Fluid Mechanics and Its Applications

Thomas Duriez
Steven L. Brunton
Bernd R. Noack

Machine Learning Control – Taming Nonlinear Dynamics and Turbulence

Springer

Fluid Mechanics and Its Applications

Volume 116

Aims and Scope of the Series

The purpose of this series is to focus on subjects in which fluid mechanics plays a fundamental role.

As well as the more traditional applications of aeronautics, hydraulics, heat and mass transfer etc., books will be published dealing with topics which are currently in a state of rapid development, such as turbulence, suspensions and multiphase fluids, super and hypersonic flows and numerical modeling techniques.

It is a widely held view that it is the interdisciplinary subjects that will receive intense scientific attention, bringing them to the forefront of technological advancement. Fluids have the ability to transport matter and its properties as well as to transmit force, therefore fluid mechanics is a subject that is particularly open to cross fertilization with other sciences and disciplines of engineering. The subject of fluid mechanics will be highly relevant in domains such as chemical, metallurgical, biological and ecological engineering. This series is particularly open to such new multidisciplinary domains.

The median level of presentation is the first year graduate student. Some texts are monographs defining the current state of a field; others are accessible to final year undergraduates; but essentially the emphasis is on readability and clarity.

More information about this series at http://www.springer.com/series/5980

Thomas Duriez · Steven L. Brunton
Bernd R. Noack

Machine Learning Control – Taming Nonlinear Dynamics and Turbulence

Springer

Thomas Duriez
Laboratorio de Fluido Dinámica
CONICET - Universidad de Buenos Aires
Buenos Aires
Argentina

Steven L. Brunton
Mechanical Engineering Department
University of Washington
Seattle, WA
USA

Bernd R. Noack
Département Mécanique-Energétique
LIMSI-CNRS, UPR 3251
Orsay
France

and

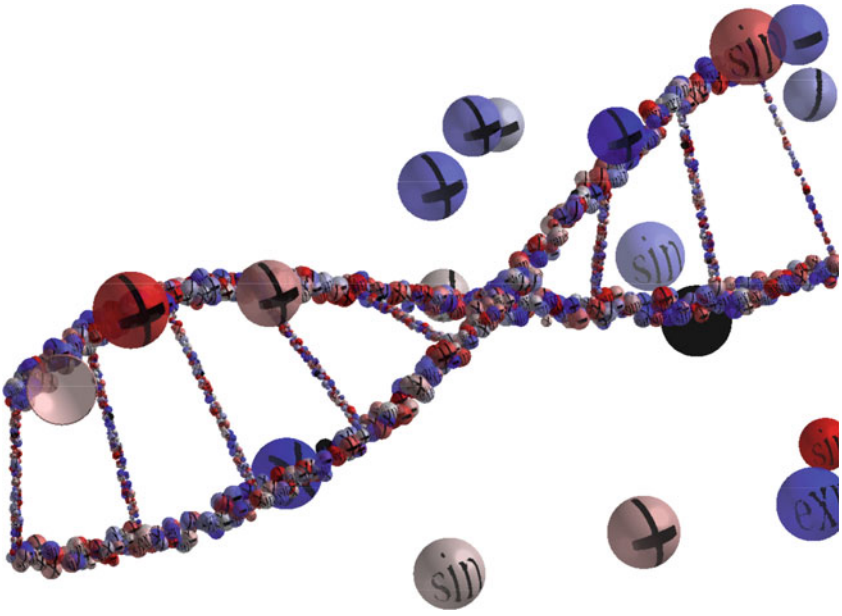
Institut für Strömungsmechanik
Technische Universität Braunschweig
Braunschweig
Germany

ISSN 0926-5112 ISSN 2215-0056 (electronic)
Fluid Mechanics and Its Applications
ISBN 978-3-319-40623-7 ISBN 978-3-319-40624-4 (eBook)
DOI 10.1007/978-3-319-40624-4

Library of Congress Control Number: 2016943421

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG Switzerland



Preface

This book is an introduction to machine learning control (MLC), a surprisingly simple model-free methodology to tame complex nonlinear systems. These systems are assumed to be manipulated by a finite number of actuators (inputs) and monitored by a finite number of sensors (outputs). The control logic is chosen to minimize a well-defined cost functional.

MLC brings together three well-established disciplines: the theory of closed-loop feedback control, machine learning and regression, and the nonlinear dynamical systems that are characteristic of turbulent fluid flows. Over the past decades, control theory has developed into a mature discipline with a beautiful theoretical foundation and powerful associated numerical algorithms. Important advances have been made to enable robust control of systems with sensor noise, external disturbances, and model uncertainty. Modern methods from control theory now pervade the engineering sciences and have transformed the industrial landscape. However, challenges remain for the control of systems with strongly nonlinear dynamics leading to broadband frequency spectra, a high-dimensional state space, and large time delays. MLC begins to address these challenges using advanced methods from machine learning to *discover* effective control laws.

Many turbulence control problems are not adequately described by linear models, have exceedingly large state spaces, and suffer from time delays from actuators to sensors via nonlinear convective fluid dynamic effects. Take for instance the aerodynamic drag minimization of a car with actuators at the back side, pressure sensors distributed over the car, and a smart feedback control logic. Numerical simulation of the underlying dynamics given by the Navier–Stokes equations requires days or weeks, while the control system requires actuation decisions on the order of milliseconds. Reduced-order models that incorporate nonlinearities, multiscale phenomena, and actuation effects have eluded many serious efforts and will likely remain elusive for years to come. In short, there may not even be a viable model for robust control design. Nevertheless, the literature contains many studies on turbulence control, with the majority either employing open-loop forcing such as periodic blowing, slowly adapting a working open-loop

strategy, or stabilizing an underlying laminar solution such as a laminar boundary layer on an aircraft wing. Feedback control responding to dominant flow structures in real time may be found in numerical studies but is rarely found in real-world experiments with turbulent flows. Indeed, turbulence control is a grand challenge problem in engineering, with far-reaching potential scientific, industrial, and societal impact.

Yet, just looking at the flight maneuvers of a bird, bat, or insect, it is clear that nature has found impressive feedback flow control solutions without employing advanced mathematical models. An eagle, for instance, can land gently on a rocky surface under gusty wind conditions and in rain by moving its wings and feathers to expertly manipulate fluid forces. More than 50 years ago, Ingo Rechenberg and Hans-Peter Schwefel have emulated nature's evolutionary way to optimize flow properties at the former Hermann-Föttinger-Institut of the Berlin Institute of Technology, Germany. Their pioneering evolutionary strategies started evolutionary computations. Subsequent innovations include evolutionary programming (Fogel Owens and Walsh 1966), genetic algorithms (Holland 1975) and genetic programming (Koza 1992). These evolutionary computations constitute an important pillar of machine learning. In a visionary publication in 1959, the artificial intelligence pioneer Arthur Samuel defined machine learning as a 'field of study that gives computers the ability to learn without being explicitly programmed.' Machine learning is a rapidly evolving discipline of computer science that is benefiting from the current explosion of big data. It has successfully improved an immense range of technologies—from the smart phone to the autopilot in Tesla's Sedan and to large-scale factory processes. In academia, nearly all scientific disciplines are profiting from machine learning.

Not surprisingly, machine learning methods may augment or replace control design in myriad applications. Robots learn to walk with dynamic programming. Genetic algorithms are used to optimize the coefficients in proportional–integral–derivate (PID) controllers. The groups of N. Benard and E. Moreau employ genetic algorithms to optimize linear sensor feedback in a flow control experiment. Reinforcement learning has been successfully applied to stabilize chaotic dynamics, and has recently also reduced cavity noise in an experiment operated by F. Lusseyran, L. Pastur and L. Mathelin. The authors of this book have pushed the first applications of genetic programming in feedback control of nonlinear dynamics, direct Navier–Stokes simulations and experimental turbulent shear flows. This book focuses on arguably one of the simplest, most versatile and yet very powerful version of machine learning control: Optimal nonlinear control laws are identified with genetic programming. Corresponding success stories are described throughout this book.

The authors have taught material from this book in several university courses. These courses focus on basic principles using simple examples, and the content requires anywhere from 15 to 30 h to cover. Our students have had backgrounds in computer science, control theory, nonlinear dynamics, or fluid mechanics. The prospective reader is not expected to have hands-on expertise in any of these fields but should come with the ambition to control a complex system. The book is

organized as follows. In Chap. 1, the reader is introduced to feedback control and its challenges for complex real-world problems. Chapter 2 constitutes the core of the book. This chapter formulates feedback control as a regression problem and employs genetic programming as a powerful regression technique to identify the best feedback law. Chapter 3 reviews classical methods of control theory against which MLC is benchmarked in Chap. 4 for linear and weakly nonlinear dynamics. These chapters provide context for feedback control, but they are not required to implement the MLC methods in Chap. 2. The hurried reader may jump to Chap. 5 if she/he is interested in strongly nonlinear dynamics applications or to Chap. 6 if she/he is interested in experimental implementations of feedback flow control. Chapter 7 distills good practices for real-world experiments that need to be taken into account in any MLC implementation. In Chap. 8 we provide an outlook on future methodological advances, which are expected to drastically amplify the applicability and performance of MLC. In addition, we list a number of future MLC applications with epic proportions.

We have profited tremendously from interactions with many colleagues on machine learning control. First, we highly appreciate André Thess for his continual encouragement to write a book about turbulence control for this Springer series. He has nurtured the idea for years before we decided to write this book. We highly appreciate the insightful and inspiring interviews with leading scholars of the field: Shervin Bagheri, Belinda Batten, Mark Glauser, Marc Schoenauer, and David Williams. These additions provide valuable perspectives for past progress and future work. Eurika Kaiser has provided continual exquisite feedback on our chapters and also contributed with her illuminating visualizations in Chap. 7 showing the performance of MLC.

