerical simulation tools to circumvent these shortcomings and yield a pathway towards enhanced aerodynamics. This lecture will shed some light on how machine learning can be used in this direction and is split into two parts. The first part focuses on how machine learning models can be employed during optimization and uncertainty quantification. This will be demonstrated based on an airfoil example. The second part will focus on special requirements for machine learning arising from the nature of fluid dynamics and showcase a few examples of research performed at the German Aerospace Center in this direction.

* philipp.bekemeyer@dlr.de.

10.1 Motivation

When designing and optimizing an aircraft, it is crucial to accurately analyze and handle the available aerodynamic data, whether scalar- or vector-valued quantities. In fact, a correct interpretation of data helps the aerodynamic engineer to gain insights into complex physical phenomena as well as reliably judge new technologies. Moreover, aerodynamic data sets represent the common interface to other disciplines such as flight mechanics, loads analysis or overall aircraft design. Quantity and quality of these data sets highly depend on the methodology used to gather them. While flight tests are known to provide data within a real environment but at a substantial cost, numerical analyses are a cheaper alternative at a reduced fidelity level. The efficient usage, combination and handling of such data sets have always been a focus of the aerodynamics community. However, in the past decade, this trend has significantly intensified due to increasing computational resources available and the rise in popularity of machine learning (ML) methods. In fact, big data, ML and deep learning (DL) are regularly seen as driving factors of the aerospace industry in the upcoming years. Looking at potential benefits and existing demands from several application fields, the German Aerospace Center (DLR), Institute of Aerodynamics and Flow Technology decided to intensify research on ML for applied aerodynamic challenges several years ago. This lecture aims at transferring knowledge gained throughout the years to interest students and colleagues who are new to the field. It is split in two parts. First, Bayesian optimization, also known as surrogate-based optimization (SBO), is introduced and demonstrated for an airfoil optimization. The same methodology is also employed to handle uncertainties. Second, an overview of the challenges and differences of ML for computation fluid dynamics compared to “classical” ML is given.

10.2 Machine Learning for Efficient Design

In commercial aviation, an ongoing need exists to decrease fuel consumption. This imperative arises from the necessity to fulfill strict environmental objectives established by governmental bodies. The European Commission delineates these ecological objectives, stipulating a 75% decrease in CO_2 and a 90% decrease in NO_x emissions per passenger kilometer by 2050 (Publications Office of the European Union, 2011). Given that the tube and wing layout that has already been highly optimized is dominating the market, and seems to do so also for

Preview Sample

upcoming years, improvements are no longer trivial to obtain. Hence, the integration of aerodynamic shape optimization with advanced technologies has become increasingly prevalent. Especially designing for laminar instead of turbulent flow on large areas of an aircraft offers a significant reduction in viscous drag and, hence, directly reduces fuel consumption. Note that while laminarity is not a new concept in general, it remains relatively novel in the context of commercial aircraft wing design. Throughout the years, direct shape optimization has been used in designing airfoils and wings to achieve natural laminar flow (NLF) (Han et al., 2018). For all these ambitions, typically, a large number of solution evaluations, e.g., CFD simulations, is necessary. This is the place where ML comes into play by ensuring intelligent data collection and usage.

10.2.1 Surrogate-based Optimization

Solving a global optimization problem may require a large number of function (black-box) evaluations depending on the design space and the non-linearity of the function. This can rapidly turn intractable and infeasible, e.g., when - (i) the design space is high-dimensional and/or (ii) the function evaluation is expensive (e.g., a typical CFD simulation). Next, an efficient optimization technique is discussed that is known as surrogate-based optimization (SBO) or Bayesian Optimization; see Forrester & Keane (2009) for further details. It leverages machine learning models to reduce computational cost while preserving high accuracy. For a general optimization problem, let $y \in \mathbb{R}$ be the quantity of interest (QoI), which is usually a performance measure such as the drag coefficient, depending (mostly non-linearly) on the design variables $\boldsymbol{x} \in \mathbb{R}^d$ at operating conditions $\boldsymbol{A}$. The goal of optimization is to find an optimal set of design variables $\boldsymbol{x}^*$ at constant operating conditions $\boldsymbol{A}_0$ while satisfying k constraints on the design variables,

$$\begin{aligned} \boldsymbol{x}^* &= \arg\min_x \{y(\boldsymbol{x}, \boldsymbol{A}_0)\}, \\ g_i(\boldsymbol{x}) &\leq 0, \quad i = 1, 2, ..., k. \end{aligned} \tag{10.1}$$

The workflow of SBO is displayed in Figure 10.1 and consists of four steps. The first step contains generating the initial DoE sampling in the design space and evaluating the objective function (and constraints) at all sampling points. Second, a surrogate model for the objective function (and constraints) is constructed and iteratively refined using an active infill criteria to sequentially reach the

Preview Sample

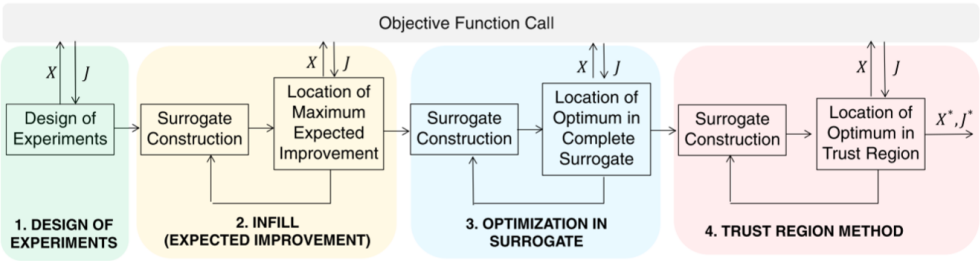


Figure 10.1: Surrogate-based optimization workflow.

optimum while evaluating the black-box solver for each proposed optimal design. It also includes a reconstruction of the surrogate model(s) after every infill iteration. Third, the location of the optimum is found using the surrogate model, which is verified by calling the black-box function evaluation. Finally, a so-called trust region approach can be employed. Instead of finding the optimal solution within the whole design space, a smaller search area, also known as the trust region, is defined and used for optimization. We use the SBO module of Surrogate-Modelling for AeRo-data Toolbox in pYthon (SMARTy) developed by DLR (Bekemeyer et al., 2022) but such optimization frameworks are also available in other toolboxes, including open-source packages.

For this lecture, we use Sobol sequences for the DoE stage (Sobol, 1979) but other sampling techniques are available and commonly used such as Latin Hypercube Sampling (LHS) (McKay et al., 2000) or Halton sequences (Halton, 1960). Especially when the computational cost needed for a single function evaluation is high, it has been found beneficial for real-world applications not to use fully random sampling technique (e.g. LHS), but instead employ sampling techniques that are deterministic. This allows for extending the DoE sampling if one realizes at a later stage that more sampling points are needed. With respect to the surrogate modeling task, different models can be used, starting from simple linear regression models all the way to deep neural networks. Most common, however, are Kriging (Gaussian process regression) models, which are introduced shortly in the following section.

Kriging or Gaussian Process Regression

The goal of the Kriging surrogate model, which is also known as the Gaussian Process regression (GPR) model here, is to efficiently provide a continuous

Preview Sample

representation of how the QoI depends on our design variables. Given a data set $\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid i = 1, \ldots, n\}$, with the n input vectors $\mathbf{x} = [x_1, \ldots, x_m]^\top$ that form the design matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$ and the observed response value y_i, we can consider the standard Bayesian linear regression model with additive Gaussian noise

$$\begin{aligned} f(\mathbf{x}) = \mathbf{x}^\top \boldsymbol{\beta}, \quad y = f(\mathbf{x}) + \varepsilon \\ \varepsilon \sim \mathcal{N}(0, \sigma_n^2) \end{aligned} \tag{10.2}$$

where $\boldsymbol{\beta}$ is a vector of weights of the linear model, f the function value and ε the Gaussian noise. Under the Gaussian noise assumption we can formulate the probability density of the observed response values:

$$p(\mathbf{y}|\mathbf{X}, \boldsymbol{\beta}) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left(-\frac{(y_i - \mathbf{x}_i^\top \boldsymbol{\beta})^2}{2\sigma_n^2}\right) = \mathcal{N}(\mathbf{X}^\top \boldsymbol{\beta}, \sigma_n^2 I) \tag{10.3}$$

with a zero mean Gaussian *prior* for the weights $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \Sigma_p)$ where Σ_p is a covariance matrix. If we introduce a mapping function from input space into feature space $\phi(\mathbf{x})$ and define the covariance function $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^\top \Sigma_p \phi(\mathbf{x})$, the predictive posterior distribution at a test point $\mathbf{x}_*$ then reads (Rasmussen, 2004)

$$\mathbb{E}(f(\mathbf{x}_*)) = k(\mathbf{x}_*)(K + \sigma^2 I)^{-1}\mathbf{y} \ , \tag{10.4}$$

$$\mathrm{Var}(f(\mathbf{x}_*)) = k(\mathbf{x}_*, \mathbf{x}_*) - k(\mathbf{x}_*)^T (K + \sigma^2 I)^{-1} k(\mathbf{x}_*) \tag{10.5}$$

with $k(\mathbf{x}_*)$ as the vector of covariances between the test point and n sampled points and $\boldsymbol{K} = K(X, X)$ denoting the corresponding covariance matrix. With the specification of the covariance function, a distribution over possible functions is implied. Further details on GPRs are provided in Chapter 2.

Adaptive Sampling for Optimization

The infill criteria employed here is *expected improvement* (EI) which is based on the normal distribution prediction (mean $\hat{y}(\boldsymbol{x})$ and standard deviation $\hat{\sigma}(\boldsymbol{x})$) of the objective function obtained from the Kriging surrogate at any given location $\boldsymbol{x}$ in design space. Using the probability of improvement with respect to the current best-sampled solution y_{min}, EI is defined as:

$$EI[(\boldsymbol{x})] = (y_{min} - \hat{y}(\boldsymbol{x}))\, \Phi\left(\frac{y_{min} - \hat{y}(\boldsymbol{x})}{\hat{\sigma}(\boldsymbol{x})}\right) + \hat{\sigma}(\boldsymbol{x})\phi\left(\frac{y_{min} - \hat{y}(\boldsymbol{x})}{\hat{\sigma}(\boldsymbol{x})}\right), \tag{10.6}$$

Preview Sample

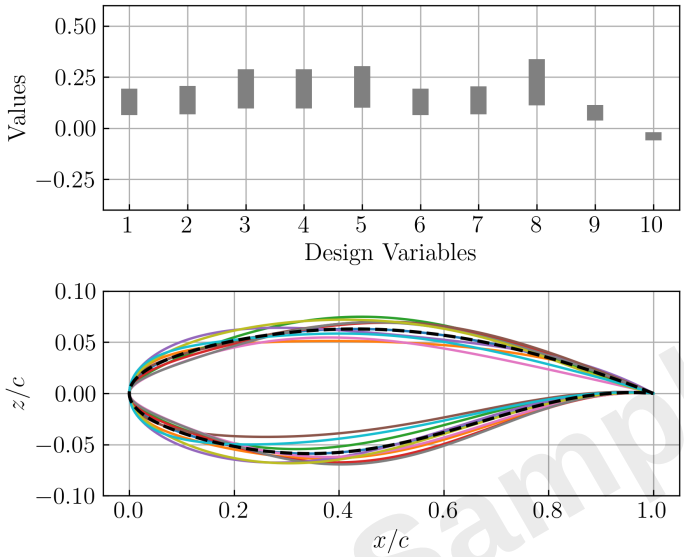


Figure 10.2: CST design variable bounds used for optimization (top). RAE2822 airfoil (dashed) and sample airfoil profiles (colored) in the design space (bottom) (Parekh et al., 2023).

Preview Sample

where Φ and ϕ are the cumulative and probability distribution functions of the standard normal distribution, respectively. The EI infill method balances exploration with exploitation in that a large expected improvement is present in the regions where a solution smaller than the current best is possible and/or regions where the model error is large. The refining continues until the convergence criteria (L_2 distance between consecutive design vectors or the EI values) or the maximum number of infill points (budget) is reached. Note that various other infill criteria have been proposed and applied throughout the years, e.g., minimum or maximum values infill. A more detailed review is available in Forrester & Keane (2009).

10.2.2 An Airfoil Use Case

In every optimization, a crucial and often underestimated aspect is parameterization. It defines how the shape changes with respect to the design parameters $\boldsymbol{x}$. Even though it is technically possible to consider all surface nodes independently,

using a more compact description is generally desired to limit the number of degrees of freedom and, hence, keep computational cost at bay. In this lecture, we use the class shape function transformations (CSTs) to describe and change an airfoil profile over-optimization iterations Kulfan (2008). The parameterization represents a two-dimensional geometry by the product of a class function $C(x/c)$, and a shape function $S(x/c)$ based on Bernstein binomials, plus a term that characterizes the trailing edge thickness:

$$\begin{aligned} \frac{\mathrm{z}}{\mathrm{c}} &= C\left(\frac{\mathrm{x}}{\mathrm{c}}\right) S\left(\frac{\mathrm{x}}{\mathrm{c}}\right) + \frac{\mathrm{x}}{\mathrm{c}} \frac{\Delta z_{TE}}{c} \\ C\left(\frac{\mathrm{x}}{\mathrm{c}}\right) &= \left(\frac{\mathrm{x}}{\mathrm{c}}\right)^{N_1} \left(1 - \frac{\mathrm{x}}{\mathrm{c}}\right)^{N_2} \quad \text{for} \quad 0 \leq \frac{\mathrm{x}}{\mathrm{c}} \leq 1 \\ S\left(\frac{\mathrm{x}}{\mathrm{c}}\right) &= \sum_{i=0}^{n} \left[X_i K_{i,n} \left(\frac{\mathrm{x}}{\mathrm{c}}\right)^{i} \left(1 - \frac{\mathrm{x}}{\mathrm{c}}\right)^{n-i} \right] \end{aligned} \tag{10.7}$$

where $K_{i,n} = \frac{n!}{i!(n-i)!}$. The exponents N_1 and N_2 define the type of geometry to be represented. An airfoil, e.g. is represented by $N_1 = 1/2$ and $N_2 = 1$, adhering to the fact that the term $\sqrt{x/c}$ yields round leading edges, whereas $(1 - x/c)$ results in sharp trailing edges. The weight factors X_i is representing the design variables. The CST parameterization guarantees C2 continuity of the surfaces and captures the space of smooth airfoil shapes. Ten design parameters are used to define this parameterization (five for the upper surface and five for the lower surface). The theoretical bounds of the CST parameters is $[-1, 1]$. However, in order to avoid unrealistic designs, we restrict the design space with respect to the baseline transonic airfoil (RAE2822) shape such that the CST bounds are $\boldsymbol{x}_0(1 \pm 0.3)$, i.e. $\pm 30\%$ of baseline design $\boldsymbol{x}_0$. Figure 10.2 shows the bounds of the design variables and a few sample airfoil profiles around the baseline. Note that larger bounds are also increasing the design freedom and might lead to even better results at the expense of nonphysical shapes. It is the responsibility of the designer to define appropriate bounds for the individual problem at hand. Other options are available besides the herein-used CST parameterization, such as free-form deformation or NURBS.

The flow around the airfoil is modeled using the DLR CFD solver TAU (Schwamborn et al., 2006). The aerodynamic quantities of interest are obtained by solving the Reynolds-averaged Navier-Stokes (RANS) equations in combination with the $k-\omega$ SST turbulence model. The unstructured mesh with 150,000 cells and 1024 surface nodes is shown in Figure 10.3. In this study we account

Preview Sample

for laminar flow and hence have to predict the transition location. Therefore, we employ the recently developed DLR $\gamma-$CAS model (François et al., 2023) that consists of a transport equation for the intermittency γ, an auxiliary field variable taking values between 0 (laminar) and 1 (turbulent) and acting as a switch for the turbulence model. Further details on the transition model can be found in François et al. (2023) while additional details on the optimization study are available in Parekh et al. (2023). Once the surface of the airfoil is changed due to a change in design parameters we employ mesh deformation to obtain a matching volume mesh.

10.2.3 Optimization Results

The surrogate-based optimization is used to solve the deterministic optimization problem

$$\begin{aligned} \boldsymbol{x}^* &= \arg\min_x \{C_D(\boldsymbol{x}, M, C_L)\} \\ \text{s.t.} \quad & t/c_{max} \geq t/c_{max,0} \end{aligned} \tag{10.8}$$

Hence, finding the airfoil shape that minimizes drag while constraining the minimum thickness. Note that, even though we account for laminar flow, we do not directly reflect this in the optimization problem but instead simply aim for the lowest drag value. The idea is that finally we want the most efficient airfoil with respect to drag and not the airfoil with the largest region of laminar flow. If the optimization problem is posed differently also results with differ

Preview Sample

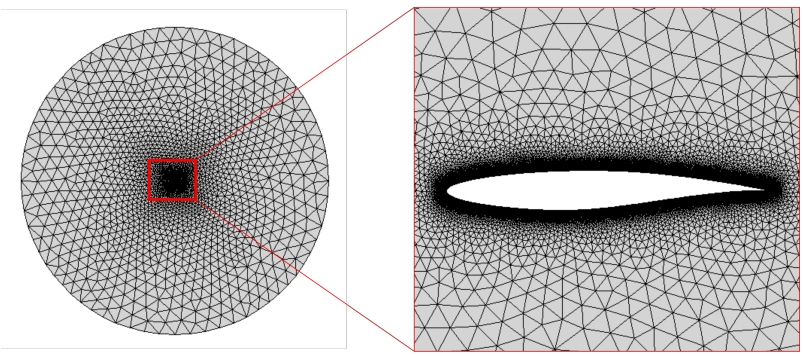


Figure 10.3: CFD grid of the baseline (RAE2822) airfoil used in the optimization problem. Every iteration in the optimization performs a mesh deformation.

Not available in preview

network receives both parameters as additional inputs. The training points are therefore sampled in a four-dimensional domain. Moreover, we introduce a locally varying artificial viscosity factor that is also determined automatically during training from the network. For optimal space filling a non-uniform point distribution is used. Half of the points are distributed uniformly across the entire physical domain $\Omega = (-1, 1) \times (-1, 1)$ using the Halton sequence (Halton, 1960). For the other half of the points, the y-coordinate is sampled using a normal distribution with a variance of $\sigma = 0.07$ and with a uniform distribution for the x-coordinate. The resulting predictions for the Mach number field for three exemplary cases are shown in Figure 10.14. One can see that for the cylindrical shape ($a = 0.1$), even at $M_\infty = 0.4$, close to the critical Mach number, the

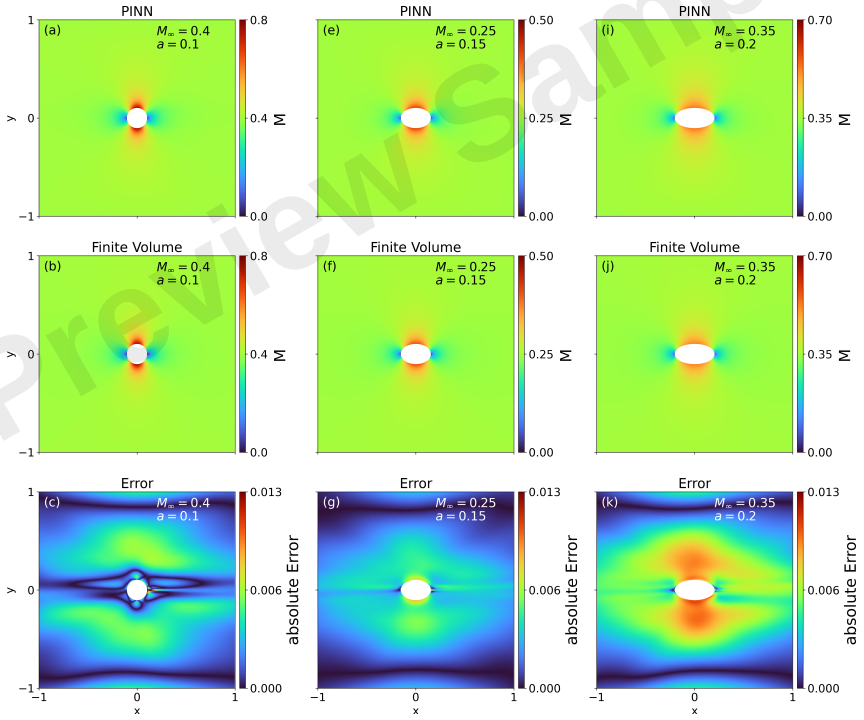


Figure 10.14: Exemplary predictions of parametric PINN for local Mach number for parametric flow around a cylinder. The first row shows the PINN approximation for three different parameter sets. The second row shows a reference solution and the third row shows the absolute error.

Preview Sample

results are visually indistinguishable from the reference solution. The plot of the absolute error reveals that inaccuracies are most prevalent near the cylinder surface and directly up-/down-stream. For the other two parameter sets with ellipsoidal shapes, a similar quality of the results can be observed. Furthermore, slight asymmetries between the upper and lower side can be observed. While this accuracy is inferior to established numerical solvers, the method might still be of value in situations where rapid predicts for various parameter combinations, e.g. optimization or loads analysis, are of interest. Further details including additional studies also on a supersonic test case can be found in Wassing et al. (2024)

10.4 Conclusions

Machine Learning has a tremendous potential to enhance and improve currently established tools as well as workflow within the field of applied aerodynamics. In fact, strength of machine learning models such as rapid turn-around times while accounting for highly non-linear phenomena naturally addresses several shortcomings of traditional methods and approaches. This lecture is split into two parts. The first part focuses on how machine learning models can be used for optimization and uncertainty quantification with an applied aerodynamics example. The second part shed some light on differences between machine learning tasks and settings within the field of fluid dynamics and outside of it. Based on this some research directions are highlighted that account for such differences. Overall machine learning is powerful tool for enhancing aerodynamics but it is not a golden bullet that magically solves all existing problems. Instead it should be employed in a knowledgeable way by practitioners. Moreover, additional research is needed to overcome still existing challenges and advance the state-of-the-art.

Preview Sample

11

Dimensionality reduction, classification and reduced-order modelling for reacting flow simulations

Alessandro Parente*
Université Libre de Bruxelles,
Bruxelles, Belgium

Preview Sample

The use of machine learning algorithms to predict the behaviors of complex systems is gaining interest in the combustion community. The key to an effective use of machine-learning tools in multi-physics problems, including combustion, is to couple them to physical and computer models, to embody in them all the prior knowledge and physical constraints that can enhance their performances, and to improve them based on the feedback coming for the validation experiments. In other words, we need to adapt the scientific method to bring machine learning into the picture and make the best use of the massive amount of data we have produced thanks to the advances in numerical computing. The present talk reviews some of the open opportunities for the application of data-driven, reduced-order modelling of combustion systems. Examples of feature extraction in turbulent combustion data, empirical low dimensional manifold identification, classification, regression and reduced-order modelling are provided.

* alessandro.parente@ulb.be. The present lecture notes are based on the research work performed by several Authors at Université Libre de Bruxelles. In particular, the contribution by Prof. Axel Coussement, Dr. Gianmarco Aversano, Ms. Kamila Zdybał, Mr. Giuseppe D'Alessio, Mr. Rafi Malik, Mr. Alberto Procacci, Ms. Laura Donato is acknowledged. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement No 714605.

11.1 Introduction

The simulation of turbulent combustion is a very challenging task, for a number of aspects beyond turbulence. Indeed, combustion is intrinsically multi-scale and multi-physics. It is characterized by a variety of scales inherently coupled, in space and time, through thermo-chemical and fluid dynamic interactions (Pope, 2013). Typical chemical mechanisms describing the evolution of fuels consist of hundreds of species involved in thousands of reactions, spanning twelve decades of temporal scales (Frassoldati et al., 2003). The interaction of these scales with the fluid dynamic ones defines the nature of the combustion regime as well as the limiting process in the determination of the overall fuel oxidation rate (Kuo & Acharya, 2012). When the characteristic chemical scales are much smaller than the fluid dynamic ones, the combustion problem becomes a mixing one (i.e., mixed is burnt (Magnussen, 1981)): combustion and chemistry are decoupled and the problem is highly simplified. Likewise, for chemical time scales much larger than the fluid dynamic ones, the system can be described taking into account chemistry only, neglecting the role of fluid dynamics altogether. The intensity of the interactions between turbulent mixing and chemistry is measured using the Damkhöler number, defined as the ratio between the characteristic mixing, τ_m, and chemical, τ_c, time scales:

$$Da = \frac{\tau_m}{\tau_c}. \tag{11.1}$$

In terms of Da number, $Da >> 1$ indicates a mixing-controlled, fast chemistry process. On the other hand, $Da << 1$ denotes a chemistry-controlled, slow chemistry process. Most practical combustion systems operate at conditions characterized by a non-negligible overlap between flow and chemical scales. This is particularly true for novel combustion technologies, where the use of diluted conditions and the enhanced mixing leads to a Da distribution close to unity. This grants some control on the combustion process, thanks to the increase of the characteristic chemical scales compared to the mixing ones. In particular, the operating conditions (temperature and compositions) can be adjusted in such a way that the emissions are kept below the required values (Cavaliere & de Joannon, 2004; J.Wünning & J.Wünning, 1997; Parente et al., 2011). The condition $Da \approx 1$ is generally referred to as finite-rate chemistry, to indicate that combustion is not infinitely fast but of finite speed. Modelling these combustion regimes is very challenging because both fluid mechanics and chemistry must be accurately modeled. In particular, chemistry cannot be described using

Preview Sample

simplified global mechanisms, resulting in a significant burden for combustion simulations. Indeed, the solution of turbulent combustion problems requires the solution of hundreds of transport equations for (tightly coupled) chemical species on top of the conservation equations for mass, momentum and energy, and the corresponding closure models (i.e., turbulence models). Beside the dimensionality problem, the transport equations of reacting scalars also require closure models, when the reacting structures are not fully resolved on the numerical grid. The challenges associated to turbulent combustion modelling makes the use of machine learning very attractive. While turbulent combustion models are spread across combustion industries, their current predictive capabilities fall well short of what would be needed in decision making for new designs and regulations (Pope, 2013). High-fidelity, direct numerical simulations (DNS) of combustion systems are still limited to particular aspect of a turbulent combustion process and simple 'building blocks'. Still, these high-fidelity simulations are rich in information that could help decode the complexity of turbulence-chemistry interactions and guide the development of filtered and lower-fidelity modelling approaches for faster evaluations. The objective of the present lecture is to demonstrate the potential of data-driven modelling in the context of combustion simulations. In particular, we present:

- The application of Principal Component Analysis (PCA) and other linear and non-linear techniques to identify low-dimensional manifolds in high-fidelity combustion data-sets and reveal the key features of complex non-equilibrium phenomena. Different techniques are compared to PCA, including Non-negative Matrix Factorization (NMF), Autoencoders and Local PCA in Section 11.2.