We have also benefited greatly from our mentors throughout our careers. BRN is deeply indebted to his turbulence control mentors Andrzej Banaszuk, Andreas Dillmann, Helmut Eckelmann, Rudibert King, and William K. George, who shared and fueled the passion for the field. SLB would like to gratefully acknowledge and thank Nathan Kutz, Naomi Leonard, Richard Murray, Clancy Rowley, and Rob Stengel, who each found unique ways to make dynamics and control theory come to life. TD would like to acknowledge Eduardo Jose Wesfreid, Jean-Luc Aider, Guillermo Artana, Luc Pastur, François Lusseyran, and Bernd R. Noack, who each have had a profound (and most beneficial) impact on his perception of the different fields he has been in contact with. This book would not have been possible without our many colleagues, collaborators, and co-authors who have shared our early enthusiasm for MLC and have dedicated significant energy to developing it: Markus Abel, Jean-Luc Aider, Zhe Bai, Diogo Barros, Jean-Paul Bonnet, Jacques Borée, Bing Brunton, Juan Martin Cabaleiro, Camila Chevot, Tom Daniel, Antoine Debien, Laurent Cordier, Christophe Cuvier, Joël Delville (d), Caroline Fourment (d), Hiroaki Fukumoto, Nicolas Gautier, Fabien Harambat, Eurika Kaiser, Laurent Keirsbulck, Azeddine Kourta, Kai von Krbek, Nathan Kutz, Jean-Charles Laurentie, Ruiying (Cecile) Li, François Lusseyran, Robert Martinuzzi, Lionel Mathelin, Nicolas Mazellier, Marek Morzyński, Christian Nayeri, Robert Niven, Akira Oyama, Vladimir Parezanović, Oliver Paschereit, Luc Pastur, Brian Polagye,

Josh Proctor, Bartosz Protas, Rolf Radespiel, Cedric Raibaudo, Jim Riley, Tony Ruiz, Michael Schlegel, Peter Scholz, Marc Segond, Richard Semaan, Tamir Shaqarin, Andreas Spohn, Michel Stanislas, Ben Strom, and Sam Taira. Many of our co-authors have applied the nascent MLC methodology in their own experiments early on, when success was far from certain. We thank our students for visiting our courses in Argentina, France, Germany, and the USA and contributing with many good questions, new ideas and encouraging project results. Anneke Pot from Springer Publisher has dependably supported us in critical decisions about book contents and the production procedure.

Buenos Aires — Thomas Duriez
Seattle — Steven L. Brunton
Paris-Saclay — Bernd R. Noack
April 2016

Contents

Abbreviations

ANN	Artificial Neural Network
ARMA(X)	Auto-Regressive Moving Average (with eXogenous input)
AVERT	Aerodynamic Validation of Emission Reducing Technologies
BPOD	Balanced Proper Orthogonal Decomposition
CROM	Cluster-based Reduced-Order Modeling
DEIM	Discrete Empirical Interpolation Method
DMD	Dynamic Mode Decomposition
EC	Evolutionary Computing
EP	Evolutionary Programming
ERA	Eigensystem Realization Algorithm
GA	Genetic Algorithm
GMFM	Generalized Mean-Field Model
GP	Genetic Programming
LML	Laboratoire de Mécanique de Lille, Université de Lille 1 Cité Scientifique, Bâtiment M3 - 59655 Villeneuve d'Ascq Cedex, France
LPV	Linear Parameter Varying
LQE	Linear Quadratic Estimation
LQG	Linear Quadratic Gaussian
LQR	Linear Quadratic Regulator
MIMO	Multiple Input Multiple Output
MISO	Multiple Input Single Output
ML	Machine Learning
MLC	Machine Learning Control
NLSA	Nonlinear Laplacian Spectral Analysis
OKID	Observer Kalman filter IDentification
PCA	Principal Component Analysis
PID	Proportional–Integral–Derivative
PIV	Particle Image Velocimetry

PMMH	Physique et Mecanique des Millieux Hétérogènes Laboratory, 10 rue Vauquelin, 75231 Paris Cedex, France
POD	Proper Orthogonal Decomposition
PPRIME	Institute Pôle Poitevin de Recherche pour l'Ingénieur en Mécanique, Matériaux et Énergétique, 11 Boulevard Marie et Pierre Curie BP 30179, 86962 Futuroscope Chasseneuil Cedex, France
PRISME	Laboratoire Pluridisciplinaire de Recherche, Ingénierie des Systémes, Mecanique, Energétique. Université d'Orléans 8 Rue Léonard de Vinci, 45072 Orléans, France
ROM	Reduced-Order Model
RT	Real-Time
SIMO	Single Input Multiple Output
SISO	Single Input Single Output
SSA	Singular Spectrum Analysis
SVM	Support Vector Machine
TUCOROM	TUrbulence COntrol using Reduced-Order Models, ANR Chair of Excellence (ANR-10-CEXC-0015), Poitiers, France
UVG	Unsteady Vortex Generator

Symbols

$\mathbf{A}$; $\mathbf{A}_d$; $\tilde{\mathbf{A}}$	State matrix (continuous time; discrete time; reduced)
$\mathbf{a}$; $\mathbf{a_k}$; a_m; a	State (vector, continuous time; vector, discrete time kth step; mth component; scalar)
$\hat{\mathbf{a}}$; $\hat{\mathbf{a}}_k$	Full-state estimate (continuous; discrete time)
$\mathbf{B}$; $\mathbf{B}_d$; $\tilde{\mathbf{B}}$	Input matrix (continuous; discrete time; reduced)
B	Amplitude of periodic forcing
$\mathbf{b}$; $\mathbf{b_k}$; b_m; b	Actuation command (vector, continuous time; vector, discrete time kth step; mth component; scalar)
$\mathscr{B}$	Matrix of control inputs
$\mathbf{C}$; $\mathbf{C}_d$; $\tilde{\mathbf{C}}$	Output matrix (continuous; discrete time; reduced)
c_μ	Momentum coefficient
$\mathscr{C}$; $\mathscr{C}_\mathbf{d}$	Controllability matrix (continuous; discrete time)
$\mathbf{D}$; $\mathbf{D}_d$; $\tilde{\mathbf{D}}$	Feedthrough matrix (continuous; discrete time, reduced)
D	Characteristic distance
d_c	Duty cycle
$\mathbf{e}_x$, $\mathbf{e}_y$, $\mathbf{e}_z$	Unity vectors associated with directions x, y and z
$\mathbb{E}$	Expectation operator
$\mathbf{F}$	Dynamics
F_D	Drag force
$\mathbf{G}$	Measurement function
g	Gain of control command in a generalized mean-field model
$\mathbf{H}$	Hankel matrix
H	Heavyside function
H_{section}	Height of the test section
$h_i(t)$; $h_{i,u}$; $h_{i,\max}$	hot-wire or hot-film signal number i (raw signal average value of the unactuated measurement average measurement under constant maximal actuation)
h_{step}; h_{ramp}	Height (of the step of the ramp)
$\mathbf{I}$	Identity matrix
i	Index of individual (or other counter)

J; J_i^j	Cost function value; of individual i in generation j
J_a	Cost on states
J_b	Cost on actuation
j	Index of generation
$\mathbf{K}$	Control function
$\mathbf{K}_f$	Kalman filter gain
$\mathbf{K}_r$	Regulator gain, full-state control matrix
L	Length of the experimental test section
L_{sep}	Separation length
l	Width of the experimental test section
ℓ	Ramp length
N_a	Number of states
N_b	Number of actuation commands
N_e	Number of individuals concerned by elitism
N_g	Number of generations
N_i	Number of individuals
N_p	Tournament size
N_s	Number of sensors
$\mathcal{O}$; $\mathcal{O}_{\mathbf{d}}$	Observability matrix (continuous time; discrete time)
P_c	Probability of crossover
P_m	Probability of mutation
P_r	Probability of replication
p	Pressure
$p(\mathbf{a})$	Probability density of states
$\mathbf{Q}$	State cost weight matrix for LQR
Q; Q_u	Flow rate to actuator jets (instantaneous; average value under constant blowing)
$\mathbf{R}$	Actuation cost weight matrix for LQR
$r_\bullet$, $r_\circ$	Amplitude of oscillators of a generalized mean-field model (Table 5.1)
Re	Reynolds number
$S_a(t)$; $S_{a,u}$	Area of backflow (instantaneous; unactuated average value)
S_b	Actuator cross section
S_j	Jet cross section
S_{ref}	Ramp reference surface
$\mathbf{s}$; $\mathbf{s}_k$; s_m; s	Sensor signal (vector, continuous time; vector, discrete time kth step; mth component; scalar)
$\hat{\mathbf{s}}$; $\hat{\mathbf{a}}_k$	Expected sensor value (continuous time; discrete time)
$\mathscr{S}$; $\bar{\mathscr{S}}$	Markov parameters; of the augmented system
T	Evaluation time
T_{rms}	Time period used to compute RMS of hot-wire signal fluctuations
t, t_0	Time, initial time
$\mathbf{U}$; $\mathbf{U}_r$	Left singular vectors of SVD (complete; reduced)

U	Characteristic velocity
$\mathbf{u}$; $\mathbf{u}_s$, $\mathbf{u}_\Delta$, $\mathbf{u}_\bullet$, $\mathbf{u}_\circ$	Velocity (vector field; steady solution; deviation due to Reynolds stresses; contribution of frequency $\omega_\bullet$; contribution of frequency $\omega_\circ$)
$\overline{\mathbf{u}}$	Slow varying mean flow
$\mathbf{u}'$	Flow fluctuations
u	Streamwise velocity component
$\mathbf{V}$; $\mathbf{V}_r$	Right singular vectors of SVD (complete reduced)
$\mathbf{V}_d$	Disturbance variance
$\mathbf{V}_n$	Noise variance
V_{Jet}	Characteristic velocity of jets
$\mathbf{v}$	Velocity vector initial condition
v	Transverse velocity component
$\mathbf{W}_{\mathscr{C}}^d$	Discrete-time controllability Gramian
$\mathbf{W}_{\mathscr{O}}^d$	Discrete-time observability Gramian
W	Mixing layer width
$\mathbf{w}$	Disturbance array
$\mathbf{w}_r$	External reference signal
$\mathbf{w}_d$	External disturbance, process noise
$\mathbf{w}_n$	Measurement noise
w	Spanwise velocity component
$\mathbf{X}$	Solution to the Riccati equation for LQR
$\mathbf{x}$	Space vector
x	Streamwise coordinate
$\mathbf{Y}$	Solution to the Riccati equation for Kalman filter
y	Transverse coordinate
$\mathbf{z}$	System output
z	Spanwise coordinate
$\beta_{\bullet\bullet}$, $\beta_{\bullet\circ}$, $\beta_{\circ\bullet}$, $\beta_{\circ\circ}$	Parameter for growth rate change in oscillators of a generalized mean-field model (Table 5.1)
γ	Penalization coefficient
$\gamma_{\bullet\bullet}$, $\gamma_{\bullet\circ}$, $\gamma_{\circ\bullet}$, $\gamma_{\circ\circ}$	Parameter for frequency change in oscillators for a generalized mean-field model (Table 5.1)
$\delta(\cdot)$	Dirac delta function
ε	Nonlinearity strength coefficient or state stabilization error
κ	Gain of the generalized mean-field model
ν	Kinematic viscosity
ρ	Fluid density
$\sum$; $\sum_r$	Singular values matrix of SVD, (complete; reduced)
σ	Oscillator growth rate
$\sigma_\bullet$; $\sigma_{\bullet\star}$; $\sigma_\circ$; $\sigma_{\circ\star}$	Growth rate of oscillators of a generalized mean-field model (Table 5.1)
τ; τ_a; τ_u	Period of time (with actuated system; with unactuated system)

$\phi_\bullet$, $\phi_\circ$	Phase of oscillators in a generalized mean-field model (Table 5.1)
χ	Backflow coefficient
Ω	Space domain
ω	Oscillator pulsation
$\omega_\bullet$; $\omega_{\bullet\bigstar}$; $\omega_\circ$; $\omega_{\circ\bigstar}$	Frequency of oscillators in a generalized mean-field model (Table 5.1)

Chapter 1
Introduction

I think it's very important to have a feedback loop, where you're constantly thinking about what you've done and how you could be doing it better.