- The development of reduced-order models (ROMs), to be used in conjunction with, or to replace high-fidelity simulation tools, to reduce the burden associated with the large number of species in detailed chemical mechanisms. First, the use of transport models based on PCA are presented in Section 11.3. Then, the application of a dynamic adaptive chemistry approach based on the combination of classification and non-linear regression is discussed in Section 11.4. Finally, the development of digital twins of combustion systems is shown in Section 11.5.

Preview Sample

11.2 Feature extraction using dimensionality reduction techniques

Dimensionality reduction techniques offer a way to represent data sets in a new, lower-dimensional basis. Such data representations are referred to as *manifolds*. Techniques such as PCA, NMF or Autoencoders can additionally extract meaningful features by exploiting the fact that the new basis can be better suited to represent certain physical phenomena underlying the original data. Compressed data representations on the low-dimensional manifolds are thus often described by features whose aim is to carry as much information in as few components as possible.

11.2.1 Combustion data sets

In reactive flow applications, the original data set, stored in a form of matrix $\mathbf{X} \in \mathbb{R}^{n_{obs} \times Q}$, is assumed to have n_{obs} rows corresponding to points on a spatial grid and Q columns of variables representing the thermo-chemical state-space: temperature, pressure and $N_s - 1$ species mass or mole fractions.* For open flames, the pressure variable can be neglected.

In the context of techniques presented here, it is additionally useful to define *mixture fraction*. It is an important scalar quantity that specifies the local fuel to oxidizer ratio and is defined as:

$$Z = \frac{m_f}{m_o + m_f}, \tag{11.2}$$

for any point on a spatial grid, where m_f is the mass of fuel and m_o is the mass of oxidizer in a mixture. This variable is of particular interest in diffusion flames characterized by two separate feeds: one for fuel and one for oxidizer, which then mix (diffuse) one into the other. The combustion of fuel happens at the stoichiometric mixture fraction Z_{st}. If $Z < Z_{st}$, the mixture is called lean (fuel deficient) and if $Z > Z_{st}$ the mixture is called fuel rich.

Data pre-processing

Pre-processing the data set prior to applying a dimensionality reduction technique can have a significant impact on the shape of low-dimensional manifolds

* since mass or mole fractions sum up to 1.0 for every observation, out of N_s species only $N_s - 1$ are independent.

Preview Sample

and subsequently on the quality of features retrieved (Parente & Sutherland, 2013b; Peerenboom et al., 2015; Zdybał et al., 2022). Most commonly, the data set $\mathbf{X}$ is first centered and scaled:

$$\mathbf{X}' = (\mathbf{X} - \bar{\mathbf{X}})\mathbf{D}^{-1}. \tag{11.3}$$

Here, $\bar{\mathbf{X}}$ contains mean observations of each variable and $\mathbf{D}$ is a diagonal matrix of scales. Other pre-processing means may include outlier detection for experimentally obtained data sets. For the rest of this discussion we assume that $\mathbf{X}$ represents matrix that has already been pre-processed.

Preview Sample

Description of the analyzed data set

In the sections to follow, results are shown from applying dimensionality reduction techniques to a numerical data set from a DNS simulation of a CO/H_2 turbulent jet (Sutherland et al., 2007) using a mechanism consisting of twelve species and 33 reactions (Yetter et al., 1991). Additional information regarding the hydrodynamics of the numerical simulation can be found in Sutherland et al. (2007). The DNS data consists of a single 2D slice extracted from the 3D domain and is arranged in a matrix $\mathbf{X}$ made up of approximately 1.5 million rows. The matrix $\mathbf{X}$ has thirteen columns corresponding to temperature and twelve species mass fractions incorporated in the chemical mechanism. The analysis presented here can nevertheless be carried over to other reactive flow data sets.

11.2.2 Theory

Principal Component Analysis

PCA (Jolliffe, 2002) projects the original data set $\mathbf{X}$ onto a new basis represented by the Principal Components (PCs). The matrix $\mathbf{A} \in \mathbb{R}^{Q \times Q}$, whose Q columns are composed of orthogonal PCs is obtained from the eigendecomposition of a covariance matrix associated with the data set. Matrix $\mathbf{X}$ can then be projected onto the PCs to obtain the scores $\mathbf{Z} \in \mathbb{R}^{n_{obs} \times Q}$, which are essentially original observations represented in a new coordinate system defined by the PCs:

$$\mathbf{Z} = \mathbf{X}\mathbf{A}. \tag{11.4}$$

By keeping a reduced number of $q < Q$ first PCs, we obtain the closest* rank-q approximation of the original matrix:

$$\mathbf{X} \approx \mathbf{X_q} = \mathbf{Z_q} \mathbf{A_q}^T, \tag{11.5}$$

where the subscript $\mathbf{q}$ denotes a truncation from Q to q components.

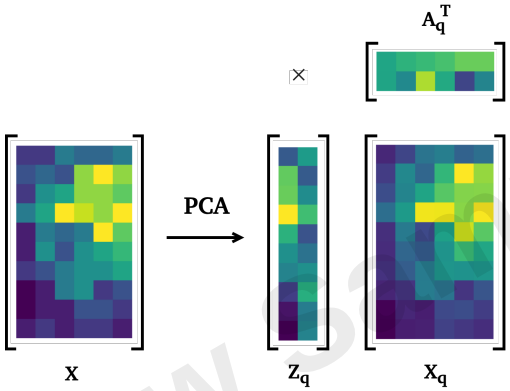


Figure 11.1: Approximation of the original data matrix $\mathbf{X}$ using a reduced set of q PCs.

Figure 11.1 shows in a simplified graphical way the low-rank approximation according to Eq. 11.3.

Preview Sample

Local Principal Component Analysis

PCA can also be applied locally (Local-PCA, LPCA) (Kambhatla & Leen, 1997), on portions of the original data set. A partitioning algorithm is first applied to the data set which results in assigning each observation to a particular cluster. Next, PCA is applied separately on each cluster's observations. Figure 11.2 shows a schematic difference between global and local approach.

LPCA can not only allow for further reduction in dimensionality by adjusting to the nonlinearities of the data set (there are separate PC-bases for each region) but it can also aid in the feature extraction process. PCs found on local portions of data can better represent the physics of a particular cluster.

* in terms of L_2, Frobenius or trace norm.

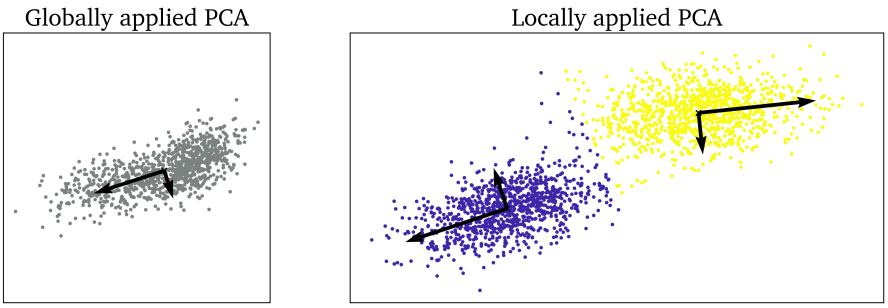


Figure 11.2: Global versus local PCA on a schematic two-dimensional data set. The arrows represent the PCs.

In LPCA, the ith observation is reconstructed from the local, q-dimensional manifold in an analogous way to Eq. 11.3:

$$\mathbf{x}_i \approx \mathbf{x}_{i,q} = \mathbf{z}_{i,q}\mathbf{A}_q^{(k)\,T} + \mathbf{r}^{(k)}. \tag{11.6}$$

Here, k is the index of a cluster to which the ith observation belongs and $\mathbf{r}^{(k)}$ is the centroid of that k^{th} cluster.

Non-negative Matrix Factorization

NMF is an algorithm that focuses on the analysis of data matrices whose elements are non-negative. Given a non-negative data matrix $\mathbf{X} \in \mathbb{R}^{n_{obs} \times Q}$, NMF aims to find two non-negative matrices $\mathbf{U} \in \mathbb{R}^{n_{obs} \times q}$ and $\mathbf{V} \in \mathbb{R}^{Q \times q}$ which minimize the following objective function:

$$O = ||\mathbf{X} - \mathbf{U}\mathbf{V}^T||. \tag{11.7}$$

Thus, NMF essentially finds a compressed approximation of the original data matrix:

$$\mathbf{X} \approx \mathbf{X_q} = \mathbf{U}\mathbf{V}^T. \tag{11.8}$$

Matrix $\mathbf{V}$ of non-negative factors can be regarded as the one containing a basis, analogous to matrix $\mathbf{A}$ found by PCA. Matrix $\mathbf{U}$ represents the compressed data, namely the NMF scores and is thus analogous to the PC scores $\mathbf{Z}$.

Preview Sample

Autoencoders

The Autoencoder is a type of unsupervised Artificial Neural Network (ANN). The simplest form of an Autoencoder is a feedforward neural network having an input layer and an output layer with the same number of neurons, and one or more hidden layers. Its aim is to learn a representation (*encoding*/*decoding*) of a data set such that an error between the input and the output layer is minimized.

Given one hidden layer, the encoding process takes as an input $\mathbf{x} \in \mathbb{R}^Q$ and maps it to $\mathbf{h} \in \mathbb{R}^q$, with $q < Q$:

$$\mathbf{h} = f(\mathbf{W}\mathbf{x} + \mathbf{b}). \tag{11.9}$$

where $\mathbf{h}$ is usually referred to as *code* or latent representation. Here, f is an activation function such as sigmoid or Rectified Linear Unit (ReLU), $\mathbf{W} \in \mathbb{R}^{q \times Q}$ is a weight matrix and $\mathbf{b} \in \mathbb{R}^q$ is a bias vector. The decoding stage, on the other hand, maps $\mathbf{h}$ to the reconstruction $\mathbf{x}'$:

$$\mathbf{x} \approx \mathbf{x}' = f'(\mathbf{W}'\mathbf{h} + \mathbf{b}'). \tag{11.10}$$

where f', $\mathbf{W}'$ and $\mathbf{b}'$ may be unrelated to f, $\mathbf{W}$ and $\mathbf{b}$.

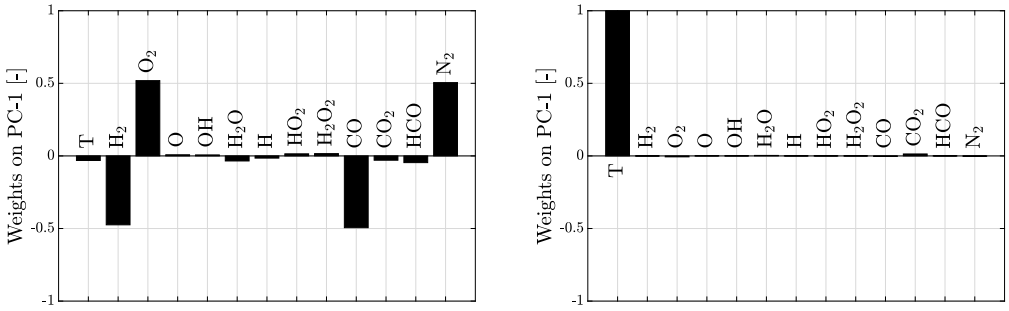
11.2.3 Results

Principal Component Analysis

PCs are ordered in a way that the first PC carry more variance present in the data set than the second PC, and so on. Thus, the first few PCs are typically the ones carrying most of the information. The structure of the PCs changes significantly for different scalings $\mathbf{D}$ of the original data. For instance, using Range scaling, where each variable is divided by the difference between maximum and minimum observation, the first PC represents the mixture fraction variable. It is worth noting here that mixture fraction was not among the variables in the original data set and yet PCA identifies it as a leading direction.

Figure 11.3a shows the weights on that first PC, with fuel stream composed of H_2 and CO having the most negative weights and the oxidizer stream O_2 and N_2 having the highest positive weights. One can compare Figure 11.3a with the first PC resulting from Pareto scaling of the data (observations divided by the square root of their standard deviation) in Figure 11.3b. PCA on Pareto-scaled data finds the larges variance in temperature variable and subsequently this

Preview Sample



(a) Range scaling (b) Pareto scaling

Figure 11.3: The first PC found by PCA on data set pre-processed with two different scaling options.

Preview Sample

PC carry almost none of the information about mass fraction of the chemical species.

Local PCA was also investigated on the analyzed DNS data set. The success of LPCA method in extracting features depends additionally (apart from the aforementioned pre-processing) on the clustering technique used. One of the many possible ways of partitioning combustion data sets is to use a conditioning variable such as mixture fraction. Observations are first split into *fuel-lean* and *fuel-rich* portions at the stoichiometric mixture fraction, as presented in Figure 11.4a. Lean and rich sides can then be further divided if more clusters are desired. Another partitioning algorithm shown in Figure 11.4b is Vector Quantization PCA (VQPCA), in which observations are assigned to the cluster in which the reconstruction error from PCA approximation is minimized.

Other machine-learning clustering techniques, such as K-Means clustering, are also worth mentioning.

For the DNS data set analyzed here, an example partitioning of the spatial observations into $k = 4$ clusters is shown in Figure 11.5. On the left we can observe a temperature profile on a spatial grid and on the right the classification of different regions of the flame to four local clusters. The VQPCA method could distinguish between the oxidizer layer (cluster k_1), the fuel jet layer (cluster k_3) and the reactive layers where the temperature and reaction rates are the highest (clusters k_2 and k_4). We come back to the interpretation of PCs found in those clusters in Section 11.2.3.

Not available in preview

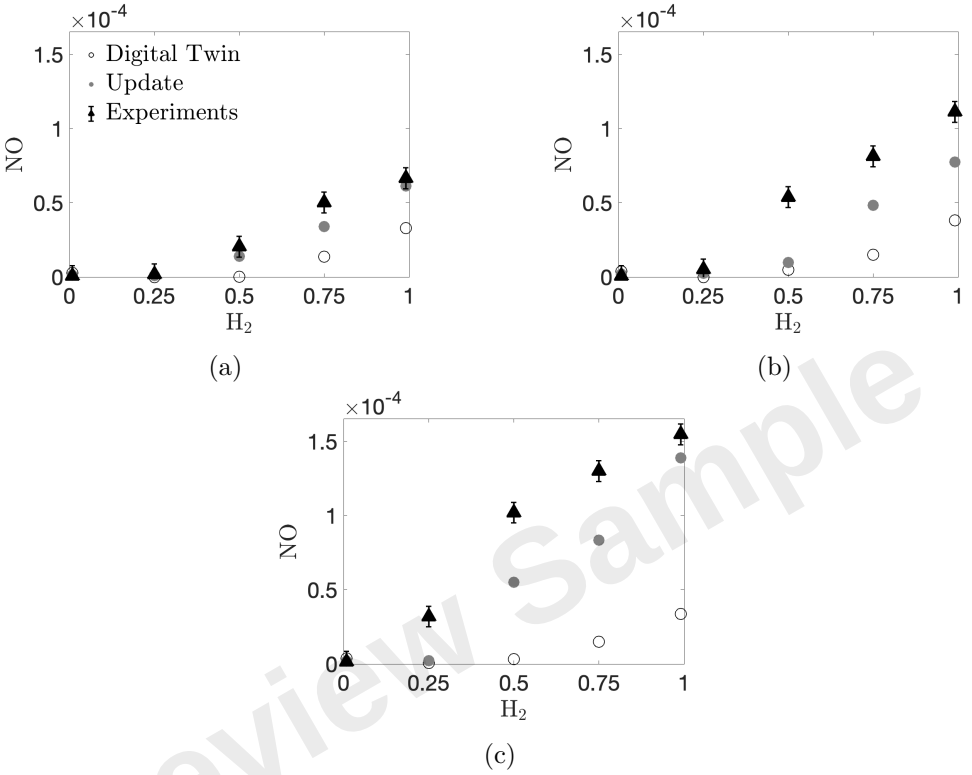
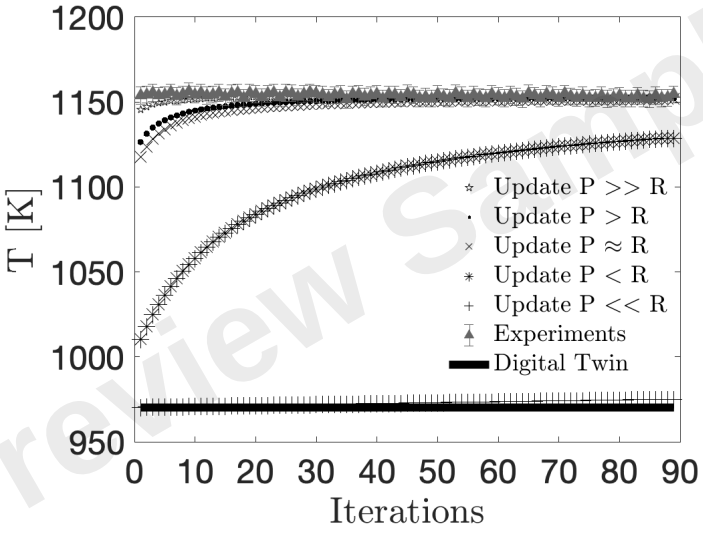


Figure 11.32: NO emissions by the DT, measured through experiments, and updated by the KF, at the outlet of the furnace, varying H_2 mol fraction for ϕ = 0.8 and ID = (a) 16, (b) 20, and (c) 25 mm.

in other words, $\mathbf{y}_k$, is trusted less than the $\mathbf{x}_k$. On the other hand, if $\mathbf{P} >> \mathbf{R}\ \mathbf{K}_k$, weights more heavily the $\mathbf{R}_k$, in other words, $\mathbf{y}_k$, is given more "trust" compared to the predicted $\mathbf{x}_k$.

This figure demonstrates the important role of the uncertainties and, in particular, the Kalman gain as it is the weighted factor through which all the DA process is based on. These scenarios exhibit how DA integrates information and uncertainties from both the model and the measurements to produce more accurate state variables compared to relying on either one alone.

Preview Sample



Preview Sample

Figure 11.33: Sensitivity analysis of the model and experiments uncertainties for the assimilated temperature value at z = 100 mm, for the case $\phi = 0.8$, ID = 16 and H_2 = 50 %.

12

Digital Twins: Challenges and Enablers

Omer San*

University of Tennessee, Knoxville, US

This chapter presents a brief overview of digital twin technology, focusing specifically on its applications relevant to fluid dynamics. The term digital twin commonly denotes a virtual representation mirroring physical assets or processes. However, it is crucial to acknowledge that the definition of a digital twin may vary across applications and institutions. Indeed, discerning what does not constitute a digital twin may provide valuable insights in our increasingly digitized world. Therefore, this chapter endeavors to elucidate the hierarchical levels of digital twin capabilities, starting with an exposition on taxonomy and terminology. Additionally, we seek to illuminate challenges and facilitators inherent in deploying digital twin technology. Given the historical precedence of fluid dynamics as a vanguard in adopting innovative design, modeling, and simulation paradigms, our overarching goal is to dispel any ambiguities that may arise among practitioners in the field when navigating the landscape of digital twin utilization.

Preview Sample

12.1 Digital Twin: Taxonomy and Terminology

In recent years, *digital twin* has become a loaded term, not only in academia but also in industry, since it touches on literally any asset management tasks beyond engineering and technology. In earlier work, we presented a digital twin perspective from a modeling point of view (Rasheed et al., 2020), and later expanded our discussion including various industry perspectives (Stadtmann et al., 2023). In those papers, various definitions of a digital twin have been provided. Without listing many definitions here, we can perhaps enlist two of them as follows:

* osan@utk.edu.

- A digital twin is a virtual representation of physical assets or processes, empowered by data and simulators, enabling real-time forecasting, optimization, monitoring, control, and informed decision-making (Rasheed et al., 2020; Stadtmann et al., 2023).
- A digital twin is a set of models that evolve over time to persistently represent the structure, behavior, and context of a unique physical system or process (AIAA Digital Engineering Integration Committee, 2020; Kapteyn et al., 2021).

Overall, these digital twins interact with physical assets in two ways: by controlling real assets and predicting their future states and by calibrating models using data from the physical asset. While the roots of this concept can be traced back to the Apollo 13 mission (Allen, 2021), it is nowadays considered an essential enabling technology that will allow decades of investment in computational science and engineering to reach their full potential. According to the Gartner, McKinsey, and PwC roadmaps, the digital twin concept has gained a multibillion-dollar market capitalization value for years to come since it involves stakeholders from the asset's creation through decommissioning stages (Rasheed et al., 2020; Stadtmann et al., 2023).

Over the last few decades, many fields have developed detailed models and simulation tools, especially for complex systems, e.g., the computational fluid dynamics packages and multiphysics software infrastructure. At the same time, there have been huge advances in sensing, data acquisition, and measurement tools, the Internet of Things, chip technologies, visualization and augmented reality concepts, cyberinfrastructure, computing infrastructure, and cloud and edge computing. Those developments pushed the forward simulation tools into optimization, uncertainty quantification, and broader inverse problem tools (Ghattas & Willcox, 2021). Therefore, there has become a re-emphasis on data assimilation and control disciplines in broader science and engineering disciplines. Without isolation from the rest of the world, the field of fluid dynamics has also witnessed an explosion of cross ideas in a rapidly evolving and expanding digitized world (Brunton & Noack, 2015; Taira et al., 2017; Brunton et al., 2020; Karniadakis et al., 2021).

Therefore, digital twin technology should perhaps be viewed as an integrative and collaborative approach to advancing and developing many disciplines on various fronts. When defining a digital twin, expanding the view of a broader representative notion of an asset or process is crucial. Perhaps that representation

Preview Sample

Not available in preview

designed for diagnostic, predictive, prescriptive, or autonomous purposes, these digital twins signify a progression toward more human-in-the-loop systems. In this paradigm, human attention is directed towards critical decision-making, while automated processes take on repetitive, dull, dirty, and dangerous tasks, optimizing the utilization of AI technologies.

Preview Sample

Preview Sample

Bibliography

A. Sandu, T. C. (2011). Chemical data assimilation, an overview. *Atmosphere*.

Abadía-Heredia, R., López-Martín, M., Carro, B., Arribas, J., Perez, J., & Le Clainche, S. (2022). A predictive hybrid reduced order model based on proper orthogonal decomposition combined with deep learning architectures. *Expert Syst. Appl.*, 187, 115910.

Abergel, F. & Temam, R. (1990). On some control problems in fluid mechanics. *Theoretical and Computational Fluid Dynamics*, 1(6), 303–325.

Abu-Mostafa, Y. S., Magndon-Ismail, M., & Lin, H.-T. (2012). *Learning from Data. A Short Course*. AMLBook.

Agarwal, K., Ram, O., Wang, J., Lu, Y., & Katz, J. (2021). Reconstructing velocity and pressure from noisy sparse particle tracks using constrained cost minimization. *Experiments in Fluids*, 62(4), 75.

Agostini, L. (2020). Exploration and prediction of fluid dynamical systems using auto-encoder technology. *Physics of Fluids*, 32(6), 067103.

Ahmed, S. E., Bhar, K., San, O., & Rasheed, A. (2020a). Forward sensitivity approach for estimating eddy viscosity closures in nonlinear model reduction. *Physical Review E*, 102(4), 043302.

Ahmed, S. E., Pawar, S., & San, O. (2020b). PyDA: A hands-on introduction to dynamical data assimilation with python. *Fluids*, 5(4), 225.

Ahmed, S. E., Pawar, S., San, O., Rasheed, A., Iliescu, T., & Noack, B. R. (2021a). On closures for reduced order models – a spectrum of first-principle to machine-learned avenues. *arXiv preprint arXiv:2106.14954*, 33(9), 091301.

Ahmed, S. E., San, O., Kara, K., Younis, R., & Rasheed, A. (2021b). Multifidelity computing for coupling full and reduced order models. *PLoS ONE*, 16(2), e0246092.

Ahmed, S. E., San, O., Rasheed, A., & Iliescu, T. (2020c). A long short-term memory embedding for hybrid uplifted reduced order models. *Physica D: Nonlinear Phenomena*, 409, 132471.

Ahmed, S. E., San, O., Rasheed, A., Iliescu, T., & Veneziani, A. (2022). Physics guided machine learning for variational multiscale reduced order modeling. *arXiv:2205.12419*.

AIAA Digital Engineering Integration Committee (2020). Digital twin: Definition & value. *AIAA and AIA Position Paper*.

Alcaraz, C. & Lopez, J. (2020). Secure interoperability in cyber-physical systems. In *Cyber Warfare and Terrorism: Concepts, Methodologies, Tools, and Applications* (pp. 521–542). IGI Global.

Alcaraz, C. & Lopez, J. (2022). Digital twin: A comprehensive survey of security threats. *IEEE Communications Surveys & Tutorials*, 24(3), 1475–1503.

Alexandrov, N. M. (2021). Computational models of trustworthiness and trust in autonomous cyber-physical-human systems. In *AIAA Science and Technology Forum and Exposition (2021 AIAA SciTech Forum)*.

Preview Sample

Allen, B. D. (2021). Digital twins and living models at NASA. In *Digital Twin Summit*.

Amarloo, A., Cinnella, P., Iosifidis, A., Forooghi, P., & Abkar, M. (2023). Data-driven reynolds stress models based on the frozen treatment of reynolds stress tensor and reynolds force vector. *Physics of Fluids*, 35(7).

Amarloo, A., Forooghi, P., & Abkar, M. (2022). Frozen propagation of Reynolds force vector from high-fidelity data into Reynolds-averaged simulations of secondary flows. *Phys. Fluids*, 34(11), 115102.

Andersen, R. (2007). *Modern Methods for Robust Regression (Quantitative Applications in the Social Sciences)*. SAGE Publications, Inc;.

Anil, R., Gupta, V., Koren, T., Regan, K., & Singer, Y. (2020). Second order optimization made practical. *arxiv*.

Arbabi, H., Korda, M., & Mezić, I. (2018). A data-driven koopman model predictive control framework for nonlinear partial differential equations. In *2018 IEEE Conference on Decision and Control (CDC)* (pp. 6409–6414).: IEEE.

Arnst, M., Ghanem, R., & Soize, C. (2010). Identification of bayesian posteriors for coefficients of chaos expansions. *Journal of Computational Physics*, 229(9), 3134–3154.

Arora, R., da Silva, B. C., & Moss, E. (2022). Model-based reinforcement learning with SINDy. *arXiv preprint arXiv:2208.14501*.

Asch, M., Bocquet, M., & Nodet, M. (2016). *Data Assimilation*. Society for Industrial and Applied Mathematics.

Association, I. R. E. (2019). Future of wind: Deployment, investment, technology, grid integration and socio-economic aspects.

Åström, K. J. & Wittenmark, B. (2008). *Adaptive control*. Courier Corporation.

Avdonin, A. & Polifke, W. (2018). Quantification of the impact of uncertainties in operating conditions on the flame transfer function with non-intrusive polynomial chaos expansion. In *ASME Turbo Expo: Power for Land, Sea, and Air, Volume 4A: Combustion, Fuels, and Emissions* (pp. V04at04a030).: Asme. Paper GT2018-75476.

Aversano, G., Bellemans, A., Li, Z., Coussement, A., Gicquel, O., & Parente, A. (2019). Application of reduced-order models based on PCA & Kriging for the development of digital twins of reacting flow applications. *Computers and Chemical Engineering*, 121, 422–441.

Aversano, G., Ferrarotti, M., & Parente, A. (2021). Digital twin of a combustion furnace operating in flameless conditions: reduced-order model development from cfd simulations. *Proceedings of the Combustion Institute*, 38(4), 5373–5381.

Bae, H. J. & Koumoutsakos, P. (2022). Scientific multi-agent reinforcement learning for wall-models of turbulent flows. *Nature Communications*, 13(1), 1443.

Bagheri, S., Brandt, L., & Henningson, D. (2009a). Input-output analysis, model reduction and control of the flat-plate boundary layer. *Journal of Fluid Mechanics*, 620, 263–298.

Preview Sample

Bagheri, S., Hoepffner, J., Schmid, P. J., & Henningson, D. S. (2009b). Input-output analysis and control design applied to a linear model of spatially developing flows. *Applied Mechanics Reviews*, 62(2), 020803–1–020803–27.

Bai, Z. (2002). Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems. *Applied Numerical Mathematics*, 43(1-2), 9–44.

Bai, Z., Brunton, S. L., Brunton, B. W., Kutz, J. N., Kaiser, E., Spohn, A., & Noack, B. R. (2017). Data-driven methods in fluid dynamics: Sparse classification from experimental data. In *Whither Turbulence and Big Data in the 21st Century* (pp. 323–342). Springer.

Bai, Z., Wimalajeewa, T., Berger, Z., Wang, G., Glauser, M., & Varshney, P. K. (2014). Low-dimensional approach for reconstruction of airfoil data via compressive sensing. *AIAA Journal*, 53(4), 920–933.

Bakir, G., Schölkopf, B., & Weston, J. (2007). On the pre-image problem in kernel methods. In *Kernel Methods in Bioengineering, Signal and Image Processing* (pp. 284–302). IGI Global.

Bakır, G. H., Weston, J., & Schölkopf, B. (2003). Learning to find pre-images. In *Proceedings of the 16th International Conference on Neural Information Processing Systems*, Nips'03 (pp. 449–456). Cambridge, MA, USA: MIT Press.

Bakır, G. H., Zien, A., & Tsuda, K. (2004). Learning to find graph pre-images. In *Lecture Notes in Computer Science* (pp. 253–261). Springer Berlin Heidelberg.

Balajewicz, M. J., Dowell, E. H., & Noack, B. R. (2013). Low-dimensional modelling of high-reynolds-number shear flows incorporating constraints from the navier–stokes equation. *Journal of Fluid Mechanics*, 729, 285–308.

Baldwin, B. & Lomax, H. (1978). Thin layer approximation and algebraic model for separated turbulent flows. AIAA Paper 78-257.

Banerjee, I. & Ierapetritou, M. (2006). An adaptive reduction scheme to model reactive flow. *Combustion and Flame*, 144(3), 619–633.

Banerjee, S., Krahl, R., Durst, F., & Zenger, C. (2007). Presentation of anisotropy properties of turbulence, invariants versus eigenvalue approaches. *Journal of Turbulence*, 8(32), N32.

Banzhaf, W., Nordin, P., Keller, R. E., & Francone, R. D. (1998). *Genetic programming: an introduction*. Morgan Kaufmann San Francisco.

Bar-Sinai, Y., Hoyer, S., Hickey, J., & Brenner, M. P. (2019). Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31), 15344–15349.

Baraniuk, R. G. (2007). Compressive sensing. *IEEE Sig. Proc. Mag.*, 24(4), 118–120.

Barkley, D. & Henderson, R. D. (1996). Three-dimensional floquet stability analysis of the wake of a circular cylinder. *J. Fluid Mech.*, 322, 215–241.

Barlow, R. S. & Frank, J. H. (1998). Effects of turbulence on species mass fractions in methane/air jet flames. *Proc. Combust. Inst.*, 27, 1087–1095.

Preview Sample

Barrault, M., Maday, Y., Nguyen, N. C., & Patera, A. T. (2004). An empirical interpolation method: Application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 339, 667–672.

Barthelmie, R. J. & Jensen, L. (2010). Evaluation of wind farm efficiency and wind turbine wakes at the Nysted offshore wind farm. *Wind Energy*, 13(6), 573–586.

Basir, S. & Senocak, I. (2022). Physics and equality constrained artificial neural networks: Application to forward and inverse problems with multi-fidelity data fusion. *Journal of Computational Physics*, 463, 111301.

Basir, S. & Senocak, I. (2023). An adaptive augmented lagrangian method for training physics and equality constrained artificial neural networks. *arxiv*.

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., et al. (2018). Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*.

Batzner, S., Smidt, T. E., Sun, L., Mailoa, J. P., Kornbluth, M., Molinari, N., & Kozinsky, B. (2021). Se (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *arXiv preprint arXiv:2101.03164*.

Beck, A. & Kurz, M. (2021). A perspective on machine learning methods in turbulence modeling. *GAMM-Mitteilungen*, 44(1).

Beckers, J. & Rixen, M. (2003). Eof calculations and data filling from incomplete oceanographic datasets. *J. Atmos. and Ocean. Tech.*, 20(12), 1839–1856.

Beetham, S. & Capecelatro, J. (2020). Formulating turbulence closures using sparse regression with embedded form invariance. *Physical Review Fluids*, 5(8), 084611.

Beetham, S., Fox, R. O., & Capecelatro, J. (2021). Sparse identification of multiphase turbulence closures for coupled fluid–particle flows. *Journal of Fluid Mechanics*, 914.

Begiashvili, B., Groun, N., Garicano-Mena, J., Le Clainche, S., & Valero, E. (2023a). Data-driven modal decomposition methods as feature detection techniques for flow problems: A critical assessment. *Phys. Fluids*, 35, 041301.

Begiashvili, B., Groun, N., Garicano-Mena, J., Le Clainche, S., & Valero, E. (2023b). Data-driven modal decomposition methods as feature detection techniques for flow problems: a critical assessment. *Phys. Fluids*, 35, 041301.

Bekemeyer, P., Barklage, A., Hines Chaves, D. A., & Stradtner, M. (2024). Recent advances in data-driven modeling for aerodynamic applications using dlr's smarty toolbox. In *AIAA SciTech 2024*.

Bekemeyer, P., Bertram, A., Hines Chaves, D. A., Dias Ribeiro, M., Garbo, A., Kiener, A., Sabater, C., Stradtner, M., Wassing, S., Widhalm, M., et al. (2022). Data-driven aerodynamic modeling using the dlr smarty toolbox. In *AIAA AVIATION 2022 Forum* (pp. 3899).

Bellman, R. (1957). A Markovian decision process. *Journal of Mathematics and Mechanics*, (pp. 679–684).

Preview Sample

Bellman, R. (1958). Dynamic programming and stochastic control processes. *Inf. Control.*, 1(3), 228–239.

Bemporad, A. & Morari, M. (2007). Robust model predictive control: A survey. In *Robustness in identification and control* (pp. 207–226). Springer.

Benner, P., Ed. (2020). *Snapshot-based methods and algorithms*. Berlin, Germany: De Gruyter.

Benner, P., Gugercin, S., & Willcox, K. (2015). A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM review*, 57(4), 483–531.

Bennetin, G., Galgani, L., Giorgilli, A., & Strelcyn, J.-M. (1980). Lyapunov characteristic exponents for smooth dynamical systems and for hamiltonian systems: A method for computing all of them. *Meccanica*, 15(9), 27.

Bentaleb, Y., Lardeau, S., & Leschziner, M. A. (2012). Large-eddy simulation of turbulent boundary layer separation from a rounded step. *Journal of Turbulence*, 13, N4.

Benzi, R. & Toschi, F. (2023). Lectures on turbulence. *Physics Reports*, 1021, 1–106.

Berkooz, G., Holmes, P., & Lumley, J. L. (1993). The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25(1), 539–575.

Berrone, S. & Oberto, D. (2022). An invariances-preserving vector basis neural network for the closure of Reynolds-averaged Navier–Stokes equations by the divergence of the Reynolds stress tensor. *Phys. Fluids*, 34(9), 095136.

Berry, T., Giannakis, D., & Harlim, J. (2020). Bridging data science and dynamical systems theory. *arXiv preprint arXiv:2002.07928*.

Bertino, L., Evensen, G., & Wackernagel, H. (2003). Sequential data assimilation techniques in oceanography. *International Statistical Review*, 71(2), 223–241.

Bertsekas, D. (2011). Approximate policy iteration: a survey and some new methods. *J. Control Theory Appl.*, 9, 310–335.

Bertsekas, D. & Tsitsiklis, J. N. (1996). *Neuro-dynamic programming*. Athena Scientific.

Bertsekas, D. P. (1995). *Dynamic programming and optimal control*. Athena scientific Belmont, MA.

Bertsekas, D. P. (2019). *Reinforcement learning and optimal control*. Athena Scientific.

Bessaih, H., Olson, E., & Titi, E. S. (2015). Continuous data assimilation with stochastically noisy data. *Nonlinearity*, 28(3), 729–753.

Bewley, T. & Liu, S. (1998). Optimal and robust control and estimation of linear paths to transition. *J. Fluid Mech.*, 365, 305–349.

Bewley, T., Luchini, P., & Pralits, J. (2016). Methods for solution of large optimal control problems that bypass open-loop model reduction. *Meccanica*, 51(12), 2997–3014.

Bewley, T. R., Moin, P., & Temam, R. (2001). DNS-based predictive control of turbulence: an optimal benchmark for feedback algorithms. *Journal of Fluid Mechanics*, 447, 179–225.

Preview Sample

Bewley, T. R., Temam, R., & Ziane, M. (2000). A general framework for robust control in fluid mechanics. *Physica D*, 138, 360–392.

Bianchi, F. M., Maiorino, E., Kampffmeyer, M. C., Rizzi, A., & Jenssen, R. (2017). *Recurrent Neural Networks for Short-Term Load Forecasting*. Springer International Publishing.

Bieker, K., Peitz, S., Brunton, S. L., Kutz, J. N., & Dellnitz, M. (2020). Deep model predictive flow control with limited sensor data and online learning. *Theoretical and computational fluid dynamics*, 34, 577–591.

Biferale, L., Bonaccorso, F., Buzzicotti, M., Clark Di Leoni, P., & Gustavsson, K. (2019). Zermelo's problem: Optimal point-to-point navigation in 2D turbulent flows using reinforcement learning. *Chaos*, 29(10), 103138.

Biglari, A. & Sutherland, J. C. (2015). An a-posteriori evaluation of principal component analysis-based models for turbulent combustion simulations. *Combust. Flame*, 162(10).

Bilger, R., Starner, S., & Kee, R. (1990). On reduced mechanisms for methane-air combustion in nonpremixed flames. *Combustion and Flame*, 80, 135–149.

Billings, S. A. (2013). *Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains*. John Wiley & Sons.

Binev, P., Cohen, A., Dahmen, W., DeVore, R. A., Petrova, G., & Wojtaszczyk, P. (2017). Data assimilation in reduced modeling. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1), 1–29.

Birkhoff, G. D. (1931). Proof of the ergodic theorem. *Proceedings of the National Academy of Sciences*, 17(12), 656–660.

Bishop, C. M. (1995). Training with Noise is Equivalent to Tikhonov Regularization. *Neural Computation*, 116, 108–116.

Bishop, C. M. & Bishop, H. (2023). *Deep Learning: Foundations and Concepts*. Springer.

Bishop, C. M. et al. (2006). *Pattern recognition and machine learning*, volume 1 of *Information science and statistics*. New York, NY: springer New York, corrected at 8th printing 2009 edition.

Blakseth, S. S., Rasheed, A., Kvamsdal, T., & San, O. (2022a). Combining physics-based and data-driven techniques for reliable hybrid analysis and modeling using the corrective source term approach. *Applied Soft Computing*, 128, 109533.

Blakseth, S. S., Rasheed, A., Kvamsdal, T., & San, O. (2022b). Deep neural network enabled corrective source term approach to hybrid analysis and modeling. *Neural Networks*, 146, 181–199.

Blanchard, A. & Sapsis, T. (2021). Bayesian optimization with output-weighted optimal sampling. *Journal of Computational Physics*, 425, 109901.

Blonigan, P. J., Fernandez, P., Murman, S. M., Wang, Q., Rigas, G., & Magri, L. (2016). Towards a chaotic adjoint for LES. In *Center for Turbulence Research, Summer Program*.

Bocquet, M. (2011). Ensemble kalman filtering without the intrinsic need for inflation. *Nonlinear Processes in Geophysics*, 18(5), 735–750.

Preview Sample

Bocquet, M. & Farchi, A. (2023). *Introduction to the principles and methods of data assimilation in the geosciences*. Technical report, École des Ponts ParisTech.

Boffetta, G., Cencini, M., Falcioni, M., & Vulpiani, A. (2002). Predictability: a way to characterize complexity. *Physics reports*, 356(6), 367–474.

Bøhn, E., Gros, S., Moe, S., & Johansen, T. A. (2023). Optimization of the model predictive control meta-parameters through reinforcement learning. *Eng. Appl. Artif. Intell.*, 123, 106211.

Bongard, J. & Lipson, H. (2007). Automated reverse engineering of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 104(24), 9943–9948.

Borra, F., Biferale, L., Cencini, M., & Celani, A. (2022). Reinforcement learning for pursuit and evasion of microswimmers at low reynolds number. *Physical Review Fluids*, 7(2), 023103.

Borrelli, F., Bemporad, A., & Morari, M. (2017). *Predictive control for linear and hybrid systems*. Cambridge University Press.

Bouhoubeiny, E. & Druault, P. (2009). Note on the POD-based time interpolation from successive PIV images. *Comptes Rendus Mécanique*, 337(11-12), 776–780.

Bouny, L. E., Khalil, M., & Adib, A. (2020). An end-to-end multi-level wavelet convolutional neural networks for heart diseases diagnosis. *Neurocomputing*, 417, 187–201.

Boyd, S. & Vandenberghe, L. (2009). *Convex optimization*. Cambridge university press.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2), 123–140.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1), 5–32.

Brener, B. P., Cruz, M. A., Macedo, M. S., & Thompson, R. L. (2024). A highly accurate strategy for data-driven turbulence modeling. *Computational and Applied Mathematics*, 43(1), 59.

Brener, B. P., Cruz, M. A., Thompson, R. L., & Anjos, R. P. (2021). Conditioning and accurate solutions of Reynolds average Navier–Stokes equations with data-driven turbulence closures. *J. Fluid Mech.*, 915.

Brenner, M., Eldredge, J., & Freund, J. (2019). Perspective on machine learning for advancing fluid mechanics. *Physical Review Fluids*, 4(10), 100501.

Brenner, M. P. & Koumoutsakos, P. (2021). Machine learning and physical review fluids: An editorial perspective. *Physical Review Fluids*, 6(7), 070001.

Breuer, M., Peller, N., Rapp, C., & Manhart, M. (2009). Flow over periodic hills–numerical and experimental study in a wide range of Reynolds numbers. *Computers & Fluids*, 38(2), 433–457.

Bridges, J. & Wernet, M. (2010). Establishing consensus turbulence statistics for hot subsonic jets. In *16th AIAA/CEAS aeroacoustics conference* (pp. 3751).

Brunton, B. W., Brunton, S. L., Proctor, J. L., & Kutz, J. N. (2016a). Sparse sensor placement optimization for classification. *SIAM Journal on Applied Mathematics*, 76(5), 2099–2122.

Preview Sample

Brunton, S. L., Brunton, B. W., Proctor, J. L., Kaiser, E., & Kutz, J. N. (2017). Chaos as an intermittently forced linear system. *Nature Communications*, 8(19), 1–9.

Brunton, S. L. & Kutz, J. N. (2019). *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press.

Brunton, S. L. & Kutz, N. J. (2022). *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press.

Brunton, S. L. & Noack, B. R. (2015). Closed-loop turbulence control: Progress and challenges. *Appl. Mech. Rev.*, 67(5), 050801.

Brunton, S. L., Noack, B. R., & Koumoutsakos, P. (2020). Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics*, 52, 477–508.

Brunton, S. L., Proctor, J. L., & Kutz, J. N. (2016b). Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15), 3932–3937.

Brunton, S. L., Proctor, J. L., & Kutz, J. N. (2016c). Sparse identification of nonlinear dynamics with control (SINDYc). *Ifac Nolcos*, 49(18), 710–715. 10th IFAC Symposium on Nonlinear Control Systems NOLCOS 2016.

Brunton, S. L., Tu, J. H., Bright, I., & Kutz, J. N. (2014). Compressive sensing and low-rank libraries for classification of bifurcation regimes in nonlinear dynamical systems. *SIAM Journal on Applied Dynamical Systems*, 13(4), 1716–1732.

Bucci, M. A., Semeraro, O., Allauzen, A., Wisniewski, G., Cordier, L., & Mathelin, L. (2019). Control of chaotic systems by deep reinforcement learning. *Proc. R. Soc. A*, 475(2231), 20190351.

Cabot, W. & Moin, P. (2000). Approximate wall boundary conditions in the large-eddy simulation of high Reynolds number flow. *Flow, Turbulence and Combustion*, 63(1-4), 269–291.

Calado, A., Poletti, R., Koloszar, L. K., & Mendez, M. A. (2023). A robust data-driven model for flapping aerodynamics under different hovering kinematics. *Physics of Fluids*, 35(4).

Camacho, E. F. & Alba, C. B. (2013). *Model predictive control*. Springer Science & Business Media.

Campo, P. J. & Morari, M. (1987). Robust model predictive control. In *1987 American control conference* (pp. 1021–1026).: Ieee.

Candès, E. J. (2006). Compressive sensing. *Proceedings of the International Congress of Mathematics*.

Candès, E. J., Romberg, J., & Tao, T. (2006). Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2), 489–509.

Candès, E. J., Romberg, J., & Tao, T. (59). Stable signal recovery from incomplete and inaccurate measurements. *Communications in Pure and Applied Mathematics*, 8(1207–1223).

Preview Sample

Candès, E. J. & Tao, T. (2006). Near optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12), 5406–5425.

Carlberg, K., Barone, M., & Antil, H. (2017). Galerkin v. least-squares petrov–galerkin projection in nonlinear model reduction. *Journal of Computational Physics*, 330, 693–734.

Carlberg, K., Bou-Mosleh, C., & Farhat, C. (2011). Efficient non-linear model reduction via a least-squares petrov–galerkin projection and compressive tensor approximations. *International Journal for Numerical Methods in Engineering*, 86(2), 155–181.

Casa, L. D. C. & Krueger, P. S. (2013). Radial basis function interpolation of unstructured, three-dimensional, volumetric particle tracking velocimetry data. *Measurement Science and Technology*, 24(6), 065304.

Cattafesta III, L. N. & Sheplak, M. (2011). Actuators for active flow control. *Annual Review of Fluid Mechanics*, 43, 247–272.

Cavaliere, A. & de Joannon, M. (2004). Mild combustion. *Prog. Energy Combust. Sci.*, 30, 329–366.

Cawley, G. C. & Talbot, N. L. C. (2010). On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation. *Journal of Machine Learning Research*, 11, 2079–2107.

Chada, N. K., Stuart, A. M., & Tong, X. T. (2020). Tikhonov Regularization within Ensemble Kalman Inversion. *SIAM Journal on Numerical Analysis*, 58(2), 1263–1294.

Champion, K., Lusch, B., Kutz, J. N., & Brunton, S. L. (2019). Data-driven discovery of coordinates and governing equations. *arXiv preprint arXiv: 1904.02107*.

Chaouat, B. (2017). The state of the art of hybrid rans/les modeling for the simulation of turbulent flows. *Flow, Turbulence and Combustion*, 99, 279–327.

Chartrand, R. (2011). Numerical differentiation of noisy, nonsmooth data. *ISRN Applied Mathematics*, 2011.

Chaturantabut, S. & Sorensen, D. C. (2010). Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 32(5), 2737–2764.

Chatzilygeroudis, K., Vassiliades, V., Stulp, F., Calinon, S., & Mouret, J.-B. (2019). A survey on policy search algorithms for learning robot controllers in a handful of trials.

Chen, K., Tu, J., & Rowley, C. (2012a). Variants of dynamic mode decomposition: boundary condition, koopman and fourier analyses. *TJ. Nonlinear Sci.*, 22, 8871—-887.

Chen, K. K., Tu, J. H., & Rowley, C. W. (2012b). Variants of dynamic mode decomposition: Boundary condition, Koopman, and Fourier analyses. *Journal of Nonlinear Science*, 22(6), 887–915.

Chen, W. (2003). New RBF collocation methods and kernel RBF with applications. In *Lecture Notes in Computational Science and Engineering* (pp. 75–86). Springer Berlin Heidelberg.

Chen, W. & Tanaka, M. (2002). A meshless, integration-free, and boundary-only RBF technique. *Computers & Mathematics with Applications*, 43(3-5), 379–391.

Preview Sample

Cherroud, S., Merle, X., Cinnella, P., & Gloerfelt, X. (2022). Sparse bayesian learning of explicit algebraic reynolds-stress models for turbulent separated flows. *International Journal of Heat and Fluid Flow*, 98, 109047.

Cherroud, S., Merle, X., Cinnella, P., & Gloerfelt, X. (2023). Space-dependent aggregation of data-driven turbulence models. *arXiv preprint arXiv:2306.16996*.

Cheung, S. H., Oliver, T. A., Prudencio, E. E., Prudhomme, S., & Moser, R. D. (2011). Bayesian uncertainty analysis with applications to turbulence modeling. *Reliability Engineering & System Safety*, 96(9), 1137–1149.

Chevalier, M., Hœpffner, J., Åkervik, E., & Henningson, D. (2007). Linear feedback control and estimation applied to instabilities in spatially developing boundary layers. *J. Fluid Mech.*, 588, 163–187.

Chien, K.-Y. (1982). Predictions of channel and boundary-layer flows with a low-Reynolds-number turbulence model. *AIAA Journal*, 20(1), 33–38.

Childs, C. M. & Washburn, N. R. (2019). Embedding domain knowledge for machine learning of complex material systems. *MRS Communications*, 9(3), 806–820.

Chinesta, F., Ladeveze, P., & Cueto, E. (2011). A short review on model order reduction based on proper generalized decomposition. *Archives of Computational Methods in Engineering*, 18(4), 395.

Choi, H. & Choi, S. (2004). Kernel isomap. *Electronics Letters*, 40(25), 1612.

Chua, K., Calandra, R., McAllister, R., & Levine, S. (2018). Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems* (pp. 4754–4765).