Elon Musk

1.1 Feedback in Engineering and Living Systems

Feedback processes are critical aspects of most living and engineering systems. *Feedback* occurs when the output of a system influences the input of the same system. *Feedback control* is a process of creating such a feedback loop to modify the behavior of a dynamical system through actuation that is informed by measurements of the system.

The very existence of humans and other endothermic animals is based on a robust feedback control: They maintain their body temperature within narrow limits despite a large range of environmental conditions and disturbances. This temperature regulation is performed with temperature monitoring and control actions, such as increasing metabolism or sweating. Similarly, air conditioning also keeps a room temperature in a narrow interval by heating or cooling via a ventilating air stream.

The world around us is actively shaped by feedback processes, from the meandering path of a river to the gene regulation that occurs inside every cell in our body. A child's education may be considered a feedback control task, where parental and societal feedback guide the child's actions towards a desired goal, such as socially acceptable behavior and the child becoming a productive member of society. The order achieved in a modern society is the result of a balance of interests regulated through active policing and the rule of laws, which are in turn shaped by a collective sense of justice and civil rights. Financial markets and portfolio management are also feedback processes based on a control logic of buying and selling stocks to reach an optimal growth or profit at a given risk over a certain time horizon. In

T. Duriez et al., *Machine Learning Control – Taming Nonlinear Dynamics and Turbulence*, Fluid Mechanics and Its Applications 116,
DOI 10.1007/978-3-319-40624-4_1

fact, currency inflation is actively manipulated by changing interest rates and issuing bonds. Our very thoughts and actions are intimately related to a massively parallel feedback architecture in our brain and nervous system, whereby external stimuli are collected and assimilated, decisions are made, and control actions are executed, resulting in our interaction with the world. Finally, the Earth's climate and temperature are maintained through a delicate balance of forcing from sources including solar irradiance, greenhouse gases, vegetation, aerosols and cloud formation, many of which are coupled through feedback.

The feedback control of fluid systems is an immensely important challenge with profound implications for technologies in energy, security, transportation, medicine, and many other endeavors. Flow control is an academically exciting research field undergoing rapid progress—comprising many disciplines, including theoretical, numerical and experimental fluid mechanics, control theory, reduced-order modeling, nonlinear dynamics and machine learning techniques. Flow control has applications of epic proportion, such as drag reduction of cars, trucks, trains, ships and submarines, lift increase of airplanes, noise reduction of ground or airborne transport vehicles, combustion efficiency and NO_X reduction, cardiac monitoring and intervention, optimization of pharmaceutical and chemical processes and weather control. The flows found in most engineering applications are turbulent, introducing the complexities of high-dimensionality, multi-scale structures, strong nonlinearities and frequency crosstalk as additional challenges.

Feedback turbulence control shares a significant overlap with the other feedback systems described above, in the sense that

- the control goal can be defined in mathematical terms;
- the control actions are also in a well-defined set;
- the unforced system has its own internal *chaotic* nonlinear dynamics, where neighboring states may rapidly diverge to different behaviors within the prediction horizon;
- the full state is only partially accessible by limited sensors;
- there is an underlying evolution equation (i.e., the Navier–Stokes equation) which provides a high-fidelity description of the system, but may not be useful for control decisions in a real-life experiment.

The last three properties are a generic consequence of high-dimensional nonlinear dynamics. However, unlike many of the systems described above, turbulence control is more benign, as the system quickly forgets its past treatment and the control experiments tend to be more reproducible. In other words, the unforced and forced systems have a statistical stationarity, i.e. statistical quantities like mean values and variances are well defined. Regardless, feedback turbulence control is significantly more complex than most academic control theory tasks, such as stabilization of an inverted pendulum. Hence, improved feedback control architectures that work for turbulence control may have significant impact in other complex systems.

Nature offers compelling examples of feedback flow control that may provide inspiration for engineering efforts. For example, eagles are expert flyers, capable of rising on thermals or landing gently on a rock or tree despite strong wind gust

perturbations and other challenging weather conditions. These maneuvers require active feedback control by sensing the current position and velocity and dynamically adjusting the control actions involving the motion of wings and feathers. An eagle's flight is robust to significant uncertainty in the environment and flight conditions, such as harsh weather and significant changes to its own body, including mass, geometry, and wing shape. It is unlikely that eagles, or other flying animals, such as birds, bats, or insects, are operating based on a high-fidelity model of the underlying Navier–Stokes equations that govern fluid flow. Instead, it is more likely that these animals have adapted and learned how to sense and modify dominant coherent structures in the fluid that are most responsible for generating forces relevant for flight. Airplanes similarly move on prescribed trajectories at predetermined speeds under varying wind and weather conditions by adjusting their control surfaces, such as flaps and ailerons, and engine thrust. However, there is still a tremendous opportunity to improve engineering flight performance using bio-inspired techniques.

This book outlines the use of machine learning to design control laws, partially inspired by how animals learn control in new environments. This *machine learning control* (MLC) provides a powerful new framework to control complex dynamical systems that are currently beyond the capability of existing methods in control.

1.2 Benefits of Feedback Control

Figure 1.1 illustrates a general feedback control system. The physical system, also called the *plant*, is depicted in the blue box. The system is monitored by sensors **s** and manipulated by actuators **b** through a control logic depicted in the yellow box.

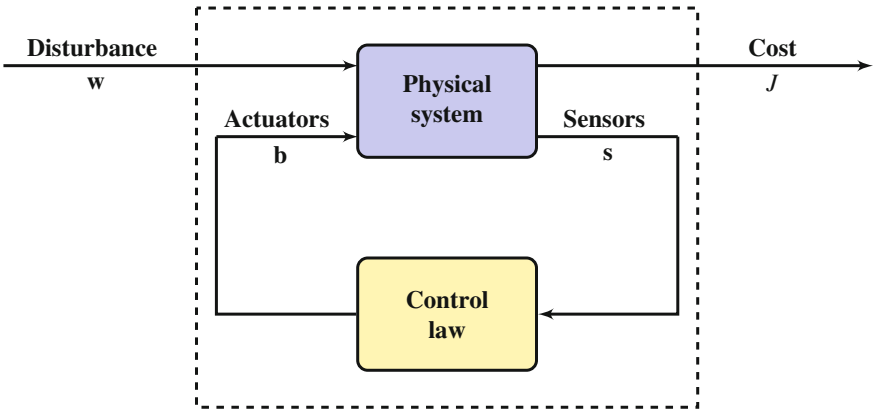


Fig. 1.1 General optimization framework for feedback control. The behavior of the physical system is modified by actuators (inputs, **b**) through a control law informed by sensor measurements of the system (outputs, **s**). The control logic is designed to shape the closed-loop response from the exogenous disturbances **w** to a high-level objective encoded by the cost function J

Moreover, the plant is subjected to sensor noise and exogenous disturbances $\mathbf{w}$ and the control shall be optimized with respect to a cost function J.

One example of an optimization tasks is drag reduction. A physically meaningful optimization problem penalizes the actuation. A well-posed drag reduction problem requests a minimization of the power required to overcome drag J_{drag} plus the invested actuation power J_{act}, i.e. the net gain $J = J_{\text{drag}} + J_{\text{act}}$. Other examples include lift increase, mixing increase and noise reduction. To keep an airplane on a desired trajectory, the thrust and lift need to be kept at a well-defined level. Thus, the control task becomes a *reference tracking problem*, in which a reference force—or other quantity—is commanded. In this case, the cost function penalizes the deviation from the desired state and the invested actuation level.

In the case of reference tracking, it is natural to first consider the open-loop control architecture shown in Fig. 1.2. In this strategy, the actuation signal $\mathbf{b}$ is chosen based on knowledge of the system to produce the desired output that matches the commanded reference signal. This is how many toasters work, where the heating element is turned on for a fixed amount of time depending on the desired setting. However, open-loop control is fundamentally incapable of stabilizing an unstable system, such as an inverted pendulum, as the plant model would have to be known perfectly without any uncertainty or disturbances. Open-loop control is also incapable of adjusting the actuation signal to compensate for unmeasured disturbances to the system.

Instead of making control decisions purely based on the desired reference, as in open-loop control, it is possible to *close the loop* by feeding back sensor measurements of the system output so that the controller knows whether or not it is achieving the desired goal. This *closed-loop feedback control* diagram is shown in Fig. 1.3. Sensor-based feedback provides a solution to the issues that occur with open-loop control. It is often possible to stabilize an unstable system with the aid of sensor feedback, whereas it is never possible to stabilize an unstable system in open-loop. In addition, closed-loop control is able to compensate for external disturbances and model uncertainties, both of which should be measured in the sensor output.