Cinnella, P., Dwight, R., & Edeling, W. N. (2016). Review of uncertainty quantification in turbulence modelling to date. Minisymposium "UQ in turbulence modelling", SIAM Uncertainty Quantification conference, Lausanne, Switzerland. 5-8 April. doi: 10.13140/RG.2.1.4512.5523.

Clainche, S. L. & Vega, J. M. (2017a). Higher order dynamic mode decomposition. *SIAM Journal on Applied Dynamical Systems*, 16(2), 882–925.

Clainche, S. L. & Vega, J. M. (2017b). Higher order dynamic mode decomposition to identify and extrapolate flow patterns. *Physics of Fluids*, 29(8), 084102.

Cleveland, W., Grosse, E., & Shyu, W. (1992). Local refgression models. *Statistical models*, 79(387), 531–554.

Cordier, L. & Bergmann, M. (2013). Proper orthogonal decomposition: an overview. In L. David & C. Schram (Eds.), *Advanced Post Processing of Experimental and Numerical Data*, volume Vki-ls: von Karman Institute for Fluid Dynamics.

Corrochano, A., D'Alessio, G., Parente, A., & Le Clainche, S. (2023a). Higher order dynamic mode decomposition to model reacting flows. *Int. J. Mech. Sci.*, 249, 108219.

Corrochano, A., Freitas, R., Parente, A., & Le Clainche, S. (2023b). A predictive physics-aware hybrid reduced order model for reacting flows. *arXiv:2301.09860*.

Review Sample

Corrochano, A., Neves, A. F., Khanal, B., Clainche, S. L., & Lawson, N. J. (2022). Des of a slingsby firefly aircraft: Unsteady flow feature extraction using pod and hodmd. *J. Aerosp. Eng.*, 35(5), 04022063.

Cortelezzi, L., Lee, K., Kim, J., & Speyer, J. (1998). Skin-friction drag reduction via robust reduced-order linear feedback control. *International Journal of Computational Fluid Dynamics*, 11(1-2), 79–92.

Coscia, D., Ivagnes, A., Demo, N., & Rozza, G. (2023). Physics-informed neural networks for advanced modeling. *Journal of Open Source Software*, 8(87), 5352.

Course, K. & Nair, P. B. (2023). State estimation of a physical system with unknown governing equations. *Nature*, 622(7982), 261–267.

Coussement, A., Gicquel, O., & Parente, A. (2013). Mg-local-pca method for reduced order combustion modelling. *Proc. Combust. Inst.*, 34, 1117–1123.

Coussement, A., Isaac, B. J., Gicquel, O., & Parente, A. (2016). Assessment of different chemistry reduction methods based on principal component analysis: Comparison of the mg-pca and score-pca approaches. *Combustion and flame*, 168, 83–97.

Cox, M. A. A. & Cox, T. F. (2008). *Multidimensional Scaling*, (pp. 315–347). Springer Berlin Heidelberg.

Cranmer, M., Greydanus, S., Hoyer, S., Battaglia, P., Spergel, D., & Ho, S. (2020a). Lagrangian neural networks. *arXiv preprint arXiv:2003.04630*.

Cranmer, M., Sanchez-Gonzalez, A., Battaglia, P., Xu, R., Cranmer, K., Spergel, D., & Ho, S. (2020b). Discovering symbolic models from deep learning with inductive biases. *arXiv preprint arXiv:2006.11287*.

Cranmer, M. D., Xu, R., Battaglia, P., & Ho, S. (2019). Learning symbolic physics with graph networks. *arXiv preprint arXiv:1909.05862*.

Crutchfield, J. P. & McNamara, B. S. (1987). Equations of motion from a data series. *Complex systems*, 1, 417–452.

Cruz, M. A., Thompson, R. L., Sampaio, L. E., & Bacchi, R. D. (2019). The use of the Reynolds force vector in a physics informed machine learning approach for predictive turbulence modeling. *Comput. Fluids*, 192, 104258.

Cuoci, A., Frassoldati, A., Faravelli, T., & Ranzi, E. (2013a). Numerical modeling of laminar flames with detailed kinetics based on the operator-splitting method. *Energy & Fuels*, 27(12), 7730–7753.

Cuoci, A., Frassoldati, A., Faravelli, T., & Ranzi, E. (2013b). Numerical modeling of laminar flames with detailed kinetics based on the operator-splitting method. *Energy & fuels*, 27(12), 7730–7753.

Cuoci, A., Frassoldati, A., Faravelli, T., & Ranzi, E. (2015). Opensmoke++: An object-oriented frame-work for the numerical modeling of reactive systems with detailed kinetic mechanisms. *Computer Physics Communications*, 192, 237–264.

Cutler, C. R. & Ramaker, B. L. (1980). Dynamic matrix control– A computer control algorithm. In *joint automatic control conference*, number 17 (pp.7̃2).

Preview Sample

Cvitanović, P., Davidchack, R. L., & Siminos, E. (2010). On the state space geometry of the Kuramoto–Sivashinsky flow in a periodic domain. *SIAM Journal on Applied Dynamical Systems*, 9(1), 1–33.

Dam, M., Brøns, M., Juul Rasmussen, J., Naulin, V., & Hesthaven, J. S. (2017). Sparse identification of a predator-prey system from simulation data of a convection model. *Physics of Plasmas*, 24(2), 022310.

Daniels, B. C. & Nemenman, I. (2015a). Automated adaptive inference of phenomenological dynamical models. *Nature communications*, 6.

Daniels, B. C. & Nemenman, I. (2015b). Efficient inference of parsimonious phenomenological models of cellular dynamics using s-systems and alternating regression. *PloS one*, 10(3), e0119821.

Davidson, A. & Hinkley, D. (2009). *Bootstrap methods and their application*. New York, NY: Cambridge University Press.

Dawson, S. (2023). The proper orthogonal decomposition. In M. A. Mendez, A. Ianiro, B. R. Noack, & S. L. Brunton (Eds.), *Data-Driven Fluid Mechanics: Combining First Principles and Machine Learning* (pp. 117–132). Cambridge University Press.

de Bezenac, E., Pajot, A., & Gallinari, P. (2019). Deep learning for physical processes: Incorporating prior scientific knowledge. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12), 124009.

De Lathauwer, L., De Moor, B., & Vandewalle, J. (2000). On the best rank-1 and rank-(r1 ,r2 ,. . .,rn) approximation of higher-order tensors. *SIAM Journal on Matrix Analysis and Applications*, 21(4), 1324–1342.

De Lathawer, L., De Moor, B., & Vandewalle, J. (2000). A multilinear singular value decomposition. *SIAM J. Matrix. Anal. Appl.*, (21), 1253–1278.

de Zordo-Banliat, M., Dergham, G., Merle, X., & Cinnella, P. (2020). Space-dependent turbulence model aggregation using machine learning. *Journal of Computational Physics*, 497, 112628.

de Zordo-Banliat, M., Merle, X., Dergham, G., & Cinnella, P. (2020). Bayesian model-scenario averaged predictions of compressor cascade flows under uncertain turbulence models. *Computers & Fluids*, 201, 104473.

de Zordo-Banliat, M., Merle, X., Dergham, G., & Cinnella, P. (2022). Estimates of turbulence modeling uncertainties in naca65 cascade flow predictions by bayesian model-scenario averaging. *International Journal of Numerical Methods for Heat & Fluid Flow*, 32, 1398–1414.

Déda, T., Wolf, W. R., & Dawson, S. T. (2023). Backpropagation of neural network dynamical models applied to flow control. *Theoretical and Computational Fluid Dynamics*, 37(1), 35–59.

Deisenroth, M. P., Faisal, A. A., & Ong, C. S. (2020). *Mathematics for Machine Learning*. Cambridge University Press.

Preview Sample

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248–255).: Ieee.

Deng, N., Noack, B. R., Morzyński, M., & Pastur, L. R. (2020). Low-order model for successive bifurcations of the fluidic pinball. *Journal of fluid mechanics*, 884, A37.

Deng, N., Noack, B. R., Morzyński, M., & Pastur, L. R. (2021). Galerkin force model for transient and post-transient dynamics of the fluidic pinball. *Journal of Fluid Mechanics*, 918.

Deng, N., Noack, B. R., Morzyński, M., & Pastur, L. R. (2022). Cluster-based hierarchical network model of the fluidic pinball–cartographing transient and post-transient, multi-frequency, multi-attractor behaviour. *Journal of Fluid Mechanics*, 934, A24.

Ding, Y., Wang, L., Li, Y., & Li, D. (2018). Model predictive control and its application in agriculture: A review. *Computers and Electronics in Agriculture*, 151, 104–117.

Diomede, T., Davolio, S., Marsigli, C., Miglietta, M., Moscatello, A., Papetti, P., Paccagnella, T., Buzzi, A., & Malguzzi, P. (2008). Discharge prediction based on multi-model precipitation forecasts. *Meteorology and Atmospheric Physics*, 101(3-4), 245–265.

Dissanayake, M. & Phan-Thien, N. (1994). Neural-network-based approximations for solving partial differential equations. *communications in Numerical Methods in Engineering*, 10(3), 195–201.

Doan, N. A. K., Polifke, W., & Magri, L. (2020). Physics-informed echo state networks. *Journal of Computational Science*, 47, 101237.

Doan, N. A. K., Polifke, W., & Magri, L. (2021). Short-and long-term predictions of chaotic flows and extreme events: a physics-constrained reservoir computing approach. *Proceedings of the Royal Society A*, 477(2253), 20210135.

Dominique, J., Van den Berghe, J., Schram, C., & Mendez, M. A. (2022). Artificial neural networks modeling of wall pressure spectra beneath turbulent boundary layers. *Physics of Fluids*, 34(3).

Donoho, D. L. (2006). Compressed sensing. *IEEE Transactions on Information Theory*, 52(4), 1289–1306.

Draper, D. (1995). Assessment and propagation of model uncertainty. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57(1), 45–97.

Du, M., Liu, N., & Hu, X. (2019). Techniques for interpretable machine learning. *Communications of the ACM*, 63(1), 68–77.

Duan, Q., Ajami, N. K., Gao, X., & Sorooshian, S. (2007). Multi-model ensemble hydrologic prediction using Bayesian model averaging. *Advances in Water Resources*, 30(5), 1371–1386.

Duraisamy, K. (2021a). Perspectives on machine learning-augmented Reynolds-Averaged and Large Eddy Simulation models of turbulence. *Physical Review Fluids*, 6, 050504.

Duraisamy, K. (2021b). Perspectives on machine learning-augmented reynolds-averaged and large eddy simulation models of turbulence. *Physical Review Fluids*, 6(5), 050504.

Preview Sample

Duraisamy, K., Iaccarino, G., & Xiao, H. (2019). Turbulence modeling in the age of data. *Annual Review of Fluid Mechanics*, 51(1), 357–377.

Durbin, P. (1991). Near-wall turbulence closure modelling without damping functions. *Theoretical and Computational Fluid Dynamics*, 3, 1–13.

Durbin, P. A. (2018). Some recent developments in turbulence closure modeling. *Annual Review of Fluid Mechanics*, 50(50), 77–103.

Duriez, T., Brunton, S. L., & Noack, B. R. (2016). *Machine Learning Control: Taming Nonlinear Dynamics and Turbulence*. Springer.

Dussauge, T. P., Sung, W. J., Pinon Fischer, O. J., & Mavris, D. N. (2023). A reinforcement learning approach to airfoil shape optimization. *Scientific Reports*, 13(1), 9753.

Díaz, P., Corrochano, A., López-Martín, M., & Clainche, S. L. (2023). Deep learning combined with singular value decomposition to reconstruct databases in fluid dynamics.

D'Alessio, G., Attili, A., Cuoci, A., Pitsch, H., & Parente, A. (2020a). Analysis of turbulent reacting jets via principal component analysis. In *Data Analysis for Direct Numerical Simulations of Turbulent Combustion* (pp. 233–251). Springer.

D'Alessio, G., Cuoci, A., Aversano, G., Bracconi, M., Stagni, A., & Parente, A. (2020b). Impact of the partitioning method on multidimensional adaptive-chemistry simulations. *Energies*, 13(10), 2567.

D'Alessio, G., Parente, A., Stagni, A., & Cuoci, A. (2020c). Adaptive chemistry via pre-partitioning of composition space and mechanism reduction. *Combust. Flame.*, 211, 68–82.

Echekki, T. & Mirgolbabaei, H. (2015). Principal component transport in turbulent combustion: a posteriori analysis. *Combust. Flame*, 162(5), 1919–1933.

Eckmann, J. P. & Ruelle, D. (1985). Ergodic theory of chaos and strange attractors. *Rev. Mod. Phys.*, 57, 617–656.

Ecmwf (2024). European centre for medium range weather forecasts.

Edeling, W., Cinnella, P., & Dwight, R. P. (2014a). Predictive RANS simulations via Bayesian model-scenario averaging. *Journal of Computational Physics*, 275, 65–91.

Edeling, W. N., Cinnella, P., Dwight, R. P., & Bijl, H. (2014b). Bayesian estimates of parameter variability in the k–ε turbulence model. *Journal of Computational Physics*, 258, 73–94.

Edeling, W. N., Schmelzer, M., Cinnella, P., & Dwight, R. P. (2018). Bayesian predictions of Reynolds-averaged Navier–Stokes uncertainties using maximum a posteriori estimates. *AIAA Journal*, (pp. 1–12).

Edwards, W., Tang, G., Mamakoukas, G., Murphey, T., & Hauser, K. (2021). Automatic tuning for data-driven model predictive control. In *2021 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 7379–7385).: Ieee.

Efron, B. & Tibshirani, R. J. (1993). *An introduction to the bootstrap*. London: Chapman & Hall/CRC.

Preview Sample

Eivazi, H., Le Clainche, S., Hoyas, S., & Vinuesa, R. (2022). Towards extraction of orthogonal and parsimonious non-linear modes from turbulent flows. *Expert Syst. Appl.*, 202, 117038.

Eivazi, H., Veisi, H., Naderi, M. H., & Esfahanian, V. (2020). Deep neural networks for nonlinear model order reduction of unsteady flows. *Physics of Fluids*, 32(10), 105104.

Elsinga, G., Scarano, F., Wieneke, B., & Oudheusden, B. (2006). Tomographic Particle Image Velocimetry. *Experiments in Fluids*, 41, 933–947.

Emory, M., Larsson, J., & Iaccarino, G. (2013). Modeling of structural uncertainties in Reynolds-averaged Navier-Stokes closures. *Physics of Fluids*, 25(11), 110822.

Emory, M., Pecnik, R., & Iaccarino, G. (2011). Modeling structural uncertainties in Reynolds-averaged computations of shock/boundary layer interactions. In *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. Orlando, FL, 2011 (AIAA, Reston, VA, 2017), paper 2011-479.

Erichson, N. B., Mathelin, L., Yao, Z., Brunton, S. L., Mahoney, M. W., & Kutz, J. N. (2020). Shallow neural networks for fluid flow reconstruction with limited sensors. *Proceedings of the Royal Society A*, 476(2238), 20200097.

Erichson, N. B., Muehlebach, M., & Mahoney, M. W. (2019). Physics-informed autoencoders for lyapunov-stable fluid flow prediction. *arXiv preprint arXiv:1905.10866*.

Evensen, G. (2003). The ensemble Kalman filter: theoretical formulation and practical implementation. *Ocean Dynamics*, 53(4), 343–367.

Evensen, G. (2009). *Data Assimilation: The Ensemble Kalman Filter*. Spinger, second edition.

Fabbiane, N., Semeraro, O., Bagheri, S., & Henningson, D. S. (2014). Adaptive and model-based control theory applied to convectively unstable flows. *Applied Mechanics Reviews*, 66(6), 060801–1–060801–20.

Fabbiane, N., Simon, B., Fischer, F., Grundmann, S., Bagheri, S., & Henningson, D. S. (2015). On the role of adaptivity for robust laminar flow control. *J. Fluid Mech.*, 767.

Fan, D., Jodin, G., Consi, T., Bonfiglio, L., Ma, Y., Keyes, L., Karniadakis, G. E., & Triantafyllou, M. S. (2019). A robotic intelligent towing tank for learning complex fluid-structure dynamics. *Science Robotics*, 4(36).

Fan, D., Yang, L., Wang, Z., Triantafyllou, M. S., & Karniadakis, G. E. (2020). Reinforcement learning for bluff body active flow control in experiments and simulations. *Proceedings of the National Academy of Sciences*, 117(42), 26091–26098.

Fan, J. & Gijbels, I. (1996). *Local polynomial modelling and its applications*. Number 66 in Monographs on statistics and applied probability series. London [u.a.]: Chapman & Hall.

Fang, Y., Zhao, Y., Waschkowski, F., Ooi, A. S., & Sandberg, R. D. (2023). Toward more general turbulence models via multicase computational-fluid-dynamics-driven training. *AIAA Journal*, 61(5), 2100–2115.

Farhat, A., Glatt-Holtz, N., Martinez, V. R., McQuarrie, S. A., & Whitehead, J. P. (2020). Data Assimilation in Large Prandtl Rayleigh–Bénard Convection from Thermal Measurements. *Siam Journal on Applied Dynamical Systems*, 19(1), 510–540.

Preview Sample

Farmer, J. D., Ott, E., & Yorke, J. A. (1983). The dimension of chaotic attractors. *Physica D: Nonlinear Phenomena*, 7(1-3), 153–180.

Fernandez, P. & Wang, Q. (2017). Lyapunov spectrum of the separated flow around the naca 0012 airfoil and its dependence on numerical discretization. *Journal of Computational Physics*, 350, 453–469.

Fernex, D., Noack, B. R., & Semaan, R. (2021). Cluster-based network modeling—from snapshots to complex dynamical systems. *Science Advances*, 7(25), eabf5006.

Ferrarotti, M., Fürst, M., Cresci, E., De Paepe, W., & Parente, A. (2018). Key modeling aspects in the simulation of a quasi-industrial 20 kw moderate or intense low-oxygen dilution combustion chamber. *Energy & fuels*, 32(10), 10228–10241.

Ferreira, C. (2001). Gene expression programming: A new adaptive algorithm for solving problems. *Complex Systems*, 13, 87–129.

Ferrero, A., Iollo, A., & Larocca, F. (2020). Field inversion for data-augmented RANS modelling in turbomachinery flows. *Computers & Fluids*, 201, 104474.

Finzi, M., Wang, K. A., & Wilson, A. G. (2020). Simplifying hamiltonian and lagrangian neural networks via explicit constraints. *Advances in Neural Information Processing Systems*, 33.

Fiore, M., Koloszar, L., Fare, C., Mendez, M. A., Duponcheel, M., & Bartosiewicz, Y. (2022). Physics-constrained machine learning for thermal turbulence modelling at low prandtl numbers. *International Journal of Heat and Mass Transfer*, 194, 122998.

Fornberg, B. (1996). *A Practical Guide to Pseudospectral Methods*. Cambridge University Press.

Fornberg, B. & Flyer, N. (2015). Solving PDEs with radial basis functions. *Acta Numerica*, 24, 215–258.

Forrester, A. I. & Keane, A. J. (2009). Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45(1-3), 50–79.

Fortin, F.-A., De Rainville, F.-M., Gardner, M.-A., Parizeau, M., & Gagné, C. (2012). DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13, 2171–2175.

Foucart, C., Charous, A., & Lermusiaux, P. F. (2023). Deep reinforcement learning for adaptive mesh refinement. *Journal of Computational Physics*, 491, 112381.

François, D. G., Krumbein, A., Krimmelbein, N., & Grabe, C. (2023). Simplified stability-based transition transport modeling for unstructured computational fluid dynamics. *Journal of Aircraft*, (pp. 1–12).

Frassoldati, A., Faravelli, T., & Ranzi, E. (2003). Kinetic modeling of the interactions between no and hydrocarbons at high temperature. *Combust. Flame*, 135, 97–112.

Frederickson, P., Kaplan, J. L., Yorke, E. D., & Yorke, J. A. (1983). The liapunov dimension of strange attractors. *Journal of differential equations*, 49(2), 185–207.

Frezat, H., Balarac, G., Le Sommer, J., Fablet, R., & Lguensat, R. (2021). Physical invariance in neural networks for subgrid-scale scalar flux modeling. *Physical Review Fluids*, 6(2), 024607.

Preview Sample

Friedman, J. (1991). Multivariate adaptive regression splines. *The Annals of Statictics*, 19(1), 1–67.

Fröhlich, J. & von Terzi, D. (2008). Hybrid LES/RANS methods for the simulation of turbulent flows. *Progress in Aerospace Sciences*, 44(5), 349–377.

Fröhlich, L. P., Küttel, C., Arcari, E., Hewing, L., Zeilinger, M. N., & Carron, A. (2022). Contextual Tuning of Model Predictive Control for Autonomous Racing. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 10555–10562).: Ieee.

Fukami, K., Fukagata, K., & Taira, K. (2019). Super-resolution reconstruction of turbulent flows with machine learning. *Journal of Fluid Mechanics*, 870, 106–120.

Fukami, K., Hasegawa, K., Nakamura, T., Morimoto, M., & Fukagata, K. (2021). Model order reduction with neural networks: Application to laminar and turbulent flows. *SN Computer Science*, 2(6).

Gabrys, B. & Strandt, S. (2008). Data-driven Soft Sensors in the Process Industry. *Computers & Chemical Engineering*.

Gad-el Hak, M. (2000). *Flow Control: Passive, Active, and Reactive Flow Management*. Cambridge University Press.

Gal, Y. & Ghahramani, Z. (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning* (pp. 1050–1059).: Pmlr.

Garcia, C. E. & Morshedi, A. (1986). Quadratic programming solution of dynamic matrix control (QDMC). *Chemical Engineering Communications*, 46(1-3), 73–87.

Garcıa, J. & Fernández, F. (2015). A comprehensive survey on safe reinforcement learning. *J. Mach. Learn. Res.*, 16(1), 1437–1480.

Gatski, T. & Speziale, C. (1993). On explicit algebraic stress models for complex turbulent flows. *Journal of Fluid Mechanics*, 254, 59–79.

George, W. K. (2017). A 50-year retrospective and the future. In *Whither Turbulence and Big Data in the 21st Century?* (pp. 13–43). Springer.

Gerolymos, G., Lo, C., Vallet, I., & Younis, B. (2012). Term-by-term analysis of near-wall second-moment closures. *AIAA journal*, 50(12), 2848–2864.

Gesemann, S., Huhn, F., Schanz, D., & Schröder, A. (2016). From noisy particle tracks to velocity, acceleration and pressure fields using b-splines and penalties. In *18th International Symposium on Applications of Laser Techniques to Fluid Mechanics*.

Ghattas, O. & Willcox, K. (2021). Learning physics-based models from data: perspectives from inverse problems and model reduction. *Acta Numerica*, 30, 445–554.

Ghojogh, B. & Crowley, M. (2019). Unsupervised and supervised principal component analysis: Tutorial.

Ghojogh, B., Ghodsi, A., Karray, F., & Crowley, M. (2020a). Locally linear embedding and its variants: Tutorial and survey.

Preview Sample

Ghojogh, B., Ghodsi, A., Karray, F., & Crowley, M. (2020b). Multidimensional scaling, sammon mapping, and isomap: Tutorial and survey.

Ghojogh, B., Ghodsi, A., Karray, F., & Crowley, M. (2021). Unified framework for spectral dimensionality reduction, maximum variance unfolding, and kernel learning by semidefinite programming: Tutorial and survey.

Girimaji, S. S. (2006). Partially-averaged Navier-Stokes model for turbulence: A Reynolds-averaged Navier–Stokes to direct numerical simulation bridging method. *Journal of Applied Mechanics*, 73(3), 413–421.

Glad, T. & Ljung, L. (2000). *Control Theory*. London: Taylor & Francis.

Glaessgen, E. & Stargel, D. (2012). The digital twin paradigm for future nasa and u.s. air force vehicles. In *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference;20th AIAA/ASME/AHS Adaptive Structures Conferenceamp; 14th AIAA*: American Institute of Aeronautics and Astronautics.

Glaz, B., Liu, L., & Friedmann, P. P. (2010). Reduced-order nonlinear unsteady aerodynamic modeling using a surrogate-based recurrence framework. *AIAA journal*, 48(10), 2418–2429.

Goldin, N., King, R., Pätzold, A., Nitsche, W., Haller, D., & Woias, P. (2013). Laminar flow control with distributed surface actuation: damping tollmien-schlichting waves with active surface displacement. *Experiments in fluids*, 54, 1–11.

Golub, G. H., Mahoney, M. W., Drineas, P., & Lim, L.-H. (2006). Bridging the gap between numerical linear algebra, theoretical computer science, and data applications. *SIAM News*, 39(8), 1–3.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Goodwin, D. G., Speth, R. L., Moffat, H. K., & Weber, B. W. (2016). Cantera: An object-oriented software toolkit for chemical kinetics, thermodynamics, and transport processes. `https://www.cantera.org`. Version 2.2.1.

Gorlé, C., Garcia-Sanchez, C., & Iaccarino, G. (2015). Quantifying inflow and RANS turbulence model form uncertainties for wind engineering flows. *Journal of Wind Engineering and Industrial Aerodynamics*, 144, 202–212.

Gorlé, C. & Iaccarino, G. (2013). A framework for epistemic uncertainty quantification of turbulent scalar flux models for Reynolds-averaged Navier-Stokes simulations. *Physics of Fluids*, 25(5), 055105.

Gorodetsky, A. A. (2017). *Continuous low-rank tensor decompositions, with applications to stochastic optimal control and data assimilation*. PhD thesis, Massachusetts Institute of Technology.

Grant, M., Boyd, S., & Ye, Y. (2008). Cvx: Matlab software for disciplined convex programming.

Gray, R. M. (2005). Toeplitz and circulant matrices: A review. *Foundations and Trends® in Communications and Information Theory*, 2(3), 155–239.

Grepl, M. A., Maday, Y., Nguyen, N. C., & Patera, A. T. (2007). Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations. *ESAIM: Mathematical Modelling and Numerical Analysis*, 41(3), 575–605.

Preview Sample

Greydanus, S., Dzamba, M., & Yosinski, J. (2019). Hamiltonian neural networks. *Advances in Neural Information Processing Systems*, 32, 15379–15389.

Groun, N., Villalba-Orero, M., Lara-Pezzi, E., Valero, E., Garicano-Mena, J., & Le Clainche, S. (2022a). Higher order dynamic mode decomposition: From fluid dynamics to heart disease analysis. *Comp. Biolog. Medic.*, 144, 105384.

Groun, N., Villalba-Orero, M., Lara-Pezzi, E., Valero, E., Garicano-Mena, J., & Le Clainche, S. (2022b). A novel data-driven method for the analysis and reconstruction of cardiac cine mri. *Comp. Biolog. Medic.*, 151, 106317.

Grüne, L., Pannek, J., Grüne, L., & Pannek, J. (2017). *Nonlinear model predictive control*. Springer.

Guastoni, L., Güemes, A., Ianiro, A., Discetti, S., Schlatter, P., Azizpour, H., & Vinuesa, R. (2021). Convolutional-network models to predict wall-bounded turbulence from wall quantities. *Journal of Fluid Mechanics*, 928, A27.

Guéniat, F., Mathelin, L., & Hussaini, M. Y. (2016). A statistical learning strategy for closed-loop control of fluid flows. *Theo. Comput. Fluid Dyn.*, 30, 1–14.

Gugercin, S., Antoulas, A. C., & Beattie, C. (2008). H_2 model reduction for large-scale linear dynamical systems. *SIAM Journal on Matrix Analysis and Applications*, 30(2), 609–638.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., et al. (2018). Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*.

Hagan, M. T., Demuth, H. B., Beale, M. H., & De Jesús, O. (1996). *Neural network design*, volume 20. Pws Pub. Boston.

Hagan, M. T. & Menhaj, M. B. (1994). Training feedforward networks with the marquardt algorithm. *IEEE Transactions on Neural Networks*, 5(6), 989–993.

Haghighat, E. & Juanes, R. (2021). Sciann: A keras/tensorflow wrapper for scientific computations and physics-informed deep learning using artificial neural networks. *Computer Methods in Applied Mechanics and Engineering*, 373.

Halton, J. H. (1960). On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik*, 2(1), 84–90.

Hamilton, F., Berry, T., & Sauer, T. (2019). Correcting observation model error in data assimilation. *Chaos*, 29(5), 53102.

Han, Z.-H., Abu-Zurayk, M., Görtz, S., & Ilic, C. (2018). Surrogate-based aerodynamic shape optimization of a wing-body transport aircraft configuration. In R. Heinrich (Ed.), *AeroStruct: Enable and Learn How to Integrate Flexibility in Design*, volume 138 of *Notes on Numerical Fluid Mechanics and Multidisciplinary Design* (pp. 257–282). Cham: Springer International Publishing.

Hansen, K. S., Barthelmie, R. J., Jensen, L. E., & Sommer, A. (2012). The impact of turbulence intensity and atmospheric stability on power deficits due to wind turbine wakes at Horns Rev wind farm. *Wind Energy*, 15(1), 183–196.

Preview Sample

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del R'ıo, J. F., Wiebe, M., Peterson, P., G'erard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., & Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362.

Hassanaly, M. & Raman, V. (2019). Ensemble-LES analysis of perturbation response of turbulent partially-premixed flames. *Proc. Combust. Inst.*, 37(2), 2249–2257.

Hasselmann, K. (1988). PIPs and POPs: The reduction of complex dynamical systems using principal interaction and oscillation patterns. *Journal of Geophysical Research*, 93(D9), 11015.

Hastie, T., Tibshirani, R., & Friedman, J. (2009a). *The Elements of Statistical Learning*, volume 1. Springer New York.

Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J., & Tibshirani, R. (2009b). *The elements of statistical learning*, volume 2. Springer.

Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1), 97–109.

Haupt, R. L. & Haupt, S. E. (2003). *Practical Genetic Algorithms*. Wiley.

Havenstrøm, S. T., Rasheed, A., & San, O. (2021). Deep reinforcement learning controller for 3D path following and collision avoidance by autonomous underwater vehicles. *Frontiers in Robotics and AI*, (pp. 211).

He, X., Tan, J., Rigas, G., & Vahdati, M. (2022). On the explainability of machine-learning-assisted turbulence modeling for transonic flows. *International Journal of Heat and Fluid Flow*, 97, 109038.

Heiberg, A., Larsen, T. N., Meyer, E., Rasheed, A., San, O., & Varagnolo, D. (2022). Risk-based implementation of COLREGs for autonomous surface vehicles using deep reinforcement learning. *Neural Networks*, 152, 17–33.

Hendrik, T. & Lumley, J. L. (1972). *A First Course in Turbulence*. Cambridge, Massachusetts.

Hervé, A., Sipp, D., Schmid, P. J., & Samuelides, M. (2012). A physics-based approach to flow control using system identification . *J. Fluid Mech.*, 702, 26–58.

Hewer, G. (1971). An iterative technique for the computation of the steady state gains for the discrete optimal regulator. *IEEE Transactions on Automatic Control*, 16(4), 382–384.

Hewing, L., Wabersich, K. P., Menner, M., & Zeilinger, M. N. (2020). Learning-based model predictive control: Toward safe learning in control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3, 269–296.

Higham, J. E., Brevis, W., & Keylock, C. J. (2016). A rapid non-iterative proper orthogonal decomposition based outlier detection and correction for PIV data. *Measurement Science and Technology*, 27(12), 125303.

Hilborn, R. C. (2000). *Chaos and nonlinear dynamics: an introduction for scientists and engineers*. Oxford university press.

Preview Sample

Hines, D. & Bekemeyer, P. (2023). Graph neural networks for the prediction of aircraft surface pressure distributions. *Aerospace Science and Technology*, 137, 108268.

Hinton, G. E. & Roweis, S. T. (2002). Stochastic neighbor embedding. In *Neural Information Processing Systems* (pp. 857–864).

Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8), 1735–1780.

Hoeting, J. A., Madigan, D., Raftery, A. E., & Volinsky, C. T. (1999). Bayesian model averaging: a tutorial. *Statistical science*, 14(4), 382–401.

Högberg, M., Bewley, T. R., & Henningson, D. S. (2003). Linear feedback control and estimation of transition in plane channel flow. *Journal of Fluid Mechanics*, 481, 149–175.

Högberg, M. & Henningson, D. S. (2002). Linear optimal control applied to instabilities in spatially developing boundary layers. *Journal of Fluid Mechanics*, 470, 151–179.

Holland, J. R., Baeder, J. D., & Duraisamy, K. (2019a). Field inversion and machine learning with embedded neural networks: Physics-consistent neural network training. In *AIAA Aviation 2019 Forum* (pp. 3200).

Holland, J. R., Baeder, J. D., & Duraisamy, K. (2019b). Towards integrated field inversion and machine learning with embedded neural networks for rans modeling. In *AIAA Scitech 2019 forum* (pp. 1884).

Holmes, P. & Guckenheimer, J. (1983). *Nonlinear oscillations, dynamical systems, and bifurcations of vector fields*, volume 42 of *Applied Mathematical Sciences*. Berlin, Heidelberg: Springer-Verlag.

Holmes, P. J., Lumley, J. L., Berkooz, G., & Rowley, C. W. (2012). *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge Monographs in Mechanics. Cambridge, England: Cambridge University Press, 2nd edition.

Honeine, P. & Richard, C. (2011). Preimage problem in kernel-based machine learning. *IEEE Signal Processing Magazine*, 28(2), 77–88.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2), 251–257.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5), 359–366.

Howland, M. F., Quesada, J. B., Martinez, J. J. P., Larrañaga, F. P., Yadav, N., Chawla, J. S., Sivaram, V., & Dabiri, J. O. (2022). Collective wind farm operation based on a predictive model increases utility-scale energy production. *Nature Energy*.

Hrovat, D., Di Cairano, S., Tseng, H. E., & Kolmanovsky, I. V. (2012). The development of model predictive control in automotive industry: A survey. In *2012 IEEE International Conference on Control Applications* (pp. 295–302).: Ieee.

Hu, B. & Linnemann, A. (2002). Toward infinite-horizon optimality in nonlinear model predictive control. *IEEE Transactions on Automatic Control*, 47(4), 679–682.

Preview Sample

Huber, P. J. (1964). Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1), 73–101.

Huhn, F. & Magri, L. (2020a). Learning ergodic averages in chaotic systems. In *Computational Science – ICCS 2020* (pp. 124–132).: Springer International Publishing.

Huhn, F. & Magri, L. (2020b). Stability, sensitivity and optimisation of chaotic acoustic oscillations. *Journal of Fluid Mechanics*, 882, A24.

Huhn, F. X. (2022). *Optimisation of chaotic thermoacoustics*. PhD thesis, University of Cambridge.

Humer, S., Frassoldati, A., Granata, S., Faravelli, T., Ranzi, E., Seiser, R., & Seshadri, K. (2007). Experimental and kinetic modeling study of combustion of jp-8, its surrogates and reference components in laminar nonpremixed flows. *Proceedings of the Combustion Institute*, 31, 393–400.

Iglesias, M. A., Law, K. J. H., & Stuart, A. M. (2013). Ensemble Kalman methods for inverse problems. *Inverse Problems*, 29(4), 045001 (20pp).

Isaac, B., Coussement, A., Gicquel, O., Smith, P., & Parente, A. (2014). Reduced-order pca models for chemical reacting flows. *Combust. Flame*, 161.

Isaac, B., Thornock, J., Sutherland, J., Smith, P., & Parente, A. (2015). Advanced regression methods for combustion modelling using principal components. *Combust. Flame*, 162(6), 2592–2601.

J. Sousa, C.G Sànchez, C. G. (2018). Improving urban flow predictions through data assimilation. *Building and Environment*.

Jäckel, F. (2021). Sensitivity analysis of discrepancy terms introduced in turbulence models using field inversion. In A. Dillmann, G. Heller, E. Krämer, & C. Wagner (Eds.), *New Results in Numerical and Experimental Fluid Mechanics XIII* (pp. 625–634). Cham: Springer International Publishing.

Jackson, C. P. (1987). A finite-element study of the onset of vortex shedding in flow past variously shaped bodies. *Journal of Fluid Mechanics*, 182, 23–45.

Jaeger, H. (2001). *The "echo state" approach to analysing and training recurrent neural networks*. Technical Report GMD Report 148, GMD - German National Research Institute for Computer Science.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning*, volume 112. Springer.

Jeon, Y., Müller, M., & Michaelis, D. (2022). Fine scale reconstruction (VIC#) by implementing additional constraints and coarse-grid approximation into VIC+. *Experiments in Fluids*, 63.

Jiang, C., Vinuesa, R., Chen, R., Mi, J., Laima, S., & Li, H. (2021). An interpretable framework of data-driven turbulence modeling using deep neural networks. *Physics of Fluids*, 33(5), 055133.

Jiang, J. & Lai, Y.-C. (2019). Model-free prediction of spatiotemporal dynamical systems with recurrent neural networks: Role of network spectral radius. *Physical Review Research*, 1(3), 033056.

Preview Sample

Jolliffe, I. (2002). *Principal component analysis*. Wiley Online Library.

Jolly, M. S., Martinez, V. R., & Titi, E. S. (2017). A data assimilation algorithm for the subcritical surface quasi-geostrophic equation. *Advanced Nonlinear Studies*, 17(1).

Jovanović, M. R. & Bamieh, B. (2005). Componentwise energy amplification in channel flows. *J. Fluid Mech.*, 534, 145–183.

Juillet, F., McKeon, B., & Schmid, P. J. (2014). Experimental control of natural perturbations in channel flow. *J. Fluid Mech.*, 752, 296–309.

J.Wünning & J.Wünning (1997). Flameless oxidation to reduce thermal NO-formation. *Prog. Energy Combust. Sci.*, 23, 81–94.

Jäckel, F. (2022). A closed-form correction for the spalart-allmaras turbulence model for separated flows. In *AIAA SCITECH 2022 Forum*.

Kaandorp, M. L. & Dwight, R. P. (2020). Data-driven modelling of the reynolds stress tensor using random forests with invariance. *Computers & Fluids*, 202, 104497.

Kaiser, E., Kutz, J. N., & Brunton, S. L. (2018a). Discovering conservation laws from data for control. In *2018 IEEE Conference on Decision and Control (CDC)* (pp. 6415–6421).: Ieee.

Kaiser, E., Kutz, J. N., & Brunton, S. L. (2018b). Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. *Proceedings of the Royal Society of London A*, 474(2219), 20180335.

Kaiser, E., Noack, B. R., Cordier, L., Spohn, A., Segond, M., Abel, M., Daviller, G., Osth, J., Krajnovic, S., & Niven, R. K. (2014). Cluster-based reduced-order modelling of a mixing layer. *J. Fluid Mech.*, 754, 365–414.

Kaiser, H. F. (1958). The varimax criterion for analytic rotation in factor analysis. *Psychometrika*, 23(3), 187–200.

Kalnay, E. (2003). *Atmospheric Modeling, Data Assimilation and Predictability*. Cambridge University Press.

Kambhatla, N. & Leen, T. K. (1997). Dimension reduction by local principal component analysis. *Neural computation*, 9(7), 1493–1516.

Kansa, E. (1990a). Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics—i surface approximations and partial derivative estimates. *Computers & Mathematics with Applications*, 19(8-9), 127–145.

Kansa, E. (1990b). Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics—II solutions to parabolic, hyperbolic and elliptic partial differential equations. *Computers & Mathematics with Applications*, 19(8-9), 147–161.

Kantz, H. & Schreiber, T. (2004). *Nonlinear time series analysis*, volume 7. Cambridge university press.

Kaptanoglu, A., Morgan, K., Hansen, C., & Brunton, S. (2021a). Physics-constrained, low-dimensional models for magnetohydrodynamics: First-principles and data-driven approaches. *Phys. Rev. E*, 104.

Preview Sample

Kaptanoglu, A. A., Callaham, J. L., Hansen, C. J., Aravkin, A., & Brunton, S. L. (2021b). Promoting global stability in data-driven models of quadratic nonlinear dynamics. *arXiv preprint arXiv:2105.01843*.

Kapteyn, M. G., Pretorius, J. V., & Willcox, K. E. (2021). A probabilistic graphical model foundation for enabling predictive digital twins at scale. *Nature Computational Science*, 1(5), 337–347.

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., & Yang, L. (2021). Physics-informed machine learning. *Nature Reviews Physics*, 3(6), 422–440.

Karpatne, A., Atluri, G., Faghmous, J. H., Steinbach, M., Banerjee, A., Ganguly, A., Shekhar, S., Samatova, N., & Kumar, V. (2017). Theory-guided data science: A new paradigm for scientific discovery from data. *IEEE Transactions on Knowledge and Data Engineering*, 29(10), 2318–2331.

Kato, H. & Obayashi, S. (2013). Approach for uncertainty of turbulence modeling based on data assimilation technique. *Computers & Fluids*, 85, 2–7.

Kawai, S. & Larsson, J. (2012). Wall-modeling in large eddy simulation: Length scales, grid resolution, and accuracy. *Physics of Fluids*, 24(1).

Kennedy, M. C. & O'Hagan, A. (2001). Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3), 425–464.

Keramati, H., Hamdullahpur, F., & Barzegari, M. (2022). Deep reinforcement learning for heat exchanger shape optimization. *International Journal of Heat and Mass Transfer*, 194, 123112.

Kevrekidis, I. G., Gear, C. W., Hyman, J. M., Kevrekidis, P. G., Runborg, O., & Theodoropoulos, C. (2003). Equation-free, coarse-grained multiscale computation: Enabling microscopic simulators to perform system-level analysis. *Communications in Mathematical Science*, 1(4), 715–762.

Khurana, P., Jäckel, F., Grabe, C., & Dwight, R. (2022). Data-driven augmentation of a rans turbulence model for transonic flow prediction. In *56th 3AF International Conference on Applied Aerodynamics* Toulouse.

Kim, J. (2003). Control of turbulent boundary layers. *Physics of Fluids (1994-present)*, 15(5), 1093–1105.

Kim, J. & Bewley, T. (2007). A linear systems approach to flow control. *Ann. Rev. Fluid Mech.*, 39, 383–417.

Kim, J.-H. (2009). Estimating classification error rate: Repeated cross-validation, repeated hold-out and bootstrap. *Computational Statistics &; Data Analysis*, 53(11), 3735–3745.

Kipf, T. N. & Welling, M. (2016). Semi-supervised classification with graph convolutional networks.

Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., & Inman, D. J. (2021). 1d convolutional neural networks and applications: A survey. *Mech. Syst. Signal Process.*, 151, 107398.

Preview Sample

Kline, S. J., Coles, D. E., & Hirst, E. (1969). *Computation of turbulent boundary layers – 1968 AFOSR-IFP-Stanford Conference: proceedings held at Stanford University, August 18-25, 1968*. Thermosciences Division, Stanford University.

Knudsen, T., Bak, T., & Svenstrup, M. (2015). Survey of wind farm control—power and fatigue optimization. *Wind Energy*, 18(8), 1333–1351.

Kochkov, D., Smith, J. A., Alieva, A., Wang, Q., Brenner, M. P., & Hoyer, S. (2021). Machine learning accelerated computational fluid dynamics. *arXiv preprint arXiv:2102.01010*.

Kou, J., Le Clainche, S., & Zhang, W. (2018). A reduced-order model for compressible flows with buffeting condition using higher order dynamic mode decomposition with a mode selection criterion. *Phys. Fluids*, 30, 016103.

Kou, J. & Zhang, W. (2019). A hybrid reduced-order framework for complex aeroelastic simulations. *Aerospace science and technology*, 84, 880–894.

Kou, J. & Zhang, W. (2021). Data-driven modeling for unsteady aerodynamics and aeroelasticity. *Progress in Aerospace Sciences*, 125, 100725.

Kouvaritakis, B. & Cannon, M. (2016). *Model Predictive Control: Classical, Robust and Stochastic*. Springer.

Koza, J. (1994). Genetic programming as a means for programming computers by natural selection. *Statistics and Computing*, 4(2).

Kramer, O. (2013). *K-Nearest Neighbors*. Springer Berlin Heidelberg.

Krasnopolsky, V. M. & Fox-Rabinovitz, M. S. (2006a). Complex hybrid models combining deterministic and machine learning components for numerical climate modeling and weather prediction. *Neural Networks*, 19(2), 122–134.

Krasnopolsky, V. M. & Fox-Rabinovitz, M. S. (2006b). A new synergetic paradigm in environmental numerical modeling: Hybrid models combining deterministic and machine learning components. *Ecological Modelling*, 191(1), 5–18.

Krishna, K., Song, Z., & Brunton, S. L. (2022). Finite-horizon, energy-efficient trajectories in unsteady flows. *Proceedings of the Royal Society A*, 478(2258), 20210255.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6), 84–90.

Kühnen, J., Song, B., Scarselli, D., Budanur, N. B., Riedl, M., Willis, A. P., Avila, M., & Hof, N. (2018). Destabilizing turbulence in pipe flow. *Nat. Phys.*, 14(4), 386–390.

Kulfan, B. M. (2008). Universal parametric geometry representation method. *Journal of Aircraft*, 45(1), 142–158.

Kuo, K. & Acharya, R. (2012). *Fundamentals of Turbulent and Multi-Phase Combustion*. Wiley.

Preview Sample

Kutz, J. N. (2017). Deep learning in fluid dynamics. *Journal of Fluid Mechanics*, 814, 1–4.

Kutz, J. N., Brunton, S. L., Brunton, B. W., & Proctor, J. L. (2016a). *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*. Siam.

Kutz, J. N., Fu, X., & Brunton, S. L. (2016b). Multi-resolution dynamic mode decomposition. *SIAM Journal on Applied Dynamical Systems*, 15(2), 713–735.

Kwok, J.-Y. & Tsang, I.-H. (2004). The pre-image problem in kernel methods. *IEEE Transactions on Neural Networks*, 15(6), 1517–1525.

La Cava, W., Orzechowski, P., Burlacu, B., de França, F. O., Virgolin, M., Jin, Y., Kommenda, M., & Moore, J. H. (2021). Contemporary symbolic regression methods and their relative performance. *arXiv preprint arXiv:2107.14351*.

Labahn, J., Wu, H., Coriton, B., Frank, J. H., & Ihme, M. (2018). Data assimilation using high-speed measurements and les to examine local extinction events in turbulent flames. *Proceedings of the Combustion Institute*.

Labahn, J., Wu, H., Harris, S. R., Coriton, B., Frank, J. H., & Ihme, M. (2019). Ensemble kalman filter for assimilating experimental data into large-eddy simulations of turbulent flows. *Flow, Turbulence and Combustion*, 104.

Lagaris, I. E., Likas, A., & Fotiadis, D. I. (1998). Artificial neural networks for solving ordinary and partial differential equations. *IEEE transactions on neural networks*, 9(5), 987–1000.

Lakshmivarahan, S., Lewis, J. M., & Jabrzemski, R. (2017). *Forecast error correction using dynamic data assimilation*. Springer, Switzerland.

Lassila, T., Manzoni, A., Quarteroni, A., & Rozza, G. (2014). Model order reduction in fluid dynamics: Challenges and perspectives. *Reduced Order Methods for Modeling and Computational Reduction. MS & A - Model. Sim. Appl., Springer*, (pp. 235–273).

Laval, J.-P. & Marquillie, M. (2011). Direct numerical simulations of converging–diverging channel flow. In *Progress in wall turbulence: understanding and modeling* (pp. 203–209). Springer.

Law, K., Stuart, A., & Zygalakis, K. (2015). *Data assimilation*. Springer.

Lazpita, E., Martínez-Sánchez, Á., Corrochano, A., Hoyas, S., Le Clainche, S., & Vinuesa, R. (2022). On the generation and destruction mechanisms of arch vortices in urban fluid flows. *Phys. Fluids*, 34(5), 051702.

Le Clainche, S. (2019a). Prediction of the optimal vortex in synthetic jets. *Energies*, 12 (9).

Le Clainche, S. (2019b). Prediction of the optimal vortex in synthetic jets. *Energies*, 12(9), 1635–1655.

Le Clainche, S. & Ferrer, E. (2018). A reduced order model to predict transient flows around straight bladed vertical axis wind turbines. *Energies*, 11, 566–578.

Le Clainche, S., Han, Z., & Ferrer, E. (2019). An alternative method to study cross-flow instabilities based on high order dynamic mode decomposition. *Phys. Fluids*, 31(9), 094101.

Preview Sample

Le Clainche, S., Izvassarov, D., Rosti, M., Brandt, L., & Tammisola, O. (2020a). Coherent structures in the turbulent flow of an elastoviscoplastic fluid. *J. Fluid Mech.*, (pp.Ã5).

Le Clainche, S., Lorente, L., & Vega, J. (2018). Wind predictions upstream wind turbines from a lidar database. *Energies*, 11 (3), 543.

Le Clainche, S., Lorente, L., & Vega, J. (2018). Wind predictions upstream wind turbines from a lidar database. *Energies*, 11(3), 543.

Le Clainche, S., Moreno-Ramos, R., Taylor, P., & Vega, J. M. (2018). A new robust method to study flight flutter testing. *J. Aircr.*

Le Clainche, S., Pérez, J., & Vega, J. (2020b). New method to capture traveling waves in flow passing a wind turbine. *Aerosp. Sci. Tech.*, 105, 105920.

Le Clainche, S., Rosti, M., & Brandt, L. (2022). A data-driven model based on modal decomposition: application to the turbulent channel flow over an anisotropic porous wall. *J. Fluid Mech.*, 939.

Le Clainche, S., Sastre, F., Velazquez, A., & Vega, J. M. (2017a). Higher order dynamic mode decomposition applied to study flow structures in noisy piv experimental data. *AIAA '17-3304 - Proc. 47th AIAA Fluid Dyn. Conf., June 5-9, Denver, CO, USA*.

Le Clainche, S. & Vega, J. M. (2017a). Higher order dynamic mode decomposition. *SIAM J. Appl. Dyn. Sys.*, 16 (2), 882–925.

Le Clainche, S. & Vega, J. M. (2017b). Higher order dynamic mode decomposition to identify and extrapolate flow patterns. *Phys. Fluids*, 29 (8), 084102.

Le Clainche, S., Vega, J. M., & Soria, J. (2017b). Higher order dynamic mode decomposition of noisy experimental data: the flow structure of a zero-net-mass-flux jet. *Exp. Therm. Fluid Sci.*, 88, 336–353.

Leclercq, C., Demourant, F., Poussot-Vassal, C., & Sipp, D. (2019). Linear iterative method for closed-loop control of quasiperiodic flows. *Journal of Fluid Mechanics*, 868, 26–65.

Lee, J. A., Lendasse, A., & Verleysen, M. (2002). Curvilinear distance analysis versus isomap. In *European Symposium on Aritficial Neural Networks* (pp. 185–192). Bruges (Belgium).