Summarizing, feedback control is, for instance, necessary for the following tasks:

- Optimize a state or output with respect to a given cost function;
- Stabilize an unstable system;

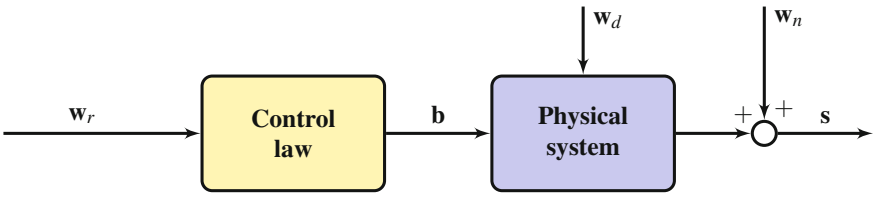


Fig. 1.2 Open-loop control diagram. A reference signal $\mathbf{w}_r$ is fed directly into an open-loop controller which specifies a pre-determined actuation signal $\mathbf{b}$. External disturbances ($\mathbf{w}_d$) and sensor noise ($\mathbf{w}_n$), as well as un-modeled system dynamics and uncertainty, degrade the overall performance

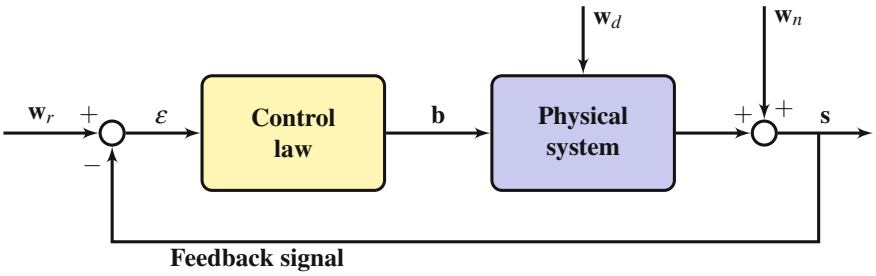


Fig. 1.3 Closed-loop feedback control diagram. The sensor signal **s** is fed back and subtracted from the reference signal $\mathbf{w}_r$. The resulting error $\boldsymbol{\varepsilon}$ is used by the controller to specify the actuation signal **b**. Feedback is generally able to stabilize unstable plant dynamics while effectively rejecting disturbances $\mathbf{w}_d$ and attenuating noise $\mathbf{w}_n$

- Attenuate sensor noise;
- Compensate for exogenous disturbances and model uncertainty.

Mathematical Formulation of Feedback Control Task

There is a powerful theory of feedback control based on dynamical systems. In this framework, the plant is modeled by an input–output system:

$$\frac{d}{dt}\mathbf{a} = \mathbf{F}(\mathbf{a}, \mathbf{b}, \mathbf{w}_d), \tag{1.1a}$$

$$\mathbf{s} = \mathbf{G}(\mathbf{a}, \mathbf{b}, \mathbf{w}_n), \tag{1.1b}$$

consisting of a coupled system of possibly nonlinear differential equations in a state variable $\mathbf{a} \in \mathbb{R}^{N_a}$, where N_a is the dimension of the state. The actuation input is given by the vector $\mathbf{b} \in \mathbb{R}^{N_b}$ and this input directly affects the state dynamics in Eq. (1.1a), along with exogenous disturbances $\mathbf{w}_d$. The sensor measurements are given by the output vector $\mathbf{s} \in \mathbb{R}^{N_s}$, and these measurements may be nonlinear functions of the state **a**, the control **b** and noise $\mathbf{w}_n$.

The control task is generally to construct a controller

$$\mathbf{b} = \mathbf{K}(\mathbf{s}, \mathbf{w}_r), \tag{1.2}$$

so that the closed-loop system has desirable properties in terms of stability, attenuation of noise, rejection of disturbances, and good reference tracking characteristics. The commanded reference signal is $\mathbf{w}_r$. These factors are encoded in the cost function J, which is generally a function of the sensor output, the actuation input, and the various external signals $\mathbf{w}_r$, $\mathbf{w}_d$, and $\mathbf{w}_n$.

With a well-designed sensor-based feedback control law, it is often possible to obtain a closed-loop system that performs optimally with respect to the chosen cost function and is robust to model uncertainty, external disturbances, and sensor noise. In fact, most modern control problems are posed in terms of optimization via cost

minimization. The perspective taken in this book is that machine learning provides a powerful new set of optimization techniques to obtain high-performance control laws even for extremely complicated systems with non-convex cost functions.

1.3 Challenges of Feedback Control

Most textbooks start with simple feedback control problems. An airplane, for instance, may need to keep a certain ground speed. The airplane has a steady-state map (model) indicating the required thrust (actuation) under ambient flow conditions and for an average airplane. Thus, the right thrust may be commanded in an open-loop manner based on the model, as illustrated in Fig. 1.2.

Yet, each airplane has its own steady-state map and an aging process (model uncertainty). Moreover, the wind (exogenous disturbance) may change the ground velocity. Model uncertainty and disturbances require a feedback element: The ground speed needs to be measured (tachometer) and the thrust needs to be adjusted. If the ground speed is too low (high), the thrust needs to be increased (decreased). The general feedback scheme is illustrated in Fig. 1.3.

Evidently, the control design is simple. There is a single state variable a (speed) which is sensed s (tachometer) and acted upon b (thrust) in a highly predictable manner and with negligible time delay. We refer to the excellent textbook of Åström and Murray [223] for the applicable control design.

The stabilization of steady solutions to the equations for laminar or transitional flows requires more refined methods. A sufficiently detailed discretized of the Navier–Stokes equation results in a system with a high-dimensional state, making it computationally expensive to design and implement controllers. In addition, time-scales may be very fast in real-world fluid applications, such as flow over a wing or in a combustor, making controllers sensitive to time delays; these time-delays may be due to sensor and actuator hardware or the computational overhead of enacting a control law. Sensor and actuator placement is also a challenge in high-dimensional fluid systems, with competing goals of decreasing time delays and increasing downstream prediction. Finally, many fluid systems are characterized by strongly non-normal linearized dynamics, meaning that the linearized Navier–Stokes equations have nearly parallel eigenvectors resulting in large transient growth of these modes in response to excitation [67, 263].

Despite inherent nonlinearity, stabilizing a steady state brings the system closer to the equilibrium solution where linearization is increasingly valid. Thus, the fluid dynamics literature contains a rich set of success stories based on linear control methods. Examples include the stabilization of the cylinder wake [65, 115, 227, 219], of the cavity flow [232], of the boundary layer [11, 172], and of the channel flow [29], just to name a few [43]. The linear quadratic regulator and linear quadratic Gaussian are among the most widely used methods for control based on computational fluid mechanics. The model-based control of experimental plants requires reduced-order models for computationally tractable on-line decisions. For details, we refer to

excellent reviews on the applications of linear control theory in fluids mechanics [13, 161, 232, 251]. The associated reduced-order modeling efforts are summarized in these reviews and elaborated in [138, 197].

Optimization of turbulent flows tends to be much more complex. In addition to the challenges outlined above, the system is strongly nonlinear and is sufficiently far from a fixed point or limit cycle that linearization is not typically useful. The nonlinearity manifests in frequency crosstalk, where actuation at a given frequency may excite or suppress entirely different frequencies. Fully turbulent dynamics are typically chaotic and evolve on a high-dimensional attractor, with the dimension of the attractor generally increasing with the turbulence intensity. These are mathematical issues in turbulence control, but there are also more practical engineering issues. These include the cost of implementing a controller (i.e., actuator and sensor hardware, modifying existing designs, etc.), computational requirements to meet exceedingly short time scales imposed by fast dynamics and small length scales, and achieving the required control authority to meaningfully modify the flow.

As a motivating example, let us assume we want to minimize the aerodynamic drag of a car with, say, 32 blowing actuators distributed over all four trailing edges and the same number of pressure sensors distributed over the car. A control logic for driving the actuators based on the sensor readings shall help to minimize the effective propulsion power required to overcome drag. This highlights the significant challenges associated with in-time control:

- High-dimensional state;
- Strong nonlinearity;
- Time delays.

A direct numerical simulation of a suitably discretized Navier–Stokes equation has not been performed for wind-tunnel conditions. Even a simplifying large eddy simulation requires at minimum tens of millions of grid points and still has a narrow low-frequency bandwidth for actuation. Secondly, the turbulent flow does not respond linearly to the actuation, so that there is no superposition principle for actuation effects. The changes to the flow caused by two actuators acting simultaneously is not given by the sum of the responses of the two actuators acting alone. Moreover, actuating at twice the actuation amplitude does not necessarily lead to twice the output. The trend may even be reversed. Thirdly, the effect of actuation is generally not measured immediately. It may take hundreds or thousands of characteristic time scales to arrive at the converged actuated state [21,205]. We refer to our review article on closed-loop turbulence control [43] for in-depth coverage of employed methods.

1.4 Feedback Turbulence Control is a Grand Challenge Problem

A high-dimensionsional state space and nonlinear dynamics do not necessarily imply unpredictable features. One liter of an ideal gas, for instance, contains $O(10^{24})$ molecules that move and collide according to Newton's laws. Elastic collisions signify

strongly nonlinear dynamics, and indeed, the numerical simulation of Newton's laws at macro-scale based on molecular dynamics will remain intractable for decades to come. Yet, statistical averages are well described as an analytically computable maximum entropy state. This is the statistical foundation of classical thermodynamics. In contrast, the search for similar closures of turbulence has eluded any comparable success [198]. One reason is the ubiquitous Kolmogorov turbulence cascade. This cascade connects large-scale energy carrying anisotropic coherent structures with nearly isotropic small-scale dissipative structures over many orders of magnitudes in scale [106]. The multi-scale physics of turbulence has eluded all mathematical simplifications. Feynman has concluded that 'Turbulence is the most important unsolved problem of classical physics.' In other words: a grand challenge problem.

Turbulence control can be considered an even harder problem compared to finding statistical estimates of the unforced state. The control problem seeks to design a small $O(\varepsilon)$ actuation that brings about a large change in the flow. Many approaches would require a particularly accurate control-oriented closure. The necessary control mechanism might be pictured as a Maxwellian demon who changes the statistical properties of the system by clever actions. Control theory methods often focus on stabilization of equilibria or trajectories. Turbulence, however, is too far from any fixed point or meaningful trajectory for the applicability of linearized methods. In the words of Andrzej Banaszuk (1999):

> The control theory of turbulence still needs to be invented.

1.5 Nature Teaches Us the Control Design

In the previous section, a generic control strategy for turbulence has been described as a grand challenge problem. Yet, an eagle can land on a rock performing impressive flight maneuvers without a PhD in fluid mechanics or control theory. Nature has found another way of control design: learning by trial and error.

It is next to impossible to *predict* the effect of a control policy in a system such as turbulence where we scarcely understand the unforced dynamics. However, it may be comparatively easy to *test* the effectiveness of a control policy in an experiment. It is then possible to evolve the control policy by systematic testing, exploiting good control policies and exploring alternative ones. Following these principles, Rechenberg [224] and Schwefel [243] have pioneered *evolutionary strategies* in design problems of fluid mechanics more than 50 years ago at TU Berlin, Germany.

In the last 5 decades, biologically inspired optimization methods have become increasingly powerful. Fleming and Purshouse [103] summarize:

> The evolutionary computing (EC) field has its origins in four landmark evolutionary approaches: evolutionary programming (EP) (Fogel, Owens, and Walsh, 1966), evolution strategies (ES) (Schwefel, 1965; Rechenberg, 1973), genetic algorithms (GA) (Holland, 1975), and genetic programming (GP) (Koza, 1992).

EP, GA and GP can be considered regression techniques to find input–output maps that minimize a cost function. Control design can also be considered a regression task: find the mapping from sensor signals to actuation commands that optimizes the goal function. Not surprisingly, evolutionary computing is increasingly used for complex control tasks. For example, EP is used for programming robot missions [272]. GA are used to find optimal parameters of linear control laws [23, 90]. And for almost two decades, GP has been employed to optimize nonlinear control laws [91]. Arguably GP is one of the most powerful regression techniques as it leads to analytical control laws of almost arbitrary form. All evolutionary methods are part of the rapidly evolving field of *machine learning*. There are many other machine learning techniques to discover input–output maps, such as decision trees, support vector machines (SVM), and neural networks, to name only a few [280]. In fact, the first example of feedback turbulence control with machine learning methods has employed a neural network [171]. In the remainder of this book, we refer to *machine learning control* as a strategy using any of the aforementioned data-driven regression techniques to discover effective control laws.

1.6 Outline of the Book

The outline of the book is as follows. Chapter 2 describes the method of machine learning control (MLC) in detail. In Chap. 3, linear control theory is presented to build intuition and describe the most common control framework. This theoretical foundation is not required to understand or implement MLC, but it does motivate the role of feedback and highlights the importance of dynamic estimation. In Chap. 4, MLC is benchmarked against known optimal control design of linear systems without and with noise. We show that MLC is capable of reproducing the optimal linear control but outperforms these methods even for weak nonlinearities. In Chap. 5 we illustrate MLC for a low-dimensional system with frequency crosstalk. A large class of fluid flows are described by such a system. We show that the linearized system is uncontrollable while MLC discovers the enabling nonlinearity for stabilization. In Chap. 6, we highlight promising results from MLC applied in real-world feedback turbulence control experiments. Chapter 7 provides a summary of best practices, tactics and strategies for implementing MLC in practice. Chapter 8 presents concluding remarks with an outlook of future developments of MLC.

1.7 Exercises

Exercise 1–1: Name two examples of feedback control systems in everyday life. Define the inputs and outputs of the system, the underlying system state and dynamics, and describe the objective function. Describe the uncertainties in the system and the types of noise and disturbances that are likely experienced.

Exercise 1–2: Consider the following plant model:

$$s = b.$$

(a) Design an open-loop controller $b = K(w_r)$ to track a reference value w_r.
(b) Now, imagine that the plant model is actually $s = 2b$. How much error is there in the open-loop controller from above if we command a value $w_r = 10$?
(c) Instead of open-loop control, implement the following closed-loop controller: $b = 10(w_r - s)$. What is the error in the closed-loop system for the same command $w_r = 10$?

Exercise 1–3: Choose a major industry, such as transportation, energy, healthcare, etc., and describe an opportunity that could be enabled by closed-loop control of a turbulent fluid. Estimate the rough order of magnitude impact this would have in terms of efficiency, cost, pollution, lives saved, etc. Now, hypothesize why these innovations are not commonplace in these industries?

Chapter 2
Machine Learning Control (MLC)

All generalizations are false, including this one.

Mark Twain

In this chapter we discuss the central topic of this book: the use of powerful techniques from machine learning to discover effective control laws. Machine learning is used to generate models of a system from data; these models should improve with more data, and they ideally generalize to scenarios beyond those observed in the training data. Here, we extend this paradigm and wrap machine learning algorithms around a complex system to learn an effective control law $\mathbf{b} = \mathbf{K}(\mathbf{s})$ that maps the system output (sensors, $\mathbf{s}$) to the system input (actuators, $\mathbf{b}$). The resulting machine learning control (MLC) is motivated by problems involving complex control tasks where it may be difficult or impossible to model the system and develop a useful control law. Instead, we leverage experience and data to *learn* effective controllers.

The machine learning control architecture is shown schematically in Fig. 2.1. This procedure involves having a well-defined control task that is formulated in terms of minimizing a cost function J that may be evaluated based on the measured outputs of the system, $\mathbf{z}$. Next, the controller must have a flexible and general representation so that a search algorithm may be enacted on the space of possible controllers. Finally, a machine learning algorithm will be chosen to discover the most suitable control law through some training procedure involving data from experiments or simulations.

In this chapter, we review and highlight concepts from machine learning, with an emphasis on evolutionary algorithms (Sect. 2.1). Next (Sect. 2.2), we explore the use of genetic programming as an effective method to discover control laws in a high-dimensional search space. In Sect. 2.3, we provide implementation details and explore illustrative examples to reinforce these concepts. The chapter concludes with exercises (Sect. 2.4), suggested reading (Sect. 2.5), and an interview with Professor Marc Schoenauer (Sect. 2.6), one of the first pioneers in evolutionary algorithms.

T. Duriez et al., *Machine Learning Control – Taming Nonlinear Dynamics and Turbulence*, Fluid Mechanics and Its Applications 116,
DOI 10.1007/978-3-319-40624-4_2

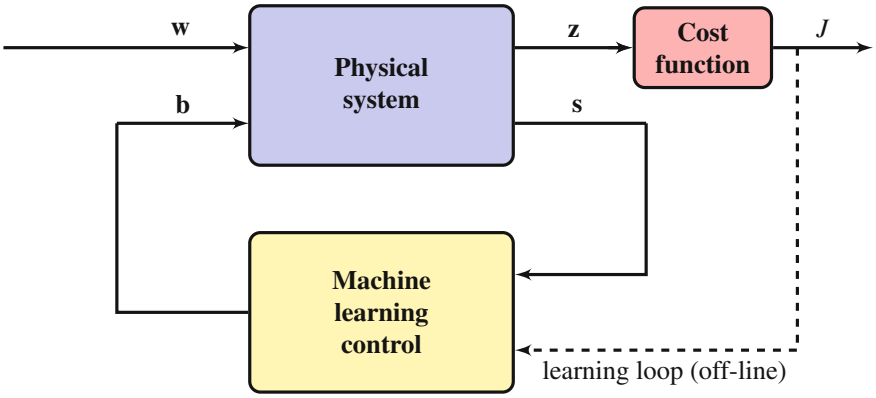


Fig. 2.1 Schematic of machine learning control wrapped around a complex system using noisy sensor-based feedback. The control objective is to minimize a well-defined cost function J within the space of possible control laws. An off-line learning loop provides experiential data to train the controller. Genetic programming provides a particularly flexible algorithm to search out effective control laws. The vector $\mathbf{z}$ contains all of the information that may factor into the cost

2.1 Methods of Machine Learning

Machine learning [30, 92, 168, 194] is a rapidly developing field at the intersection of statistics, computer science, and applied mathematics, and it is having transformative impact across the engineering and natural sciences. Advances in machine learning are largely being driven by commercial successes in technology and marketing as well as the availability of vast quantities of data in nearly every field. These techniques are now pervading other fields of academic and industrial research, and they have already provided insights in astronomy, ecology, finance, and climate, to name a few. The application of machine learning to design feedback control laws has tremendous potential and is a relatively new frontier in data-driven engineering.

In this section, we begin by discussing similarities between machine learning and classical methods from system identification. These techniques are already central in control design and provide context for machine learning control. Next, we introduce the evolutionary approaches of genetic algorithms and genetic programming. Genetic programming is particularly promising for machine learning control because of its generality in optimizing both the structure and parameters associated with a controller. Finally, we provide a brief overview of other promising methods from machine learning that may benefit future MLC efforts.

2.1.1 System Identification as Machine Learning

Classical system identification may be considered an early form of machine learning, where a dynamical system is characterized through training data. The resulting models approximate the input–output dynamics of the true system and may be used to design controllers with the methods described in Chap. 3. The majority of methods in system identification are formulated for linear systems and provide models of dubious quality for systems with strongly nonlinear dynamics. There are, however, extensions to linear parameter varying (LPV) systems, where the linear system depends on a time-varying parameter [16, 247].

There is an expansive literature on system identification, with many techniques having been developed to characterize aerospace systems during the 1960s to the 1980s [150, 174]. The eigensystem realization algorithm (ERA) [151, 181] and observer Kalman filter identification (OKID) [152, 213] techniques build input–output models using time-series data from a dynamical systems; they will be discussed more in Chap. 3. These methods are based on time-delay coordinates, which are reminiscent of the Takens embedding [260]. The singular spectrum analysis (SSA) from climate science [7, 36–38] provides a similar characterization of a time-series but without generating input–output models. Recently SSA has been extended in the nonlinear Laplacian spectral analysis (NLSA) [117], which includes kernel methods from machine learning.

The dynamic mode decomposition (DMD) [229, 238, 270] is a promising new technique for system identification that has strong connections to nonlinear dynamical systems through Koopman spectral analysis [41, 47, 162, 163, 170, 187, 188, 282]. DMD has recently been extended to include sparse measurements [45] and inputs and control [217]. The DMD method has been applied to numerous problems beyond fluid dynamics [229, 238], where it originated, including epidemiology [218], video processing [99, 124], robotics [25], and neuroscience [40]. Other prominent methods include the autoregressive moving average (ARMA) models and extensions.

Decreasing the amount of data required for the training and execution of a model is often important when a fast prediction or decision is required, as in turbulence control. Compressed sensing and machine learning have already been combined to achieve *sparse decision making* [39, 46, 169, 216, 279], which may dramatically reduce the latency in a control decision. Many of these methods combine system identification with clustering techniques, which are a cornerstone of machine learning. Cluster-based reduced-order models (CROMs) are especially promising and have recently been developed in fluids [154], building on cluster analysis [49] and transition matrix models [241].