Lee, J. H. (2011). Model predictive control: Review of the three decades of development. *International Journal of Control, Automation and Systems*, 9, 415–424.

Lee, J.-W. & Oh, J.-H. (1997). Hybrid learning of mapping and its Jacobian in multilayer neural networks. *Neural Computation*, 9(5), 937–958.

Lee, K. & Carlberg, K. T. (2020). Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. *Journal of Computational Physics*, 404, 108973.

Lee, S. & You, D. (2019). Data-driven prediction of unsteady flow over a circular cylinder using deep learning. *Journal of Fluid Mechanics*, 879, 217–254.

Leibig, C., Allken, V., Ayhan, M. S., Berens, P., & Wahl, S. (2017). Leveraging uncertainty information from deep neural networks for disease detection. *Scientific Reports*, 7(1), 1–14.

Lewis, F. L., Vrabie, D., & Syrmos, V. L. (2012). *Optimal control*. John Wiley & Sons.

Preview Sample

Lewis, J. M., Lakshmivarahan, S., & Dhall, S. (2006). *Dynamic data assimilation: a least squares approach*, volume 104. Cambridge University Press, Cambridge.

Li, K., Kou, J., & Zhang, W. (2019). Deep neural network for unsteady aerodynamic and aeroelastic modeling across multiple mach numbers. *Nonlinear Dynamics*, 96(3), 2157–2177.

Li, Q., Dietrich, F., Bollt, E. M., & Kevrekidis, I. G. (2017). Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the Koopman operator. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(10), 103111.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2020). Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., & Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.

Limón, D., Alamo, T., Salas, F., & Camacho, E. F. (2006). On the stability of constrained MPC without terminal constraint. *IEEE transactions on automatic control*, 51(5), 832–836.

Ling, J., Jones, R., & Templeton, J. (2016a). Machine learning strategies for systems with invariance properties. *Journal of Computational Physics*, 318, 22–35.

Ling, J., Kurzawski, A., & Templeton, J. (2016b). Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics*, 807, 155–166.

Ling, J. & Templeton, J. (2015). Evaluation of machine learning algorithms for prediction of regions of high reynolds averaged navier stokes uncertainty. *Physics of Fluids*, 27(8), 085103.

Liu, D., Litvinenko, A., Schillings, C., & Schulz, V. (2017). Quantification of airfoil geometry-induced aerodynamic uncertainties—comparison of approaches. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1), 334–352.

Liu, J., Huang, J., Zhou, Y., Li, X., Ji, S., Xiong, H., & Dou, D. (2022). From distributed machine learning to federated learning: A survey. *Knowledge and Information Systems*, (pp. 1–33).

Liu, W., Fang, J., Rolfo, S., Moulinec, C., & Emerson, D. R. (2021). An iterative machine-learning framework for rans turbulence modeling. *International Journal of Heat and Fluid Flow*, 90, 108822.

Ljung, L. (1999). *System identification*. Wiley Online Library.

Loiseau, J.-C. (2020). Data-driven modeling of the chaotic thermal convection in an annular thermosyphon. *Theoretical and Computational Fluid Dynamics*, 34(4), 339–365.

Loiseau, J.-C. & Brunton, S. L. (2018). Constrained sparse Galerkin regression. *Journal of Fluid Mechanics*, 838, 42–67.

Loiseau, J.-C., Noack, B. R., & Brunton, S. L. (2018a). Sparse reduced-order modeling: sensor-based dynamics to full-state estimation. *Journal of Fluid Mechanics*, 844, 459–490.

Preview Sample

Loiseau, J. C., Noack, B. R., & Brunton, S. L. (2018b). Sparse reduced-order modelling: sensor-based dynamics to full-state estimation. *Journal of Fluid Mechanics*, 844, 459–490.

Lopez-Martin, M., Le Clainche, S., & Carro, B. (2021). Model-free short-term fluid dynamics estimator with a deep 3d-convolutional neural network. *Exp. Syst. Appl.*, 177, 114924.

Lorente, L., Vega, J., & Velazquez, A. (2013). Efficient computation of the pod manifold containing the information required to generate a multi-parameter aerodynamic database. *Aerosp. Sci. Tech.*, 25(1), 152–160.

Lorenz, E. N. (1963). Deterministic nonperiodic flow. *Journal of the atmospheric sciences*, 20(2), 130–141.

Lorenz, E. N. (1969). Atmospheric predictability as revealed by naturally occurring analogues. *Journal of the Atmospheric sciences*, 26(4), 636–646.

Losse, N. R., King, R., Zengl, M., Rist, U., & Noack, B. R. (2011). Control of tollmien–schlichting instabilities by finite distributed wall actuation. *Theoretical and Computational Fluid Dynamics*, 25, 167–178.

Lu, F., Tu, X., & Chorin, A. J. (2017). Accounting for model error from unresolved scales in ensemble Kalman filters by stochastic parameterization. *Monthly Weather Review*, 145(9), 3709–3723.

Lu, L., Jin, P., Pang, G., Zhang, Z., & Karniadakis, G. E. (2021a). Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3), 218–229.

Lu, L., Meng, X., Mao, Z., & Karniadakis, G. E. (2021b). Deepxde: A deep learning library for solving differential equations. *SIAM review*, 63(1), 208–228.

Lu, Z., Pathak, J., Hunt, B., Girvan, M., Brockett, R., & Ott, E. (2017). Reservoir observers: Model-free inference of unmeasured variables in chaotic systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(4), 041102.

Luchini, P. & Bottaro, A. (2014). Adjoint equations in stability analysis. *Ann. Rev. Fluid*, 46, 493–517.

Lukoševičius, M. (2012). A practical guide to applying echo state networks. In *Neural networks: Tricks of the trade* (pp. 659–686). Springer.

Lukoševičius, M. & Jaeger, H. (2009). Reservoir computing approaches to recurrent neural network training. *Comput. Sci. Rev.*, 3(3), 127–149.

Lumley, J. (1970). *Stochastic Tools in Turbulence*. New York: Academic Press.

Lumley, J. L. (1967). The structure of inhomogeneous turbulent flows. *In: Yaglam, A.M., Tatarsky, V.I. (eds.) Int. Colloq. Fine Scale Atmos. & Radio Wave Propag., Doklady Akad. Nauk SSSR, Nauka, Moscow.*

Lumley, J. L. (1978). Computational modeling of turbulent flows. *Advances in applied mechanics*, 18(123), 213.

Lumley, J. L. & Poje, A. (1997). Low-dimensional models for flows with density fluctuations. *Physics of Fluids*, 9(7), 2023–2031.

Preview Sample

Luo, F.-M., Xu, T., Lai, H., Chen, X.-H., Zhang, W., & Yu, Y. (2022). A survey on model-based reinforcement learning.

Lusch, B., Kutz, J. N., & Brunton, S. L. (2018). Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9(1), 4950.

Ma, Z., Ahuja, S., & Rowley, C. W. (2011). Reduced-order models for control of fluids using the eigensystem realization algorithm. *Theor. Comp. Fluid Dyn.*, 25(1-4), 233–247.

Maass, W., Natschläger, T., & Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation*, 14(11), 2531–2560.

Maceda, G. Y. C., Li, Y., Lusseyran, F., Morzyński, M., & Noack, B. R. (2021). Stabilization of the fluidic pinball with gradient-enriched machine learning control. *Journal of Fluid Mechanics*, 917.

Maciejowski, J. M. (2001). *Predictive control with constraints*. Pearson Education.

Mackey, A., Schaeffer, H., & Osher, S. (2014). On the compressive spectral method. *Multiscale Modeling & Simulation*, 12(4), 1800–1827.

Magnussen, B. F. (1981). On the structure of turbulence and a generalized eddy dissipation concept for chemical reaction in turbulent flow. *in: 19th AIAA Aerospace Science Meeting*.

Magri, L. (2019). Adjoint methods as design tools in thermoacoustics. *Applied Mechanics Reviews*, 71(2).

Magri, L., Schmid, P. J., & Moeck, J. P. (2023). Linear flow analysis inspired by mathematical methods from quantum mechanics. *Annual Review of Fluid Mechanics*, 55, 541–574.

Majda, A. J. & Harlim, J. (2012). Physics constrained nonlinear regression models for time series. *Nonlinearity*, 26(1), 201.

Malik, M. R., Isaac, B. J., Coussement, A., Smith, P. J., & Parente, A. (2018). Principal component analysis coupled with nonlinear regression for chemistry reduction. *Combustion and Flame*, 187, 30–41.

Malik, M. R., Obando Vega, P., Coussement, A., & Parente, A. (2021). Combustion modeling using principal component analysis: A posteriori validation on sandia flames d, e and f. *Proceedings of the Combustion Institute*, 38(2), 2635–2643.

Mallat, S. (1999). *A wavelet tour of signal processing*. Academic press.

Mandler, H. & Weigand, B. (2022). A realizable and scale-consistent data-driven non-linear eddy viscosity modeling framework for arbitrary regression algorithms. *International Journal of Heat and Fluid Flow*, 97, 109018.

Mangan, N. M., Brunton, S. L., Proctor, J. L., & Kutz, J. N. (2016). Inferring biological networks by sparse identification of nonlinear dynamics. *IEEE Transactions on Molecular, Biological, and Multi-Scale Communications*, 2(1), 52–63.

Mangan, N. M., Kutz, J. N., Brunton, S. L., & Proctor, J. L. (2017). Model selection for dynamical systems via sparse regression and information criteria. *Proceedings of the Royal Society A*, 473(2204), 1–16.

Preview Sample

Manohar, K., Brunton, B. W., Kutz, J. N., & Brunton, S. L. (2018). Data-driven sparse sensor placement: Demonstrating the benefits of exploiting known patterns. *IEEE Control Systems Magazine*, 38(3), 63–86.

Manohar, K., Brunton, S. L., & Kutz, J. N. (2017). Environmental identification in flight using sparse approximation of wing strain. *Journal of Fluids and Structures*, 70, 162–180.

Mao, Z., Jagtap, A. D., & Karniadakis, G. E. (2020). Physics-informed neural networks for high-speed flows. *Computer Methods in Applied Mechanics and Engineering*, 360, 112789.

Mardt, A., Pasquali, L., Wu, H., & Noé, F. (2018). VAMPnets: Deep learning of molecular kinetics. *Nature Communications*, 9(5).

Margazoglou, G. & Magri, L. (2023). Stability analysis of chaotic systems from data. *Nonlinear Dynamics*, 111(9), 8799–8819.

Margheri, L., Meldi, M., Salvetti, M., & Sagaut, P. (2014). Epistemic uncertainties in RANS model free coefficients. *Computers & Fluids*, 102, 315–335.

Mariotti, A., Salvetti, M., Omrani, S., & Witteween, J. (2016). Stochastic analysis of the impact of freestream conditions on the aerodynamics of a rectangular 5:1 cylinder. *Computers and Fluids*, 136, 170–192.

Marques, P. A., Ahizi, S., & Mendez, M. A. (2024). Real-time data assimilation for the thermodynamic modeling of cryogenic fuel tanks. *Energy*, 302, 131739.

Marra, L., Meilán-Vila, A., & Discetti, S. (2024). Self-tuning model predictive control for wake flows. *Journal of Fluid Mechanics*, 983, A26.

Marsland, S., Shapiro, J., & Nehmzow, U. (2002). A self-organising network that grows when required. *Neural Networks*, 15(8-9), 1041–1058.

Martinez, S. L. C., Sastre, F., Vega, J. M., & Angel, V. (2017). Higher order dynamic mode decomposition applied to post-process a limited amount of noisy PIV data. In *47th AIAA Fluid Dynamics Conference*: American Institute of Aeronautics and Astronautics.

Martínez-Sánchez, Á., Lazpita, E., Corrochano, A., Le Clainche, S., Hoyas, S., & Vinuesa, R. (2023a). Data-driven assessment of arch vortices in simplified urban flows. *Int. J. Heat Fluid Flow*, 100, 109101.

Martínez-Sánchez, A., López, E., Le Clainche, S., Lozano-Durán, A., Srivastava, A., & Vinuesa, R. (2023b). Causality analysis of large-scale structures in the flow around a wall-mounted square cylinder. *J. Fluid Mech.*, A1, 967.

Martins, J. R. R. A. & Ning, A. (2021). *Engineering Design Optimization*. Cambridge University Press.

Mata, L., Abadía-Heredia, R., Lopez-Martin, M., Pérez, J. M., & Clainche, S. L. (2023). Forecasting through deep learning and modal decomposition in two-phase concentric jets. *Exp. Syst. Appl.*, 232, 120817.

Maulik, R., San, O., & Jacob, J. D. (2020). Spatiotemporally dynamic implicit large eddy simulation using machine learning classifiers. *Physica D: Nonlinear Phenomena*, 406, 132409.

Preview Sample

Maulik, R., San, O., Jacob, J. D., & Crick, C. (2019a). Sub-grid scale model classification and blending through deep learning. *Journal of Fluid Mechanics*, 870, 784–812.

Maulik, R., San, O., Rasheed, A., & Vedula, P. (2019b). Subgrid modelling for two-dimensional turbulence using neural networks. *Journal of Fluid Mechanics*, 858, 122–144.

Mayne, D., Rawlings, J., Rao, C., & Scokaert, P. (2000). Constrained model predictive control: Stability and optimality. *Automatica*, 36(6), 789–814.

Mayne, D. Q. (2014). Model predictive control: Recent developments and future promise. *Automatica*, 50(12), 2967–2986.

McConaghy, T. (2011). Ffx: Fast, scalable, deterministic symbolic regression technology. In R. Riolo, E. Vladislavleva, & J. H. Moore (Eds.), *Genetic Programming Theory and Practice IX* (pp. 235–260). New York, NY: Springer.

McKay, M. D., Beckman, R. J., & Conover, W. J. (2000). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1), 55–61.

Mendez, C., Le Clainche, S., Moreno-Ramos, R., & Vega, J. (2021). A new automatic, very efficient method for the analysis of flight flutter testing data. *Aerosp. Sci. Tech.*, 114, 106749.

Mendez, M. A. (2022). Statistical treatment, fourier and modal decomposition. *VKI Lecture Series "Fundamentals and recent advances in Particle Image Velocimetry and Lagrangian Particle Tracking", ISBN 978-2-87516-181-9*.

Mendez, M. A. (2023a). Continuous and discrete linear time-invariant systems. In M. A. Mendez, A. Ianiro, B. R. Noack, & S. L. Brunton (Eds.), *Data-Driven Fluid Mechanics: Combining First Principles and Machine Learning* (pp. 61–88). Cambridge University Press.

Mendez, M. A. (2023b). Generalized and multiscale modal analysis. In M. A. Mendez, A. Ianiro, B. R. Noack, & S. L. Brunton (Eds.), *Data-Driven Fluid Mechanics: Combining First Principles and Machine Learning* (pp. 153–181). Cambridge University Press.

Mendez, M. A. (2023c). Linear and nonlinear dimensionality reduction from fluid mechanics to machine learning. *Measurement Science and Technology*, 34(4), 042001.

Mendez, M. A., Balabane, M., & Buchlin, J.-M. (2018a). Multi-scale proper orthogonal decomposition (mPOD). In *AIP Conference Proceedings*.

Mendez, M. A., Balabane, M., & Buchlin, J.-M. (2019). Multi-scale proper orthogonal decomposition of complex fluid flows. *Journal of Fluid Mechanics*, 870, 988–1036.

Mendez, M. A., Hess, D., Watz, B. B., & Buchlin, J.-M. (2020a). Multiscale proper orthogonal decomposition (mpod) of tr-piv data—a case study on stationary and transient cylinder wake flows. *Measurement Science and Technology*, 31(9), 094014.

Mendez, M. A., Hess, D., Watz, B. B., & Buchlin, J.-M. (2020b). Multiscale proper orthogonal decomposition (mPOD) of TR-PIV data—a case study on stationary and transient cylinder wake flows. *Meas. Sci. Tech.*, 31(9), 094014.

Mendez, M. A., Marques, P., Schena, L., Van den Berghe, J., Fiore, M., Ahizi, S., Pino, F., & Poletti, R. (2023). Hands on machine learning. `https://www.vki.ac.be/index.php/events-ls/events/eventdetail/552/-/online-on-site-hands-on-machine-learning-for-fluid-dynamics-2023`. Accessed: 2024-01-16.

Preview Sample

Mendez, M. A., Raiola, M., Masullo, A., Discetti, S., Ianiro, A., Theunissen, R., & Buchlin, J.-M. (2017). Pod-based background removal for particle image velocimetry. *Experimental Thermal and Fluid Science*, 80, 181–192.

Mendez, M. A., Scelzo, M., & Buchlin, J.-M. (2018b). Multiscale modal analysis of an oscillating impinging gas jet. *Experimental Thermal and Fluid Science*, 91, 256–276.

Meng, X. & Karniadakis, G. E. (2020). A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse pde problems. *Journal of Computational Physics*, 401, 109020.

Menter, F. R. (1992). Improved two-equation k-ω turbulence models for aerodynamic flows. *Nasa Sti/recon Technical Report N*, 93, 22809.

MetOffice (2024). The met office.

Meyer, E., Heiberg, A., Rasheed, A., & San, O. (2020a). COLREG-compliant collision avoidance for unmanned surface vehicle using deep reinforcement learning. *IEEE Access*, 8, 165344–165364.

Meyer, E., Robinson, H., Rasheed, A., & San, O. (2020b). Taming an autonomous surface vehicle for path following and collision avoidance using deep reinforcement learning. *IEEE Access*, 8, 41466–41481.

Mezić, I. (2005). Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41(1-3), 309–325.

Mika, S., Schölkopf, B., Smola, A., Müller, K.-R., Scholz, M., & Rätsch, G. (1998). Kernel pca and de-noising in feature spaces. In M. Kearns, S. Solla, & D. Cohn (Eds.), *Advances in Neural Information Processing Systems*, volume 11: MIT Press.

Miller, B. K., Geiger, M., Smidt, T. E., & Noé, F. (2020). Relevance of rotationally equivariant convolutions for predicting molecular properties. *arXiv preprint arXiv:2008.08461*.

Misiti, M., Misiti, Y., Oppenheim, G., & Poggi, J.-M. (2015). *Wavelet Toolbox User's Guide*. Mathworks.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529.

Moerland, T. M., Broekens, J., Plaat, A., & Jonker, C. M. (2022). Model-based reinforcement learning: A survey.

Mohammed, R., Tanoff, M., Smooke, M., Schaffer, A., & Long, M. (1998). Computational and experimental study of a forced, time-varying, axisymmetric, laminar diffusion flame. *Symposium (International) on Combustion*, 27(1), 693–702.

Mohan, A. T., Lubbers, N., Livescu, D., & Chertkov, M. (2020). Embedding hard physical constraints in neural network coarse-graining of 3D turbulence. *arXiv preprint arXiv:2002.00021*.

Mohan, P., Fitzsimmons, N., & Moser, R. D. (2017). Scaling of lyapunov exponents in homogeneous isotropic turbulence. *Physical Review Fluids*, 2(11), 114606.

Molnar, C. (2020). *Interpretable machine learning*. Lulu. com.

Preview Sample

Monnier, B., Goudarzi, S. A., Vinuesa, R., & Wark, C. (2018). Turbulent structure of a simplified urban fluid flow studied through stereoscopic particle image velocimetry. *Boundary Layer Meteorol.*, 166, 239–268.

Mons, V., Chassaing, J.-C., Gomez, T., & Sagaut, P. (2016). Reconstruction of unsteady viscous flows using data assimilation schemes. *Journal of Computational Physics*, 316, 255–280.

Moreno, A., Jarzabek, A., Perales, J., & Vega, J. (2016). Aerodynamic database reconstruction via gappy high order singular value decomposition. *Aerosp. Sci. Tech.*, 52, 115–128.

Morton, J., Jameson, A., Kochenderfer, M. J., & Witherden, F. (2018). Deep dynamical modeling and control of unsteady fluid flows. *Advances in Neural Information Processing Systems*, 31.

Moser, R. D., Kim, J., & Mansour, N. N. (1999). Direct numerical simulation of turbulent channel flow up to re τ= 590. *Physics of fluids*, 11(4), 943–945.

Mou, C., Koc, B., San, O., Rebholz, L. G., & Iliescu, T. (2021). Data-driven variational multiscale reduced order models. *Computer Methods in Applied Mechanics and Engineering*, 373, 113470.

Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. Adaptive computation and machine learning series. Cambridge, Massachusetts and London, England: MIT press.

Muñoz, E., Dave, H., D'Alessio, G., Bontempi, G., Parente, A., & Le Clainche, S. (2023). Extraction and analysis of flow features in planar synthetic jets using different machine learning techniques. *Phys. Fluids*, 35(9), 094107.

Muñoz, E. & Le Clainche, S. (2022). On the topology patterns and symmetry breaking in two planar synthetic jets. *Phys. Fluids*, 34(2).

Nair, A. G. & Taira, K. (2015). Network-theoretic approach to sparsified discrete vortex dynamics. *Journal of Fluid Mechanics*, 768, 549–571.

Nair, A. G., Yeh, C.-A., Kaiser, E., Noack, B. R., Brunton, S. L., & Taira, K. (2018). Cluster-based feedback control of turbulent post-stall separated flows. *arXiv preprint arXiv:1809.07220*, 875, 345–375.

Nair, D. S., Hochgeschwender, N., & Olivares-Mendez, M. A. (2022). Maximum likelihood uncertainty estimation: Robustness to outliers.

Nastac, G., Labahn, J., Magri, L., & Ihme, M. (2017). Lyapunov exponent as a metric for assessing the dynamic content and predictability of large-eddy simulations. *Physical Review Fluids*, 2(9), 094606.

Navon, I. M. (2009). Data assimilation for numerical weather prediction: a review. In *Data assimilation for atmospheric, oceanic and hydrologic applications* (pp. 21–65). Springer.

Nelles, O. (2001). *Nonlinear System Identification*. Springer Berlin Heidelberg.

Nguyen-Tuong, D., Seeger, M., & Peters, J. (2009). Model learning with local gaussian process regression. *Advanced Robotics*, 23(15), 2015–2034.

Preview Sample

Ni, A. & Wang, Q. (2017). Sensitivity analysis on chaotic dynamical systems by non-intrusive least squares shadowing (nilss). *Journal of Computational Physics*, 347, 56–77.

Noack, B. R., Afanasiev, K., Morzynski, M., Tadmor, G., & Thiele, F. (2003). A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *Journal of Fluid Mechanics*, 497, 335–363.

Noack, B. R., Ehlert, A., Nayeri, C. N., & Morzynski, M. (2023). Analysis, modeling, and control of the cylinder wake. In M. A. Mendez, A. Ianiro, B. R. Noack, & S. L. Brunton (Eds.), *Data-Driven Fluid Mechanics* (pp. 3–19). Cambridge University Press.

Noack, B. R., Morzynski, M., & Tadmor, G. (2011). *Reduced-order modelling for flow control*, volume 528. Springer Science & Business Media.

Noack, B. R. & Morzyński, M. (2017). *The fluidic pinball—a toolkit for multiple-input multiple-output flow control (version 1.0)*. Technical report, Tech. Rep. 02/2017. Chair of Virtual Engineering, Poznan University of Technology.

Nocedal, J. & Wright, S. J. (2006). *Numerical Optimization*. Springer New York.

Novati, G., de Laroussilhe, H. L., & Koumoutsakos, P. (2021). Automating turbulence modelling by multi-agent reinforcement learning. *Nature Machine Intelligence*, 3(1), 87–96.

Novati, G., Mahadevan, L., & Koumoutsakos, P. (2019). Controlled gliding and perching through deep-reinforcement-learning. *Physical Review Fluids*, 4(9), 093902.

Oh, H., Amici, R., Bomarito, G., Zhe, S., Kirby, R., & Hochhalter, J. (2023). Genetic programming based symbolic regression for analytical solutions to differential equations. *arxiv*.

Olazaran, M. (1993). A sociological history of the neural network controversy. In *Advances in Computers* (pp. 335–425). Elsevier.

Oldewurtel, F., Parisio, A., Jones, C. N., Gyalistras, D., Gwerder, M., Stauch, V., Lehmann, B., & Morari, M. (2012). Use of model predictive control and weather forecasts for energy efficient building climate control. *Energy and buildings*, 45, 15–27.

Oliver, T. A. & Moser, R. D. (2011). Bayesian uncertainty quantification applied to RANS turbulence models. *Journal of Physics: Conference Series*, 318, 042032.

Oppenheim, A. V. & Schafer, R. W. (2009). *Discrete-Time Signal Processing*. Prentice-Hall Signal Processing Series. Pearson;.

Oseledets, V. I. (1968). A multiplicative ergodic theorem. characteristic lyapunov, exponents of dynamical systems. *Trudy Moskovskogo Matematicheskogo Obshchestva*, 19, 179–210.

Otto, S. E. & Rowley, C. W. (2019). Linearly-recurrent autoencoder networks for learning dynamics. *SIAM Journal on Applied Dynamical Systems*, 18(1), 558–593.

Ouyang, L., Wang, F.-Y., Tian, Y., Jia, X., Qi, H., & Wang, G. (2023). Artificial identification: a novel privacy framework for federated learning based on blockchain. *IEEE Transactions on Computational Social Systems*.

Preview Sample

Oyelade, O. N. & Ezugwu, A. E. (2022). A novel wavelet decomposition and transformation convolutional neural network with data augmentation for breast cancer detection using digital mammogram. *Scientific Reports*, 12(1).

Özalp, E., Margazoglou, G., & Magri, L. (2023). Physics-informed long short-term memory for forecasting and reconstruction of chaos. In *Computational Science – ICCS 2023* (pp. 382–389).