In the traditional framework, machine learning has been employed to model the input–output characteristics of a system. Controllers are then designed based on these models using traditional techniques. Machine learning control circumvents this process and instead directly learns effective control laws without the need for a model of the system.

2.1.2 Genetic Algorithms

Evolutionary algorithms form an important category of machine learning techniques that adapt and optimize through a process mimicking natural selection. A population of individuals, called a generation, compete at a given task with a well-defined cost function, and there are rules to propagate successful strategies to the next generation. In many tasks, the search space is exceedingly large and there may be multiple extrema so that gradient search algorithms yield sub-optimal results. Combining gradient search with Monte Carlo may improve the quality of the solution, but this is extremely expensive. Evolutionary algorithms provide an effective alternative search strategy to find nearly optimal solutions in a high-dimensional search space.

Genetic algorithms (GA) are a type of evolutionary algorithm that are used to identify and optimize parameters of an input–output map [76, 122, 137]. In contrast, genetic programming (GP), which is discussed in the next section, is used to optimize both the structure and parameters of the mapping [164, 166]. Genetic algorithms and genetic programming are both based on the propagation of generations of individuals by selection through fitness. The individuals that comprise a generation are initially populated randomly and each individual is evaluated and their performance assessed based on the evaluated cost function. An individual in a genetic algorithm corresponds to a set of parameter values in a parameterized model to be optimized; this parameterization is shown in Fig. 2.2. In genetic programming, the individual corresponds to both the structure of the control law and the specific parameters, as discussed in the next section.

After the initial generation is populated with individuals, each is evaluated and assigned a fitness based on their performance on the cost function metric. Individuals with a lower cost solution have a higher fitness and are more likely to advance to the next generation. There are a set of rules, or genetic operations, that determine how successful individuals advance to the next generation:

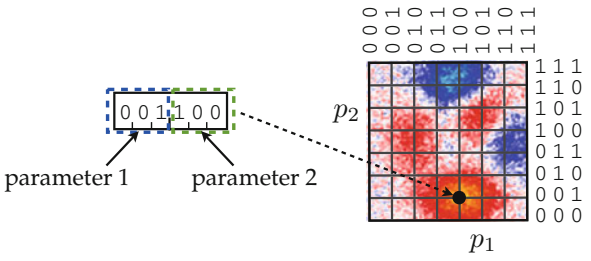


Fig. 2.2 Representation of an individual (parameter) in genetic algorithms. This binary representation encodes two parameters that are each represented with a 3-bit binary expression. Each parameter value has an associated cost (*right*), with *red* indicating the lowest cost solution. Modified from Brunton and Noack, *Applied Mechanics Reviews*, 2015 [43]

Elitism: a handful of the most fit individuals advance directly to the next generation. Elitism guarantees that the top individuals from each generation do not degrade in the absence of noise.

Replication: individuals advance directly to the next generation with a probability related to fitness; also called asexual reproduction in genetic programming.

Crossover: two individuals are selected based on their fitness and random sections of their parameters are exchanged. These two individuals advance with the exchanged information.

Mutation: individuals advance with random portions of their parameter representation replaced with random new values.

Mutation serves to explore the search space, providing access to global minima, while crossover serves to exploit successful structures and optimize locally. Successful individuals from each generation advance to the next generation through these four genetic operations. New individuals may be added in each generation for variety. This is depicted schematically for the genetic algorithm in Fig. 2.3. Generations are evolved until the performance converges to a desired stopping criterion.

Evolutionary algorithms are not guaranteed to converge to global minima. However, they have been successful in many diverse applications. It is possible to improve the performance of evolutionary algorithms by tuning the number of individuals in a generation, the number of generations, and the relative probability of each genetic operation. In the context of control, genetic algorithms are used to tune the parameters of a control law with a predefined structure. For example, GA may be used to tune the gains of a proportional-integral-derivative (PID) control law [271].

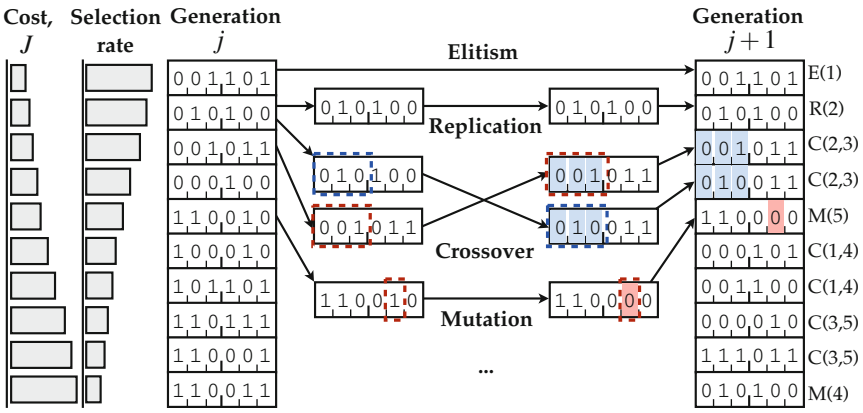


Fig. 2.3 Genetic operations to advance one generation of parameters to the next in a genetic algorithm. The probability of an individual from generation j being selected for generation $j+1$ is related inversely to the cost function associated with that individual. The genetic operations are elitism, replication, crossover, and mutation. Modified from Brunton and Noack, *Applied Mechanics Reviews*, 2015 [43]

2.1.3 Genetic Programming

Genetic programming (GP) [164, 166] is an evolutionary algorithm that optimizes both the structure and parameters of an input–output map. In the next section, GP will be used to iteratively learn and refine control laws, which may be viewed as nonlinear mappings from the outputs of a dynamical system (sensors) to the inputs of the system (actuators) to minimize a given cost function associated with the control task.

The mapping discovered by genetic programming is represented as a recursive function tree, as shown in Fig. 2.4 for the case of a control law $b = \mathbf{K}(\mathbf{s})$. In this representation, the root of the tree is the output variable, each branching point is a mathematical operation, such as $+$, $-$, $\times$, $/$, and each branch may contain additional functions. The leaves of the tree are the inputs and constants. In the case of MLC the inputs are sensor measurements and the root is the actuation signal.

Genetic programming uses the same evolutionary operations to advance individuals across generations that are used in genetic algorithms. The operations of replication, crossover, and mutation are depicted schematically in Fig. 2.5 for genetic programming. As in other evolutionary algorithms, the selection probability of each genetic operation is chosen to optimize the balance between exploration of new structures and exploitation of successful structures.

In the sections and chapters that follow, we will explore the use of genetic programming for closed-loop feedback control. In particular, we will show that using genetic programming for machine learning control results in robust turbulence control in extremely nonlinear systems where traditional control methodologies typically fail. We will also generalize the inputs to include time-delay coordinates on the sensor measurements and generalize the function operations to include filter functions to emulate dynamic state estimation.

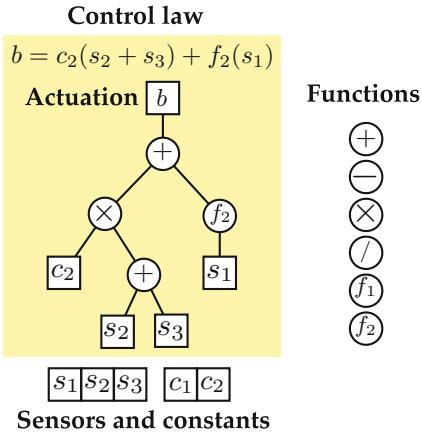


Fig. 2.4 Individual function tree representation used in genetic programming. Modified from Brunton and Noack, *Applied Mechanics Reviews*, 2015 [43]

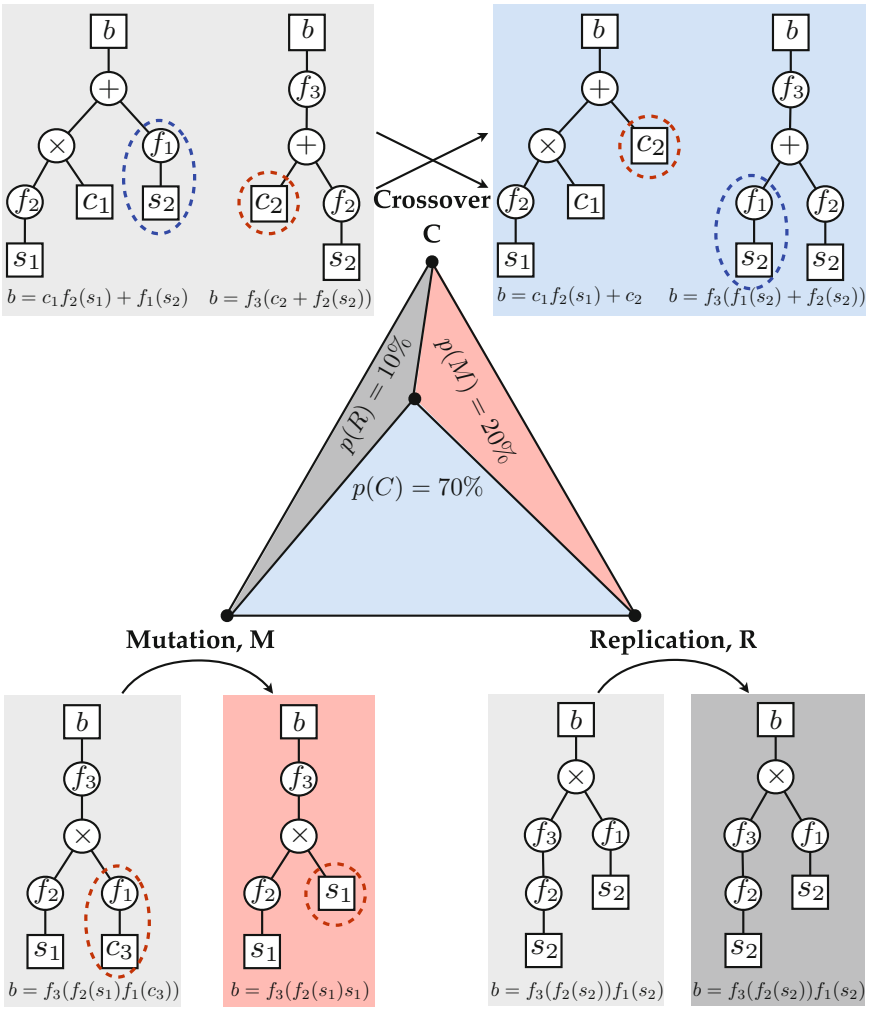


Fig. 2.5 Genetic operations to advance one generation of function trees to the next generation in a genetic program. The operations that advance individuals from one generation to the next are crossover, mutation, and replication. In crossover, random branches from two individual expressions are exchanged. In mutation, a branch is randomly selected and replaced with another randomly generated branch. In replication, the individual is copied directly to the next generation. Modified from Brunton and Noack, *Applied Mechanics Reviews*, 2015 [43]

2.1.4 Additional Machine Learning Methods

There is a vast and growing literature on machine learning. This section only provides a brief overview of some of the most promising methods that may be used in machine learning control efforts. More extensive treatments may be found in a number of excellent texts [30, 92, 168, 194]. In addition, there is a good overview of the top ten methods in data mining [280].

The presentation on genetic programming above provides a high-level overview of the method, which may be directly implemented for machine learning control. In reality, there is an entire field of research expanding and developing these methods. Genetic programming has been used to discover governing equations and dynamical systems directly from measurement data [32, 220, 240]. There are variants on genetic programming using the elastic net [185], which result in fast function identification. It is also possible to use sparse regression [146, 266] to identify nonlinear dynamical systems from data [44]; this method is related to the recent use of compressed sensing for dynamical system identification [275].

Another promising field of research involves artificial neural networks (ANNs). ANNs are designed to mimic the abstraction capabilities and adaptability found in animal brains. A number of individual computational units, or *neurons*, are connected in a graph structure, which is then optimized to fit mappings from inputs to outputs. It is possible to train the network with stimulus by modifying connections strengths according to either supervised or unsupervised reinforcement learning. There are many approaches to modify network weights, although gradient search algorithms are quite common [61, 129]. Neural networks have been used in numerous applications, including to model and control turbulent fluids [95, 171, 189, 193]. Network-theoretic tools have been applied more generally in fluid modeling recently [154, 195]. ANNs have also been trained to perform principal components analysis (PCA), also known as proper orthogonal decomposition (POD) [203], as well as nonlinear extensions of PCA [157, 204].

Neural networks have proven quite adaptable and may be trained to approximate most input–output functions to arbitrary precision with enough layers and enough training. However, these models are prone to overfitting and require significant amounts of training data. Recently, neural networks have seen a resurgence in research and development with the associated field of deep learning [69, 78, 132]. These algorithms have shown unparalleled performance on challenging tasks, such as image classification, leveraging large data sets collected by corporations such as Google, etc. This is a promising area of research for any data-rich field, such as turbulence modeling and control, which generates tremendous amounts of data.

There are many other important machine learning algorithms. Support vector machines (SVMs) [242, 253, 259] are widely used because of their accuracy, simple geometric interpretation, and favorable scaling to systems with high-dimensional input spaces. Decision trees [222] are also frequently used for classification in machine learning; these classifications are based on a tree-like set of decisions, providing simple and interpretable models. Multiple decision tree models may be

combined, or *bagged*, resulting in a random forest model [33]. Ensemble methods in machine learning, including *bagging* and *boosting*, have been shown to have significantly higher classification accuracy than that of an individual classifier [83, 105, 237].

Many of the foundational methods in big data analysis [131] have been applied largely to static data problems in artificial intelligence and machine vision. There is a significant opportunity to leverage these techniques for the modeling and control of dynamical systems. These methods may be used for clustering and categorical decisions, dimensionality reduction and feature extraction, nonlinear regression, and occlusion inference and outlier rejection. Machine learning is having transformative impact across the scientific and engineering disciplines. There is tremendous opportunity ahead to employ machine learning solutions to modern engineering control.

2.2 MLC with Genetic Programming

Now we use genetic programming (GP) as a search algorithm to find control laws in MLC. This section provides details about GP in a context that is specific to control.

2.2.1 Control Problem

Before applying any machine learning, it is necessary to pose the control problem as a well-defined cost minimization. In particular, the performance of a given control law is judged based on the value of a cost function J, and the machine learning algorithms will serve to minimize this cost.

There are many ways to formulate a cost function in order to promote different control objectives. In fact, cost function optimization is the basis of most modern control approaches [93, 252], which will be discussed more in Chap. 3. Consider a simplified cost function that depends on the state $\mathbf{a}$ and the actuation $\mathbf{b}$:

$$J(\mathbf{a}, \mathbf{b}). \tag{2.1}$$

We often assume that the effects of the state and actuation on the cost are separable:

$$J(\mathbf{a}, \mathbf{b}) = J_a + \gamma\, J_b, \tag{2.2}$$

where J_a is a performance measure on the state of the system and J_b is a value associated with the cost of actuation. The *penalization parameter* γ provides an explicit tuning knob to give priority to either the actuation cost (γ large) or the state cost (γ small). More objectives can be added to the cost function J by including norms on the various transfer functions from inputs to outputs; these may promote good reference tracking, disturbance rejection, noise attenuation, robustness, etc., and

a good treatment of these choices is provided in [252]. In addition, the complexity of the controller **K** may be penalized to avoid overfitting.

2.2.2 Parameterization of the Control Law

When using genetic programming for MLC, the control laws are represented as recursive expression trees (also known as *function trees*), as shown in Fig. 2.6. These control laws are the *individuals* that will populate a generation in genetic programming. Expression trees are usually built from a number of elementary functions which can take any number of arguments but return a single value; example function nodes are $+, -, \times, /$, sin, tanh, ... The arguments of these functions may be leaves or subtrees. In MLC, the root of the tree is the actuation signal b and the leaves are components of the sensor vector **s** or constants.

The tree in Fig. 2.6 represents the function:

$$b(s_1, s_2) = \cos(s_1) + \tanh((s_1 \times s_2)/0.21) - 2.32, \tag{2.3}$$

where s_1 and s_2 are the time-varying sensor values. It is useful to represent the function tree as a LISP (LISt Processor) expression.

Several representations can be used to manipulate functions inside the genetic programming architecture (e.g. trees, linear programming). We choose to use a tree-like representation. Two main advantages of this representation are that expression

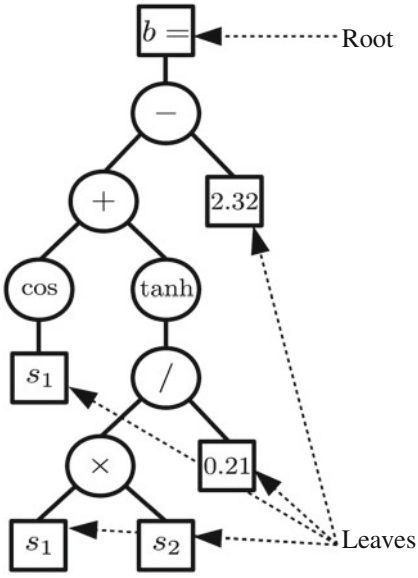


Fig. 2.6 An expression tree representing the controller function given by: $b = \mathbf{K}(\mathbf{s}) = \cos(s_1) + \tanh((s_1 \times s_2)/0.21) - 2.32$

trees are readily interpretable, and they are easily synthesized and manipulated computationally using a recursive language such as LISP or Scheme.

Equation (2.3) can be written as the LISP string in parenthesized Polish prefix notation:

$$(-\ (+\ (\cos\ \ s_1)\ (\tanh\ (/\ (\times\ s_1\ s_2)\ 0.21)))\ 2.32).$$

Though less readable than the expression tree, recursive algorithms can generate, derive, manipulate, and evaluate these expressions. The generation process of any individual control law starts at the root. Then a first element is chosen from the pool of admissible basis functions and operators. If a basis function or operator is chosen, new elements are added as arguments, and the process is iterated to include their arguments until all branches have leaves.

This process of individual generation is subject to limitations. A given tree-depth (the maximum distance between any leaf and the root) can be prescribed by preventing the last branch from generating a leaf before the aforementioned tree-depth is reached and by enforcing the termination of the branch in leaves when the tree-depth is reached. Similarly it is possible to ensure that each branch reaches the same given tree-depth which generates a full-density tree with the maximum number of operations. MLC can implement any of these distributions, from fully random trees to a given tree-depth distribution with a specified proportion of dense and less dense individuals. The first generation starts with a distribution of rather low tree-depth (2 to 8) and an equal distribution (1:1) of dense and less dense individuals in the default parameters of `OpenMLC`. This choice generally ensures enough diversity for the creation of subsequent generations. The initially low tree-depth takes into account that the genetic operations (see Sect. 2.2.7) have a tendency to make the trees grow. This phenomenon is known as bloating of the trees. To enforce more diversity in the population, any candidate individual is discarded if it already exists in the current population.

2.2.3 Genetic Programming as a Search Algorithm

The flowchart for genetic programming is given in Fig. 2.7. An initial set (*generation*) of N_i control laws (*individuals*) is evaluated according to the cost function J. Next, successful individuals are selected to advance to the next generation and are evolved by genetic operations: elitism, replication, crossover and mutation. This procedure is repeated until a convergence or stopping criterion is met.

The implementation of genetic programming as a search algorithm for MLC is shown in Fig. 2.8. In this schematic, control laws are expression trees that take sensor outputs $\mathbf{s}$ of a dynamical system and synthesize actuation inputs $\mathbf{b}$. These controller individuals are optimized through the genetic programming algorithm.