Ozoliņš, V., Lai, R., Caflisch, R., & Osher, S. (2013). Compressed modes for variational problems in mathematics and physics. *Proceedings of the National Academy of Sciences*, 110(46), 18368–18373.

Paehler, L., Callaham, J., Ahnert, S., Adams, N., & Brunton, S. (2023). Hydrogym: A reinforcement learning control framework for fluid dynamics. *Bulletin of the American Physical Society*.

Pan, C., Xue, D., & Wang, J. (2015). On the accuracy of dynamic mode decomposition in estimating instability of wave packet. *Experiments in Fluids*, 56(8).

Pan, S. & Duraisamy, K. (2020). Physics-informed probabilistic learning of linear embeddings of nonlinear dynamics with guaranteed stability. *SIAM Journal on Applied Dynamical Systems*, 19(1), 480–509.

Pang, G., Lu, L., & Karniadakis, G. E. (2019). fpinns: Fractional physics-informed neural networks. *SIAM Journal on Scientific Computing*, 41(4), A2603–a2626.

Pao, H.-T. (2008). A comparison of neural network and multiple regression analysis in modeling capital structure. *Expert Systems with Applications*, 35, 720–727.

Parekh, J., Bekemeyer, P., Helm, S., François, D. G., & Grabe, C. (2023). Laminar airfoil design under uncertainties using the dlr γ transition model. In *Deutscher Luft- und Raumfahrtkongress*.

Parente, A., Sutherland, J., Dally, B., Tognotti, L., & Smith, P. (2011). Investigation of the mild combustion regime via principal component analysis. *Proceedings of the Combustion Institute*, 33(2), 3333–3341.

Parente, A. & Sutherland, J. C. (2013a). Principal component analysis of turbulent combustion data: Data pre-processing and manifold sensitivity. *Combust. Flame.*, 160(2), 340–350.

Parente, A. & Sutherland, J. C. (2013b). Principal component analysis of turbulent combustion data: Data pre-processing and manifold sensitivity. *Combustion and flame*, 160(2), 340–350.

Parish, E. J. & Duraisamy, K. (2016). A paradigm for data-driven predictive modeling using field inversion and machine learning. *Journal of Computational Physics*, 305, 758–774.

Parussini, L. & Pediroda, V. (2007). Fictitious domain with least-squares spectral element method to explore geometric uncertainties by non-intrusive polynomial chaos method. *Computer modeling in engineering and science*, 22(1), 41–63.

Pastur, L. R., Deng, N., Morzyński, M., & Noack, B. R. (2019). Reduced-order modeling of the fluidic pinball. In *Chaotic Modeling and Simulation International Conference* (pp. 205–213). Cham: Springer Springer International Publishing.

Preview Sample

Pathak, J., Hunt, B., Girvan, M., Lu, Z., & Ott, E. (2018a). Model-free prediction of large spatiotemporally chaotic systems from data: a reservoir computing approach. *Physical review letters*, 120(2), 024102.

Pathak, J., Lu, Z., Hunt, B. R., Girvan, M., & Ott, E. (2017). Using machine learning to replicate chaotic attractors and calculate lyapunov exponents from data. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(12), 121102.

Pathak, J., Wikner, A., Fussell, R., Chandra, S., Hunt, B. R., Girvan, M., & Ott, E. (2018b). Hybrid forecasting of chaotic processes: Using machine learning in conjunction with a knowledge-based model. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 28(4), 041101.

Pawar, S., Ahmed, S. E., San, O., & Rasheed, A. (2020a). An evolve-then-correct reduced order model for hidden fluid dynamics. *Mathematics*, 8(4), 570.

Pawar, S., Ahmed, S. E., San, O., Rasheed, A., & Navon, I. M. (2020b). Long short-term memory embedded nudging schemes for nonlinear data assimilation of geophysical flows. *Physics of Fluids*, 32(7), 076606.

Pawar, S., Rahman, S. M., Vaddireddy, H., San, O., Rasheed, A., & Vedula, P. (2019). A deep learning enabler for nonintrusive reduced order modeling of fluid flows. *Physics of Fluids*, 31(8), 085101.

Pawar, S. & San, O. (2021). Data assimilation empowered neural network parametrizations for subgrid processes in geophysical flows. *Physical Review Fluids*, 6(5), 050501.

Pawar, S., San, O., Rasheed, A., & Navon, I. M. (2021). A nonintrusive hybrid neural-physics modeling of incomplete dynamical systems: Lorenz equations. *GEM-International Journal on Geomathematics*, 12(1), 1–31.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

Peerenboom, K., Parente, A., Kozák, T., Bogaerts, A., & Degrez, G. (2015). Dimension reduction of non-equilibrium plasma kinetic models using principal component analysis. *Plasma Sources Science and Technology*, 24(2), 025004.

Peng, W., Zhang, J., Zhou, W., Zhao, X., Yao, W., & Chen, X. (2021). Idrlnet: A physics-informed neural network library. *ArXiv*, abs/2107.04320.

Penland, C. (1996). A stochastic model of IndoPacific sea surface temperature anomalies. *Physica D: Nonlinear Phenomena*, 98(2-4), 534–558.

Penland, C. & Magorian, T. (1993). Prediction of niño 3 sea surface temperatures using linear inverse modeling. *Journal of Climate*, 6(6), 1067–1076.

Pepiot-Desjardins, P. & Pitsch, H. (2008). An efficient error-propagation-based reduction method for large chemical kinetic mechanisms. *Combustion and Flame*, 154(1-2), 67–81.

Perot, B. & Moin, P. (1996). *A new approach to turbulence modeling.* Technical report, Proceedings of Summer Research Program, Center of Turbulence Research, Stanford University, Stanford, CA, USA.

Preview Sample

Pikovsky, A. & Politi, A. (2016). *Lyapunov exponents: a tool to explore complex dynamics*. Cambridge University Press.

Pino, F., Schena, L., Rabault, J., & Mendez, M. A. (2023). Comparative analysis of machine learning methods for active flow control. *Journal of Fluid Mechanics*, 958, A39.

Piomelli, U. & Balaras, E. (2002). Wall-layer models for large-eddy simulations. *Annual Review of Fluid Mechanics*, 34(1), 349–374.

Platteeuw, P. D. A., Loeven, G. J. A., & Bijl, H. (2008). Uncertainty quantification applied to the k–ϵ model of turbulence using the probabilistic collocation method. In *10th AIAA Non-Deterministic Approaches Conference*. Paper no.: 2008-2150.

Poletti, R., Schena, L., Ninni, D., & Mendez, M. A. (2024). Modulo: a python toolbox for data-driven modal decomposition. *Journal of Open Software*.

Polyak, B. (1964). Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5), 1–17.

Pope, S. (1975). A more general effective-viscosity hypothesis. *Journal of Fluid Mechanics*, 72(2), 331–340.

Pope, S. B. (2000). *Turbulent flows*. Cambridge: Cambridge university press.

Pope, S. B. (2013). Small scales, many species and the manifold challenges of turbulent combustion. *Proc. Combust. Inst.*, 34, 1–31.

Poroseva, S. V., Hussaini, M. Y., & Woodruff, S. L. (2006). Improving the predictive capability of turbulence models using evidence theory. *AIAA Journal*, 44(6), 1220–1228.

Pratama, D. A., Abo-Alsabeh, R. R., Bakar, M. A., Salhi, A., & Ibrahim, N. F. (2023). Solving partial differential equations with hybridized physic-informed neural network and optimization approach: Incorporating genetic algorithms and l-bfgs for improved accuracy. *Alexandria Engineering Journal*, 77, 205–226.

Prince, S. J. (2023). *Understanding Deep Learning*. MIT press.

Proctor, J. L., Brunton, S. L., Brunton, B. W., & Kutz, J. N. (2014). Exploiting sparsity and equation-free architectures in complex systems (invited review). *The European Physical Journal Special Topics*, 223(13), 2665–2684.

Propoi, A. (1963). Use of linear programming methods for synthesizing sampled-data automatic systems. *Automn. Remote Control*, 24(7), 837–844.

Prudencio, E. & Cheung, S. H. (2012). Parallel adaptive multilevel sampling algorithms for the bayesian analysis of mathematical models. *International Journal for Uncertainty Quantification*, 2(3), 215–237.

Psichogios, D. C. & Ungar, L. H. (1992). A hybrid neural network-first principles approach to process modeling. *AIChE Journal*, 38(10), 1499–1511.

Publications Office of the European Union (2011). *Flightpath 2050 : Europe's vision for aviation : maintaining global leadership and serving society's needs.* Publications Office of the European Union.

Review Sample

Puterman, M. L. (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Qin, S. J. & Badgwell, T. A. (1997). An overview of industrial model predictive control technology. In *AIChE Symposium Series*, volume 93 (pp. 232–256).: New York, NY: American Institute of Chemical Engineers, 1971-c2002.

Quarteroni, A. & Rozza, G. (2013). *Reduced Order Methods for Modeling and Computational Reduction*, volume 9 of *MS&A – Modeling, Simulation & Appplications*. Springer.

Rabault, J., Kuchta, M., Jensen, A., Réglade, U., & Cerardi, N. (2019). Artificial neural networks trained through deep reinforcement learning discover control strategies for active flow control. *Journal of fluid mechanics*, 865, 281–302.

Rabault, J., Ren, F., Zhang, W., Tang, H., & Xu, H. (2020). Deep reinforcement learning in fluid mechanics: A promising method for both active flow control and shape optimization. *J Hydrodynam B .*, 32, 234–246.

Racca, A. (2023). *Neural networks for the prediction of chaos and turbulence*. PhD thesis, University of Cambridge.

Racca, A. & Magri, L. (2021). Robust optimization and validation of echo state networks for learning chaotic dynamics. *Neural Networks*, 142, 252–268.

Racca, A. & Magri, L. (2022). Data-driven prediction and control of extreme events in a chaotic flow. *Physical Review Fluids*, 7(10), 104402.

Raffel, M., Willert, C. E., Scarano, F., Kähler, C. J., Wereley, S. T., & Kompenhans, J. (2018). *Particle Image Velocimetry*. Springer, 3rd edition.

Rahman, S. M., Pawar, S., San, O., Rasheed, A., & Iliescu, T. (2019). Nonintrusive reduced order modeling framework for quasigeostrophic turbulence. *Physical Review E*, 100(5), 053306.

Raiola, M., Discetti, S., & Ianiro, A. (2015). On PIV random error minimization with optimal POD-based low-order reconstruction. *Experiments in Fluids*, 56(4).

Raissi, M., Perdikaris, P., & Karniadakis, G. (2019a). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378, 686–707.

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019b). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378, 686–707.

Raissi, M., Yazdani, A., & Karniadakis, G. E. (2020). Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481), 1026–1030.

Ranzi, E., Frassoldati, A., Stagni, A., Pelucchi, M., Cuoci, A., & Faravelli, T. (2014). Reduced kinetic schemes of complex reaction systems: fossil and biomass derived transportation fuels. *International Journal of Chemical Kinetics*, 46, 512–542.

Rasheed, A., San, O., & Kvamsdal, T. (2020). Digital twin: Values, challenges and enablers from a modeling perspective. *IEEE Access*, 8, 21980–22012.

Preview Sample

Rasmussen, C. E. (2004). *Gaussian Processes in Machine Learning*. Berlin, Heidelberg: Springer Berlin Heidelberg.

Rasmussen, C. E., Williams, C. K., et al. (2006). *Gaussian processes for machine learning*, volume 1. Springer.

Rasmussen, C. E. & Williams, C. K. I. (2005). *Gaussian Processes for Machine Learning*. Adaptive computation and machine learning. Cambridge, Mass.: MIT Press Ltd, 3. print edition.

Ratz, M., Fiorini, D., Simonini, A., Cierpka, C., & Mendez, M. A. (2022). Analysis of an unsteady quasi-capillary channel flow with time-resolved PIV and RBF-based super-resolution. *Journal of Coatings Technology and Research*.

Ratz, M. & Mendez, M. A. (2024). A meshless and binless approach to compute statistics in 3d ensamble ptv.

Rawlings, J. B., Mayne, D. Q., & Diehl, M. (2017). *Model predictive control: theory, computation, and design*, volume 2. Nob Hill Publishing Madison, WI.

Ray, J., Dechant, L., Lefantzi, S., Ling, J., & Arunajatesan, S. (2018). Robust Bayesian calibration of k–ε model for compressible jet-in-crossflow simulations. *AIAA Journal*. In press.

Recht, B. (2019). A tour of reinforcement learning: The view from continuous control. *Annu. Rev. Control Robot. Auton. Syst.*, 2, 253–279.

Reddy, G., Wong-Ng, J., Celani, A., Sejnowski, T. J., & Vergassola, M. (2018). Glider soaring via reinforcement learning in the field. *Nature*, 562(7726), 236–239.

Reichel, L. & Trefethen, L. N. (1992). Eigenvalues and pseudo-eigenvalues of toeplitz matrices. *Linear Algebra and its Applications*, 162–164, 153–185.

Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., et al. (2019). Deep learning and process understanding for data-driven Earth system science. *Nature*, 566(7743), 195–204.

Ren, F., Hu, H.-b., & Tang, H. (2020). Active flow control using machine learning: A brief review. *Journal of Hydrodynamics*, 32(2), 247–253.

Ren, Z. & Pope, S. (2008). Second-order splitting schemes for a class of reactive systems. *Journal of Computational Physics*, 227(17), 8165–8176.

Richalet, J., Rault, A., Testud, J., & Papon, J. (1978). Model predictive heuristic control. *Automatica (journal of IFAC)*, 14(5), 413–428.

Rincón, M. J., Amarloo, A., Reclari, M., Yang, X. I., & Abkar, M. (2023). Progressive augmentation of reynolds stress tensor models for secondary flow prediction by computational fluid dynamics driven surrogate optimisation. *International Journal of Heat and Fluid Flow*, 104, 109242.

Rivest, R. L., Cormen, T. H. D. C., Leiserson, C. E. M., & Stein, C. C. U. (2009). *Introduction to Algorithms*. MIT Press Ltd.

Roberts, A. J. (2014). *Model emergent dynamics in complex systems*. Siam.

Preview Sample

Rodi, W. (1976). A new algebraic relation for calculating the Reynolds stresses. *Gesellschaft Angewandte Mathematik und Mechanik Workshop Paris France*, 56, 219.

Rogowski, M., Yeung, B. C., Schmidt, O. T., Maulik, R., Dalcin, L., Parsani, M., & Mengaldo, G. (2024). Unlocking massively parallel spectral proper orthogonal decompositions in the pyspod package. *Computer Physics Communications*, 302, 109246.

Romero, A., Song, Y., & Scaramuzza, D. (2024). Actor-critic model predictive control. In *2024 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 14777–14784).: IEEE.

Rosenblatt, F. (1957). *The Perceptron—a perceiving and recognizing automaton.* Technical Report Report 85-460-1, Cornell Aeronautical Laboratory.

Rossiter, J. A. (2017). *Model-based predictive control: a practical approach.* CRC press.

Routray, A., Osuri, K. K., Pattanayak, S., & Mohanty, U. C. (2016). Introduction to data assimilation techniques and ensemble kalman filter. In *Advanced Numerical Modeling and Data Assimilation Techniques for Tropical Cyclone Prediction* (pp. 307–330). Springer Netherlands.

Roweis, S. T. & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500), 2323–2326.

Rowley, C. W. (2005). Model reduction for fluids, using balanced proper orthogonal decomposition. *Int. J. Bifurcation Chaos*, 15(03), 997–1013.

Rowley, C. W., Colonius, T., & Murray, R. M. (2004). Model reduction for compressible flows using POD and Galerkin projection. *Physica D*, 189, 115–129.

Rowley, C. W. & Dawson, S. T. (2017). Model reduction for flow analysis and control. *Annual Review of Fluid Mechanics*, 49, 387–417.

Rowley, C. W., Mezić, I., Bagheri, S., Schlatter, P., & Henningson, D. (2009). Spectral analysis of nonlinear flows. *J. Fluid Mech.*, 645, 115–127.

Rudy, S. H., Brunton, S. L., Proctor, J. L., & Kutz, J. N. (2017). Data-driven discovery of partial differential equations. *Science Advances*, 3(e1602614).

Ruelle, D. (1979). Ergodic theory of differentiable dynamical systems. *Publications Mathématiques de l'Institut des Hautes Études Scientifiques*, 50(1), 27–58.

Rumelhart, D. E. & McClelland, J. L. (1989). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations.* MIT PRess.

Rumsey, C. L., Coleman, G. N., & Wang, L. (2022). In search of data-driven improvements to rans models applied to separated flows. In *AIAA Scitech 2022 Forum* (pp. 0937).

Sabater, C., Bekemeyer, P., & Görtz, S. (2020). Efficient bilevel surrogate approach for optimization under uncertainty of shock control bumps. *AIAA Journal*, 58(12), 5228–5242.

Sacha, D., Sedlmair, M., Zhang, L., Lee, J. A., Peltonen, J., Weiskopf, D., North, S. C., & Keim, D. A. (2017). What you see is what you can change: Human-centered machine learning by interactive visualization. *Neurocomputing*, 268, 164–175.

Preview Sample

Sagaut, P. (2006). *Large Eddy Simulations for Incompressible Flows: An Introduction*. Springer.

Sagaut, P., Deck, S., & Terracol, M. (2013). *Multiscale and Multiresolution Approaches in Turbulence*. Imperial College Press, second edition.

Saïdi, I. B. H., Schmelzer, M., Cinnella, P., & Grasso, F. (2022). CFD-driven symbolic identification of algebraic reynolds-stress models. *Journal of Computational Physics*, 457, 111037.

San, O., Pawar, S., & Rasheed, A. (2022). Variational multiscale reinforcement learning for discovering reduced order closure models of nonlinear spatiotemporal transport systems. *Scientific Reports*, 12(1), 17947.

San, O., Pawar, S., & Rasheed, A. (2023). Decentralized digital twins of complex dynamical systems. *Scientific Reports*, 13(1), 20087.

San, O., Rasheed, A., & Kvamsdal, T. (2021). Hybrid analysis and modeling, eclecticism, and multifidelity computing toward digital twin revolution. *GAMM-Mitteilungen*, 44(2), e202100007.

Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., & Battaglia, P. (2020). Learning to simulate complex physics with graph networks. In *International Conference on Machine Learning*, Icml'20 (pp. 8459–8468).: Pmlr JMLR.org.

Sandberg, R. D. & Zhao, Y. (2022). Machine-learning for turbulence and heat-flux model development: A review of challenges associated with distinct physical phenomena and progress to date. *International Journal of Heat and Fluid Flow*, 95, 108983.

Sandri, M. (1996). Numerical calculation of lyapunov exponents. *Mathematica Journal*, 6(3), 78–84.

Sandu, A., Daescu, D. N., Carmichael, G. R., & Chai, T. (2005). Adjoint sensitivity analysis of regional air quality models. *Journal of Computational Physics*, 204(1), 222–252.

Šarler, B. (2007). From global to local radial basis function collocation method for transport phenomena. In *Advances in Meshfree Techniques* (pp. 257–282). Springer Netherlands.

Sasaki, Y. & Tsubakino, D. (2020). Designs of feedback controllers for fluid flows based on model predictive control and regression analysis. *Energies*, 13(6).

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2009). The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1), 61–80.

Schaeffer, H. (2017). Learning partial differential equations via data discovery and sparse optimization. In *Proc. R. Soc. A*, volume 473 (pp. 20160446).: The Royal Society.

Schaeffer, H., Caflisch, R., Hauck, C. D., & Osher, S. (2013). Sparse dynamics for partial differential equations. *Proceedings of the National Academy of Sciences USA*, 110(17), 6634–6639.

Schaeffer, H. & McCalla, S. G. (2017). Sparse model selection via integral terms. *Physical Review E*, 96(2), 023302.

Schanz, D., Gesemann, S., & Schröder, A. (2016). Shake-The-Box: Lagrangian particle tracking at high particle image densities. *Experiments in Fluids*, 57.

Preview Sample

Schena, L., Marques, P., Poletti, R., Ahizi, S., den Berghe, J. V., & Mendez, M. A. (2023). Reinforcement twinning: from digital twins to model-based reinforcement learning. *Journal of Computational Science*.

Scherl, I., Strom, B., Shang, J. K., Williams, O., Polagye, B. L., & Brunton, S. L. (2020). Robust principal component analysis for particle image velocimetry. *Physical Review Fluids*, 5(054401).

Schiller, U. D. & Steil, J. J. (2005). Analyzing the weight dynamics of recurrent learning algorithms. *Neurocomputing*, 63, 5–23.

Schlegel, M. & Noack, B. R. (2015). On long-term boundedness of galerkin models. *Journal of Fluid Mechanics*, 765, 325–352.

Schlegel, M., Noack, B. R., & Tadmor, G. (2004). *Low-dimensional Galerkin models and control of transitional channel flow*. Technical Report 01/2004, Hermann-Föttinger-Institut für Strömungsmechanik, Technische Universität Berlin, Germany.

Schmelzer, M., Dwight, R. P., & Cinnella, P. (2020). Discovery of algebraic reynolds-stress models using sparse symbolic regression. *Flow, Turbulence and Combustion*, 104(2), 579–603.

Schmid, P. J. (2010). Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656, 5–28.

Schmidt, E. (1907). Zur theorie der linearen und nichtlinearen integralgleichungen. *Mathematische Annalen*, 63(4), 433–476.

Schmidt, M. & Lipson, H. (2009). Distilling free-form natural laws from experimental data. *Science*, 324(5923), 81–85.

Schmidt, M. D., Vallabhajosyula, R. R., Jenkins, J. W., Hood, J. E., Soni, A. S., Wikswo, J. P., & Lipson, H. (2011). Automated refinement and inference of analytical models for metabolic networks. *Physical biology*, 8(5), 055011.

Schmidt, O. T. & Colonius, T. (2020). Guide to spectral proper orthogonal decomposition. *AIAA Journal*, 58(3), 1023–1033.

Schmitt, F. G. (2007). About Boussinesq's turbulent viscosity hypothesis: historical remarks and a direct evaluation of its validity. *Comptes Rendus Mécanique*, 335, 617–627.

Schneiders, J. & Scarano, F. (2016). Dense velocity reconstruction from tomographic PTV with material derivatives. *Experiments in Fluids*, 57.

Schölkopf, B. & Smola, A. J. (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

Schoukens, J. & Ljung, L. (2019). Nonlinear system identification: A user-oriented road map. *IEEE Control Systems*, 39(6), 28–99.

Schröder, A. & Schanz, D. (2023). 3d lagrangian particle tracking in fluid mechanics. *Annual Review of Fluid Mechanics*, 55(1), 511–540.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Preview Sample

Schwamborn, D., Gerhold, T., & Heinrich, R. (2006). The DLR TAU-Code: Recent Applications in Research and Industry. In P. Wesseling (Ed.), *ECCOMAS CFD 2006: Proceedings of the European Conference on Computational Fluid Dynamics* [s. l.]: Delft University of Technology.

Schwenzer, M., Ay, M., Bergs, T., & Abel, D. (2021). Review on model predictive control: An engineering perspective. *The International Journal of Advanced Manufacturing Technology*, 117(5-6), 1327–1349.

Schwer, D., Lu, P., & Green Jr., W. (2003). An adaptive chemistry approach to modeling complex kinetics in reacting flows. *Combustion and Flame*, 133(4), 451–465.

Schölkopf, B., Smola, A., & Müller, K.-R. (1997). Kernel principal component analysis. In *Lecture Notes in Computer Science* (pp. 583–588). Springer Berlin Heidelberg.

Sciacchitano, A., Leclaire, B., & Schröder, A. (2022). Main results of the analysis of the homer lagrangian particle tracking and data assimilation database. In *20th International Symposium on the Application of Laser and Imaging Techniques to Fluid Mechanics*.

Semeraro, O., Bagheri, S., Brandt, L., & Henningson, D. S. (2013). Transition delay in a boundary layer flow using active control. *J. Fluid Mech.*, 731(9), 288–311.

Semeraro, O., Lusseyran, F., Pastur, L., & Jordan, P. (2016). Qualitative dynamics of wavepackets in turbulent jets. *arXiv preprint arXiv:1608.06750*, 2(094605).

Semeraro, O., Pralits, J. O., Rowley, C. W., & Henningson, D. S. (2013). Riccati-less approach for optimal control and estimation: an application to two-dimensional boundary layers. *Journal of Fluid Mechanics*, 731, 394–417.

Shi, Y. & Zhang, K. (2021). Advanced model predictive control framework for autonomous intelligent mechatronic systems: A tutorial overview and perspectives. *Annual Reviews in Control*, 52, 170–196.

Sieber, M., Paschereit, C. O., & Oberleithner, K. (2016). Spectral proper orthogonal decomposition. *Journal of Fluid Mechanics*, 792, 798–828.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., & Riedmiller, M. (2014). Deterministic Policy Gradient Algorithms. In *International Conference on Machine Learning* (pp.–-).

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of go without human knowledge. *nature*, 550(7676), 354–359.

Simonsen, A. & Krogstad, P.-Å. (2005). Turbulent stress invariant analysis: Clarification of existing terminology. *Physics of Fluids*, 17(8), 088103.

Singh, A. P. & Duraisamy, K. (2016). Using field inversion to quantify functional errors in turbulence closures. *Physics of Fluids*, 28, 045110.

Singh, A. P., Medida, S., & Duraisamy, K. (2017). Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils. *AIAA Journal*, 55(7), 2215–2227.

Singh, J., Azamfar, M., Li, F., & Lee, J. (2020). A systematic review of machine learning algorithms for prognostics and health management of rolling element bearings: fundamentals, concepts and applications. *Measurement Science and Technology*, 32(1), 012001.

Preview Sample

Sipp, D. & Schmid, P. J. (2016). Linear closed-loop control of fluid instabilities and noise-induced perturbations: A review of approaches and tools. *Applied Mechanics Reviews*, 68(2), 020801.

Sirovich, L. (1987). Turbulence and the dynamic of coherent structures, parts i–iii. *Q. Appl. Math.*, 45(3), 561.

Sirovich, L. (1991). Analysis of turbulent flows by means of the empirical eigenfunctions. *Fluid Dyn. Res.*, 8, 85–100.

Smith, G. P., Golden, D. M., Frenklach, M., Moriarty, N. W., Eiteneer, B., Goldenberg, M., Bowman, C. T., Hanson, R. K., Song, S., Jr., W. C. G., Lissianski, V. V., & Qin, Z. (1999).

Smith, J. O. (2007). *Mathematics of the Discrete Fourier Transform (DFT): with Audio Applications*. W3K Publishing; 2 edition.

Smith, P. J., Thornhill, G. D., Dance, S. L., Lawless, A. S., Mason, D. C., & Nichols, N. K. (2012). Data assimilation for state and parameter estimation: application to morphodynamic modelling. *Quarterly Journal of the Royal Meteorological Society*, 139(671).

Smola, A. & Scholkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing*, 14(3), 199–222.

Smola, A. J. & Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and computing*, 14(3), 199–222.

Sobester, A., Nair, P., & Keane, A. (2008). Genetic programming approaches for solving elliptic partial differential equations. *IEEE Transactions on Evolutionary Computation*, 12(4), 469–478.

Sobol, I. M. (1979). On the systematic search in a hypercube. *SIAM Journal on Numerical Analysis*, 16(5), 790–793.

Ŝolín, P. (2005). *Partial differential equations and the finite element method*. John Wiley & Sons.

Son, H., Cho, S. W., & Hwang, H. J. (2023). Enhanced physics-informed neural networks with augmented lagrangian relaxation method (al-pinns). *Neurocomputing*, 548, 126424.

Sorokina, M., Sygletos, S., & Turitsyn, S. (2016). Sparse identification for nonlinear optical communication systems: SINO method. *Optics express*, 24(26), 30433–30443.

Spalart, P. (2000). Strategies for turbulence modelling and simulations. *International Journal of Heat and Fluid Flow*, 21, 252–263.

Spalart, P. (2023). An old-fashioned framework for machine learning in turbulence modeling. *arxiv*.

Spalart, P. & Shur, M. (1997). On the sensitization of simple turbulence models to rotation and curvature. *Aerospace Science and Technology*, 1, 297–302.

Spalart, P. R. (2009). Detached-eddy simulation. *Annual Review of Fluid Mechanics*, 41, 181–202.

Preview Sample

Spalart, P. R. (2015). Philosophies and fallacies in turbulence modeling. *Progress in Aerospace Sciences*, 74, 1–15.

Spalart, P. R. & Allmaras, S. R. (1992). A one-equation turbulence model for aerodynamic flows. AIAA Paper 1992-439.

Sperotto, P., Pieraccini, S., & Mendez, M. A. (2022). A Meshless Method to Compute Pressure Fields from Image Velocimetry. *Measurement Science and Technology*, 33.

Sperotto, P., Ratz, M., & Mendez, M. A. (2024). Spicy: a python toolbox for meshless assimilation from image velocimetry using radial basis functions. *Journal of Open Source Software*, 9(93), 5749.

Speziale, C. (1995). *A review of Reynolds Stress models for Turbulent shear flows*. Technical Report Nasa-cr-195054, Nasa.

Speziale, C. G. (1987). On nonlinear k-l and k-ε models of turbulence. *Journal of Fluid Mechanics*, 178, 459–475.

Stadtmann, F., Rasheed, A., Kvamsdal, T., Johannessen, K. A., San, O., et al. (2023). Digital twins in wind energy: Emerging technologies and industry-informed future directions. *IEEE Access*, 11, 110762–110795.

Stengel, R. F. (2012). *Optimal control and estimation*. Courier Corporation.

Stevens, B. & Colonius, T. (2020). Enhancement of shock-capturing methods via machine learning. *Theoretical and Computational Fluid Dynamics*, 34, 483–496.

Stöcker, Y., Golla, C., Jain, R., Fröhlich, J., & Cinnella, P. (2023). Dns-based turbulent closures for sediment transport using symbolic regression. *Flow, Turbulence and Combustion*, (pp. 1–25).

Strang, G. (1968). On the construction and comparison of difference schemes. *SIAM journal on numerical analysis*, 5(3), 506–517.

Sturzebecher, D. & Nitsche, W. (2003). Active cancellation of Tollmien–Schlichting instabilities on a wing using multi-channel sensor actuator systems. *Intl J. Heat and Fluid Flow*, 24, 572–583.

Sugihara, G., May, R., Ye, H., Hsieh, C.-h., Deyle, E., Fogarty, M., & Munch, S. (2012). Detecting causality in complex ecosystems. *Science*, 338(6106), 496–500.

Sutherland, J. C. & Parente, A. (2009). Combustion modeling using principal component analysis. *Proceedings of the Combustion Institute*, 32, 1563–1570.

Sutherland, J. C., Smith, P. J., & Chen, J. H. (2007). A quantitative method for a priori evaluation of combustion reaction models. *Combustion Theory and Modelling*, 11(2), 287–303.

Sutton, R. S. & Barto, A. G. (1998). *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

Sutton, R. S. & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Preview Sample

Sutton, R. S., Barto, A. G., & Williams, R. J. (1992). Reinforcement learning is direct adaptive optimal control. *IEEE Control Systems Magazine*, 12(2), 19–22.

Suykens, J. A. K., Vandewalle, J. P. L., & Moor, B. L. R. D. (1996). *Artificial Neural Networks for Modelling and Control of Non-Linear Systems*. Springer US.

Swets, D. L. & Weng, J. (1996). Using discriminant eigenfeatures for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 18(8), 831–836.

Swischuk, R., Mainini, L., Peherstorfer, B., & Willcox, K. (2019). Projection-based model reduction: Formulations for physics-based machine learning. *Computers & Fluids*, 179, 704–717.

Taboga, M. (2021). Maximum likelihood estimation. In *Lectures on probability theory and mathematical statistics*.

Taghizadeh, S., Witherden, F. D., & Girimaji, S. S. (2020). Turbulence closure modeling with data-driven techniques: physical compatibility and consistency considerations. *New Journal of Physics*, 22(9), 093023.

Taira, K., Brunton, S. L., Dawson, S., Rowley, C. W., Colonius, T., McKeon, B. J., Schmidt, O. T., Gordeyev, S., Theofilis, V., & Ukeiley, L. S. (2017). Modal analysis of fluid flows: An overview. *AIAA Journal*, 55(12), 4013–4041.

Taira, K., Hemati, M. S., Brunton, S. L., Sun, Y., Duraisamy, K., Bagheri, S., Dawson, S., & Yeh, C.-A. (2020). Modal analysis of fluid flows: Applications and outlook. *AIAA Journal*, 58(3), 998–1022.

Takeishi, N., Kawahara, Y., & Yairi, T. (2017). Learning Koopman invariant subspaces for dynamic mode decomposition. In *Advances in Neural Information Processing Systems* (pp. 1130–1140).

Talagrand, O. (1997). Assimilation of observations, an introduction (gtspecial issueltdata assimilation in meteology and oceanography: Theory and practice). *Journal of the Meteorological Society of Japan*, 75, 191–209.

Talagrand, O. & Courtier, P. (1987). Variational assimilation of meteorological observations with the adjoint vorticity equation. i: Theory. *Quarterly Journal of the Royal Meteorological Society*, 113(478), 1311–1328.

Tebaldi, C. & Knutti, R. (2007). The use of the multi-model ensemble in probabilistic climate projections. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 365(1857), 2053–2075.

Tenenbaum, J. B., de Silva, V., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500), 2319–2323.

Tesauro, G. (1994). TD-gammon, a self-teaching backgammon program, achieves master-level play. *Neural computation*, 6(2), 215–219.

Thaler, S., Paehler, L., & Adams, N. A. (2019). Sparse identification of truncation errors. *Journal of Computational Physics*, 397, 108851.

Preview Sample

Theie Havenstrøm, S., Rasheed, A., & San, O. (2020). Deep reinforcement learning controller for 3D path-following and collision avoidance by autonomous underwater vehicles. *arXiv*, (pp. arXiv–2006).

Thomas, N., Smidt, T., Kearnes, S., Yang, L., Li, L., Kohlhoff, K., & Riley, P. (2018). Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint arXiv:1802.08219*.

Thompson, R. L., Mishra, A. A., Iaccarino, G., Edeling, W., & Sampaio, L. (2019). Eigenvector perturbation methodology for uncertainty quantification of turbulence models. *Physical Review Fluids*, 4(4), 044603.

Thomsen, K. & Sørensen, P. (1999). Fatigue loads for wind turbines operating in wakes. *Journal of Wind Engineering and Industrial Aerodynamics*, 80(1-2), 121–136.

Tibshirani, R. (1996a). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, (pp. 267–288).

Tibshirani, R. (1996b). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58(1), 267–288.

Tikhonov, A. N., Goncharsky, A., Stepanov, V., & Yagola, A. G. (2013). *Numerical methods for the solution of ill-posed problems*, volume 328. Springer Science & Business Media.

Tinoco, E. N., Brodersen, O. P., Keye, S., Laflin, K. R., Feltrop, E., Vassberg, J. C., Mani, M., Rider, B., Wahls, R. A., Morrison, J. H., Hue, D., Roy, C. J., Mavriplis, D. J., & Murayama, M. (2018). Summary data from the sixth aiaa cfd drag prediction workshop: Crm cases. *Journal of Aircraft*, 55(4), 1352–1379.

Tipping, M. E. (2001). Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning research*, 1, 211–244.

Torgerson, W. S. (1952). Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4), 401–419.

Towne, A., Schmidt, O. T., & Colonius, T. (2018). Spectral proper orthogonal decomposition and its relationship to dynamic mode decomposition and resolvent analysis. *Journal of Fluid Mechanics*, 847, 821–867.

Tracey, B., Duraisamy, K., & Alonso, J. (2013). Application of supervised learning to quantify uncertainties in turbulence and combustion modeling. *AIAA Paper*, 259, 2013. Dallas, TX, paper 2013-0259.

Tran, G. & Ward, R. (2016). Exact recovery of chaotic systems from highly corrupted data. *arXiv preprint arXiv:1607.01067*.

Tropp, J. A. & Gilbert, A. C. (2007). Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53(12), 4655–4666.

Tsoulos, I. G. & Lagaris, I. E. (2006). Solving differential equations with genetic programming. *Genetic Programming and Evolvable Machines*, 7(1), 33–54.

Tu, J. H., Rowley, C. W., Luchtenburg, D. M., Brunton, S. L., & Kutz, J. N. (2014). On dynamic mode decomposition: theory and applications. *Journal of Computational Dynamics*, 1(2), 391–421.

Preview Sample

Tucker, L. R. (1966). Some mathematical notes on three-mode factor analysis. *Psychometrika*, 16, 279–311.

Turk, M. & Pentland, A. (1991). Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1), 71–86.

Udrescu, S.-M. & Tegmark, M. (2020). Ai feynman: A physics-inspired method for symbolic regression. *Science Advances*, 6(16), eaay2631.

Uhlemann, T. H., Schock, C., Lehmann, C., Freiberger, S., & Steinhilper, R. (2017). The Digital Twin: Demonstrating the Potential of Real Time Data Acquisition in Production Systems. *Procedia Manufacturing*, 9, 113–120.

Uruba, V. (2012). Decomposition methods in turbulence research. *EPJ Web of Conferences*, 25, 01095.

Uruba, V. (2015). Near wake dynamics around a vibrating airfoil by means of PIV and oscillation pattern decomposition at reynolds number of 65 000. *Journal of Fluids and Structures*, 55, 372–383.

Uruba, V. & Procházka, P. (2019). On interpretation of spatiotemporal data decomposition. In *15th International Conference on Fluid Control, Measurements and Visualization*.

van der Maaten, L. & Hinton, G. E. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, 9, 2579–2605.

Van Overschee, P. & De Moor, B. (2012). *Subspace identification for linear systems: Theory—Implementation—Applications*. Springer Science & Business Media.

van Wieringen, W. N. (2015). Lecture notes on ridge regression.

Vanneschi, L. & Poli, R. (2012). Genetic programming — introduction, applications, theory and open issues. In G. Rozenberg, T. Bäck, & J. N. Kok (Eds.), *Handbook of Natural Computing* (pp. 709–739). Berlin, Heidelberg: Springer Berlin Heidelberg.

Vazquez, S., Leon, J. I., Franquelo, L. G., Rodriguez, J., Young, H. A., Marquez, A., & Zanchetta, P. (2014). Model predictive control: A review of its applications in power electronics. *IEEE industrial electronics magazine*, 8(1), 16–31.

Vega, J. & Le Clainche, S. (2020). Higher order dynamic mode decomposition and its applications. *BOOK: Elsevier*.

Venturi, D. & Karniadakis, G. E. (2004). Gappy data and reconstruction procedures for flow past a cylinder. *Journal of Fluid Mechanics*, 519, 315–336.

Vergassola, M., Villermaux, E., & Shraiman, B. I. (2007). Infotaxis as a strategy for searching without gradients. *Nature*, 445(7126), 406–409.

Verma, S., Novati, G., & Koumoutsakos, P. (2018). Efficient collective swimming by harnessing vortices through deep reinforcement learning. *Proceedings of the National Academy of Sciences*, 115(23), 5849–5854.

Vidal, R., Bruna, J., Giryes, R., & Soatto, S. (2017). Mathematics of deep learning. *arXiv preprint arXiv:1712.04741*.

Preview Sample

Vignon, C., Rabault, J., & Vinuesa, R. (2023). Recent advances in applying deep reinforcement learning for flow control: Perspectives and future directions. *Physics of Fluids*, 35(3).

Viquerat, J., Rabault, J., Kuhnle, A., Ghraieb, H., Larcher, A., & Hachem, E. (2021). Direct shape optimization through deep reinforcement learning. *Journal of Computational Physics*, 428, 110080.

Virgolin, M. & Pissis, S. P. (2022). Symbolic regression is NP-hard. *Transactions on Machine Learning Research*.

Virtanen, P. & et al. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17, 261–272.

Vlachas, P., Pathak, J., Hunt, B., Sapsis, T., Girvan, M., Ott, E., & Koumoutsakos, P. (2020). Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics. *Neural Networks*, 126, 191–217.

Vlachas, P. R., Byeon, W., Wan, Z. Y., Sapsis, T. P., & Koumoutsakos, P. (2018). Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. *Proc. R. Soc. A*, 474(2213), 20170844.

Vladimir Cherkassky, F. M. M. (2008). *Learning from Data*. John Wiley & Sons.

Volpiani, P. S., Bernardini, R. F., & Franceschini, L. (2022). Neural network-based eddy-viscosity correction for rans simulations of flows over bi-dimensional bumps. *International Journal of Heat and Fluid Flow*, 97, 109034.

Volpiani, P. S., Meyer, M., Franceschini, L., Dandois, J., Renac, F., Martin, E., Marquet, O., & Sipp, D. (2021). Machine learning-augmented turbulence modeling for rans simulations of massively separated flows. *Physical Review Fluids*, 6(6), 064607.

von Storch, H. & Xu, J. (1990). Principal oscillation pattern analysis of the 30- to 60-day oscillation in the tropical troposphere. *Climate Dynamics*, 4(3), 175–190.

Wahab, O. A., Mourad, A., Otrok, H., & Taleb, T. (2021). Federated machine learning: Survey, multi-level classification, desirable criteria and future directions in communication and networking systems. *IEEE Communications Surveys & Tutorials*, 23(2), 1342–1397.

Wallin, S. & Johansson, A. V. (2000). An explicit algebraic Reynolds stress model for incompressible and compressible turbulent flows. *Journal of Fluid Mechanics*, 403, 89–132.

Wang, B., Li, H., Guo, Y., & Wang, J. (2023a). PPFLHE: A privacy-preserving federated learning scheme with homomorphic encryption for healthcare data. *Applied Soft Computing*, 146, 110677.

Wang, J., Wu, J.-L., & Xiao, H. (2016). Incorporating prior knowledge for quantifying and reducing model-form uncertainty in RANS simulations. *International Journal for Uncertainty Quantification*, 6(2).

Wang, J.-X., Wu, J.-L., & Xiao, H. (2017). Physics-informed machine learning approach for reconstructing reynolds stress modeling discrepancies based on dns data. *Physical Review Fluids*, 2(3), 034603.

Wang, L. (2009). *Model predictive control system design and implementation using MATLAB®*. Springer Science & Business Media.

Preview Sample

Wang, Q., Yan, L., Hu, G., Li, C., Xiao, Y., Xiong, H., Rabault, J., & Noack, B. R. (2022). DRLinfluids: An open-source python platform of coupling deep reinforcement learning and openfoam. *Physics of Fluids*, 34(8).

Wang, R., Kashinath, K., Mustafa, M., Albert, A., & Yu, R. (2020a). Towards physics-informed deep learning for turbulent flow prediction. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 1457–1466).

Wang, R., Walters, R., & Yu, R. (2020b). Incorporating symmetry into deep dynamics models for improved generalization. *arXiv preprint arXiv:2002.03061*.

Wang, S., Teng, Y., & Perdikaris, P. (2021). Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5), A3055–a3081.

Wang, W. X., Yang, R., Lai, Y. C., Kovanis, V., & Grebogi, C. (2011). Predicting catastrophes in nonlinear dynamical systems by compressive sensing. *Physical Review Letters*, 106, 154101–1–154101–4.

Wang, Y., Cheung, S. W., Chung, E. T., Efendiev, Y., & Wang, M. (2020). Deep multiscale model learning. *Journal of Computational Physics*, 406, 109071–109071.

Wang, Y., Su, Z., Guo, S., Dai, M., Luan, T. H., & Liu, Y. (2023b). A survey on digital twins: architecture, enabling technologies, security and privacy, and future prospects. *IEEE Internet of Things Journal*, 10(17), 14956–14987.

Wang, Y., Sun, J., Bai, J., Anitescu, C., Eshaghi, M. S., Zhuang, X., Rabczuk, T., & Liu, Y. (2024). Kolmogorov arnold informed neural network: A physics-informed deep learning framework for solving forward and inverse problems based on kolmogorov arnold networks. *Arxiv*.

Wang, Z., Akhtar, I., Borggaard, J., & Iliescu, T. (2012). Proper orthogonal decomposition closure models for turbulent flows: a numerical comparison. *Computer Methods in Applied Mechanics and Engineering*, 237, 10–26.

Wassing, S., Langer, S., & Bekemeyer, P. (2024). Physics-informed neural networks for parametric compressible euler equations. *Computers & Fluids*, 270, 106164. arXiv 2307.14045.

Watkins, C. J. & Dayan, P. (1992). Q-learning. *Machine learning*, 8(3-4), 279–292.

Watt, J., Borhani, R., & Katsaggelos, A. K. (2020). *Machine Learning Refined: Foundations, Algorihtms and Applications*. Cambridge University Press.

Weatheritt, J. & Sandberg, R. (2016). A novel evolutionary algorithm applied to algebraic modifications of the RANS stress–strain relationship. *Journal of Computational Physics*, 325, 22–37.

Weatheritt, J. & Sandberg, R. D. (2017). The development of algebraic stress models using a novel evolutionary algorithm. *International Journal of Heat and Fluid Flow*, 68, 298–318.

Wehmeyer, C. & Noé, F. (2018). Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. *The Journal of Chemical Physics*, 148(241703).

Preview Sample

Welch, P. (1967). The use of fast fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms. *IEEE Transactions on Audio and Electroacoustics*, 15(2), 70–73.

Welling, M. (2013). Kernel ridge regression. Notes from.

Werbos, P. (1974). *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Dept of Applied Mathematics, Harvard University.

Werbos, P. J. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10), 1550–1560.

Werner, S. & Peitz, S. (2023). Learning a model is paramount for sample efficiency in reinforcement learning control of pdes.

Whiteley, J. (2014). Finite element methods. *A Practical Guide*, 1.

Wilcox, D. C. (2006). *Turbulence modeling for CFD*. DCW Industries, 3 edition.

Willard, J., Jia, X., Xu, S., Steinbach, M., & Kumar, V. (2020). Integrating physics-based modeling with machine learning: A survey. *arXiv preprint arXiv:2003.04919*.

Willcox, K. & Peraire, J. (2002). Balanced model reduction via the proper orthogonal decomposition. *AIAA Journal*, 40(11), 2323–2330.

Williams, F. A. (2010). Chemical-kinetic mechanisms for combustion applications.

Williams, M. O., Kevrekidis, I. G., & Rowley, C. W. (2015). A data–driven approximation of the Koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25(6), 1307–1346.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8, 229–256.

Wu, H., Ihme, M., Juniper, M. P., & Magri, L. (2019). Data assimilation and optimal calibration in nonlinear models of flame dynamics. *ASME. J. Eng. Gas Turbines Power*.

Wu, J.-L., Wang, J.-X., & Xiao, H. (2016). A Bayesian calibration–prediction method for reducing model-form uncertainties with application in RANS simulations. *Flow, Turbulence and Combustion*, 97(3), 761–786.

Wu, J.-L., Xiao, H., & Paterson, E. (2018a). Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework. *Physical Review Fluids*, 3(7), 074602.

Wu, J.-L., Xiao, H., Sun, R., & Wang, Q. (2018b). RANS equations with Reynolds stress closure can be ill-conditioned. Submitted. arXiv preprint arXiv:1803.05581.

Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Philip, S. Y., et al. (2008). Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1), 1–37.

Xiao, D. & Papadakis, G. (2019). Nonlinear optimal control of transition due to a pair of vortical perturbations using a receding horizon approach. *Journal of Fluid Mechanics*, 861, 524–555.

Preview Sample

Xiao, H. & Cinnella, P. (2019). Quantification of model uncertainty in rans simulations: A review. *Progress in Aerospace Sciences*, 108, 1–31.

Xiao, H. & Jenny, P. (2012). A consistent dual-mesh framework for hybrid LES/RANS modeling. *Journal of Computational Physics*, 231(4), 1848–1865.

Xiao, H., Wu, J.-L., Wang, J.-X., Sun, R., & Roy, C. (2016). Quantifying and reducing model-form uncertainties in reynolds-averaged navier–stokes simulations: A data-driven, physics-informed bayesian approach. *Journal of Computational Physics*, 324, 115–136.

Yang, J., Dzanic, T., Petersen, B., Kudo, J., Mittal, K., Tomov, V., Camier, J.-S., Zhao, T., Zha, H., Kolev, T., et al. (2023). Reinforcement learning for adaptive mesh refinement. In *International Conference on Artificial Intelligence and Statistics* (pp. 5997–6014).: Pmlr.

Yang, L., Zhang, D., & Karniadakis, G. E. (2020). Physics-informed generative adversarial networks for stochastic differential equations. *SIAM Journal on Scientific Computing*, 42(1), A292–a317.

Yang, Q., Liu, Y., Chen, T., & Tong, Y. (2019). Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology*, 10(2), 1–19.

Yang, X. I. A., Sadique, J., Mittal, R., & Meneveau, C. (2015). Integral wall model for large eddy simulations of wall-bounded turbulent flows. *tpf*, 27(2), 025112.

Yao, Z., Gholami, A., Shen, S., Keutzer, K., & Mahoney, M. W. (2020). Adahessian: An adaptive second order optimizer for machine learning. *arxiv*.

Ye, H., Beamish, R. J., Glaser, S. M., Grant, S. C., Hsieh, C.-h., Richards, L. J., Schnute, J. T., & Sugihara, G. (2015). Equation-free mechanistic ecosystem forecasting using empirical dynamic modeling. *Proceedings of the National Academy of Sciences*, 112(13), E1569–e1576.

Ye, Y. & Tse, E. (1989). An extension of Karmarkar's projective algorithm for convex quadratic programming. *Mathematical programming*, 44, 157–179.

Yetter, R. A., Dryer, F., & Rabitz, H. (1991). A comprehensive reaction mechanism for carbon monoxide/hydrogen/oxygen kinetics. *Combustion Science and Technology*, 79(1-3), 97–128.

Yeung, E., Kundu, S., & Hodas, N. (2017). Learning deep neural network representations for koopman operators of nonlinear dynamical systems. *arXiv preprint arXiv:1708.06850*.

Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: Lstm cells and network architectures. *Neural Comput.*, 31(7), 1235—-1270.

Yuksel, S. E., Wilson, J. N., & Gader, P. D. (2012). Twenty Years of Mixture of Experts. *IEEE transactions on neural networks and learning systems*, 23(8), 1177–1193.

Zdybał, K., Armstrong, E., Sutherland, J. C., & Parente, A. (2022). Cost function for low-dimensional manifold topology assessment. *Scientific Reports*, 12(1), 14496.

Zerfas, C., Rebholz, L. G., Schneier, M., & Iliescu, T. (2019). Continuous data assimilation reduced order models of fluid flow. *Computer Methods in Applied Mechanics and Engineering*, 357, 112596.

Zhang, C., Xie, Y., Bai, H., Yu, B., Li, W., & Gao, Y. (2021). A survey on federated learning. *Knowledge-Based Systems*, 216, 106775.

Preview Sample

Not available in preview