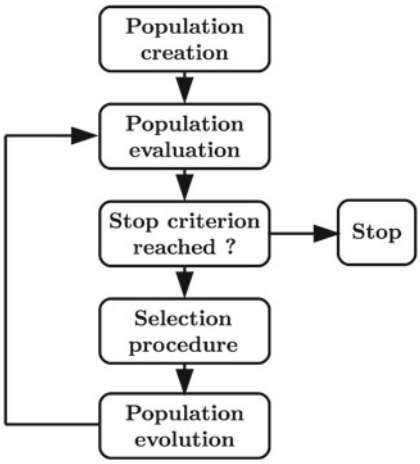


Fig. 2.7 Flowchart for the genetic programming algorithm

Advanced material 2.1 Operation protection.
It is important to note that not all functions are defined for all real-valued arguments. For instance, the second argument of the division operator must not be zero and the argument of the logarithmic functions must be strictly positive. Thus, such functions must be protected. When the translation from LISP is achieved using the `OpenMLC` toolbox through the `readmylisp_to_formal_MLC.m` function, '(/ arg1 arg2)' is interpreted as 'my_div(arg1,arg2)', where `my_div.m` defines the function:

$$\begin{aligned}\text{my\_div}(arg1, arg2) &= \frac{arg1}{arg2}, && \text{if } |arg2| > 10^{-3}\\ &= \frac{arg2}{|arg2|}\frac{arg1}{10^{-3}}, && \text{if } 0 < |arg2| < 10^{-3}\\ &= \frac{arg1}{10^{-3}}, && \text{if } |arg2| = 0.\end{aligned}$$

Similarly, '(log arg1)' is interpreted as 'my_log(arg1)', where `my_log.m` defines the function:

$$\begin{aligned}\text{my\_log}(arg1) &= \frac{arg1}{|arg1|}\log(|arg1|), && \text{if } |arg1| > 10^{-3}\\ &= \frac{arg1}{|arg1|}\log(10^{-3}), && \text{if } 0 < |arg1| < 10^{-3}\\ &= 0, && \text{if } |arg2| = 0.\end{aligned}$$

In case the interpretation of the LISP expression is carried out by another function, for example on a real-time processing unit, these protections have to be implemented in order to avoid unexpected behaviors. These protections can be easily changed by editing both surrogate functions, or by defining another function to be called in the `parameters.opset` structure.

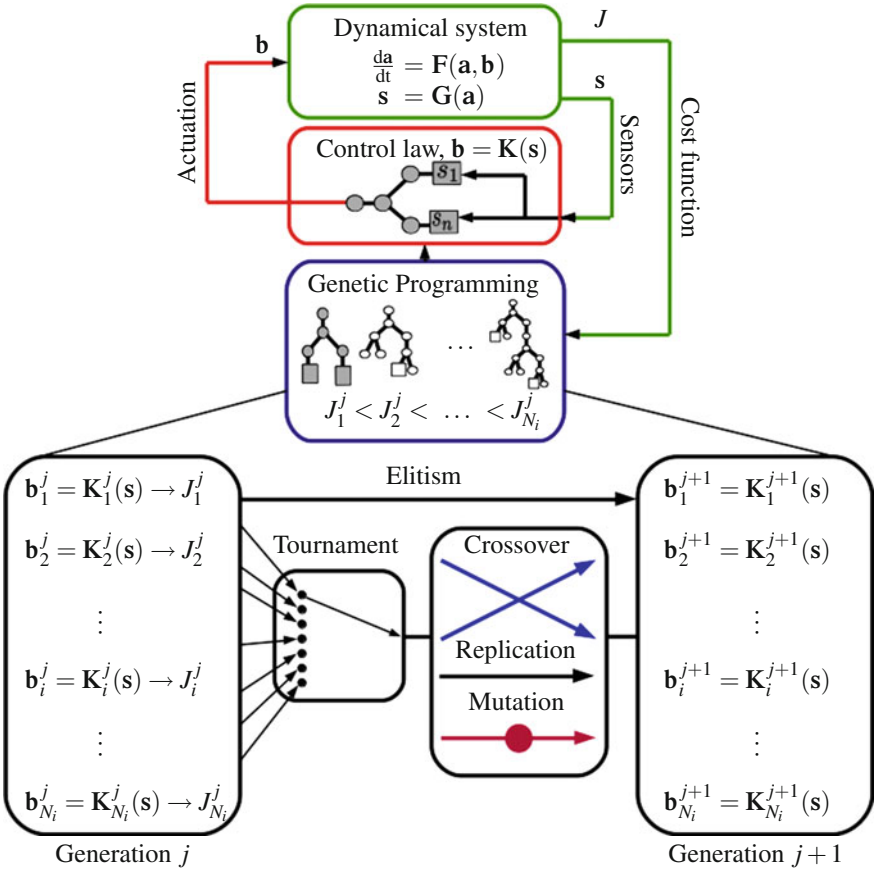


Fig. 2.8 Model-free control design using GP for MLC. During the learning phase, each control law candidate is evaluated by the dynamical system or experimental plant. This process is iterated over many generations of individuals. At convergence, the best individual with the smallest cost function value (in *grey* in the GP box) is used for real-time control

2.2.4 Initializing a Generation

In genetic programming an entire population of individuals forms a generation, and these individuals compete to advance to future generations. A population contains N_i individuals that must be initialized. To form an expression tree, the initialization algorithm works from the root to the trees. The principle is to work from one seed (a marker which indicates where the tree is supposed to grow) or N_b seeds (if the actuation input **b** has multiple components), decide on a node (function/operation, or leaf), add as many new seeds as necessary (if this is an operation or function, include as many seeds as arguments) and recursively call the function that grows the tree until all new seeds have been generated. The process stops once all seeds

have been replaced by leaves. Those leaves are chosen between randomly generated constants and one of the N_s sensors in **s**. This algorithm can be configured so that some statistical properties of the population can be specified:

Tree-depth: If an operation or a leaf is selected through a probabilistic process, it is possible to prescribe the minimal and maximal tree-depth of the trees by forcing or forbidding the creation of leaves according to the tree-depth and/or the existence of other leaves at said depth.

Density: It is possible to prescribe a specific tree depth for all leaves. This is achieved by forbidding any leaf before a given depth, and forcing leaf selection at that depth. The resulting individuals possess the maximal number of operations possible (modulo the number of arguments of the selected operations) for the prescribed tree depth. Such an individual is referred to as a full (density) individual.

The initial population corresponds to the first exploration of the search space. As such, this population has a crucial impact on the following steps of the search process: future generations will converge around the local minima found in the initial generation. Therefore, it is important that this population contains as much diversity as possible. A first measure is to reject any duplicate individual in the population. Diversity is also enforced using a Gaussian distribution of the tree-depth (between 2 and 8 by default in `OpenMLC`), with half the individuals having full density.

2.2.5 Evaluating a Generation

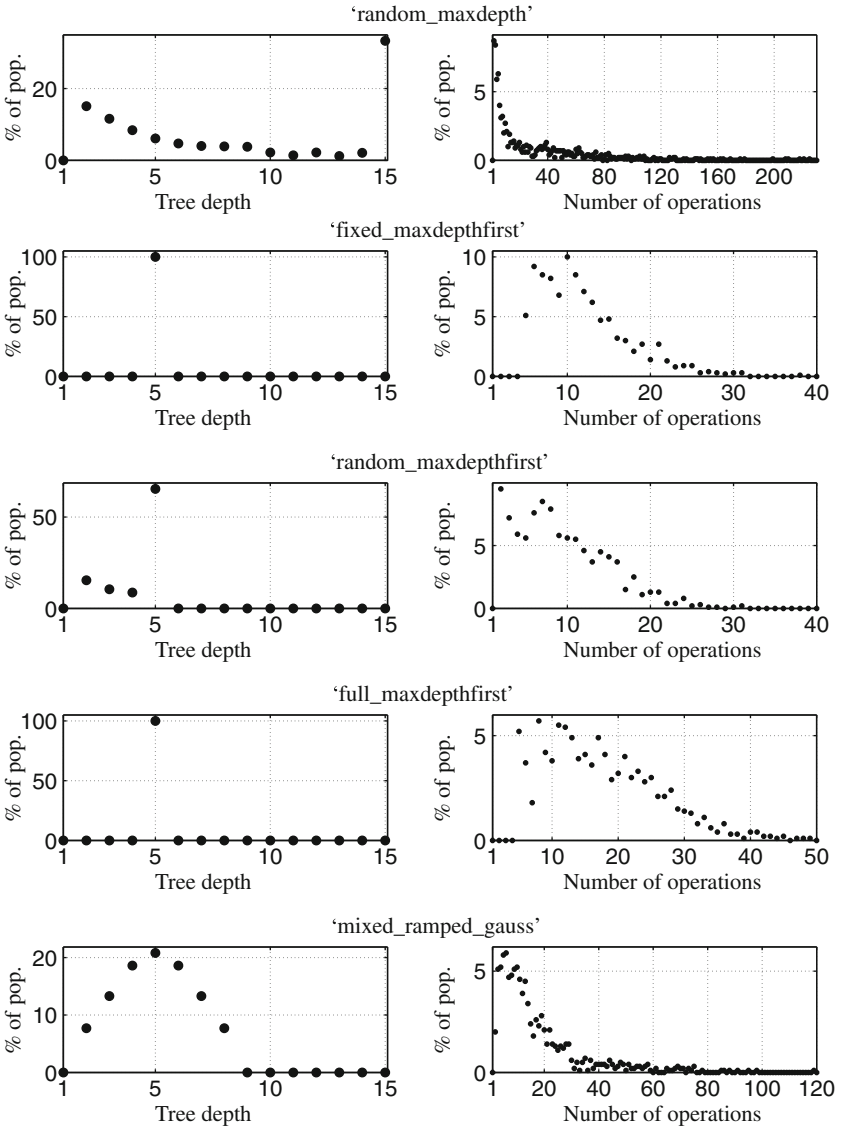
After every new generation is formed, each individual must be evaluated based on their performance with respect to the regression problem. The value of the cost

Advanced material 2.2 Creation algorithm pseudo-code.
The function that generates expression trees as LISP expressions in `OpenMLC` is *generate_indiv_regressive.m*. As any expression tree manipulation function, the creation algorithm is auto-recursive. The principle is to work from a seed, decide on a node (function/operation, or leaf), add as many seeds as necessary (if this is an operation or function) and call the function back as many time as new seeds have been generated:

1 : **new_LISP=***grow_tree*(**old_LISP**,**parameters**) % declaration
2 : find first seed in old_LISP, so that **old_LISP=**'part1 seed part2'
3 : decide if the next node is an operation or a leaf
    if it is a leaf: replace the seed with either a constant, either a sensor
        and return **new_LISP=**'part1 leaf part2'
    if it is an operation: choose one randomly (called 'op') and replace seed so that:
    new_LISP='part1 (op seed) part2' if 'op' takes one argument
    new_LISP='part1 (op seed seed) part2' if 'op' takes two arguments
4 : recursively call back **new_LISP=***grow_tree*(**new_LISP**,**parameters**) as many times as new seeds have been added
5 : return **new_LISP**

An expression tree with N_b independent subtrees (corresponding to N_b actuation inputs) can be created using the tree growing function on '(root seed repeated N_b times)'.

Advanced material 2.3 Effect of generation parameters in the first population diversity. The tree-depth distribution is shown for each initialization method for the first generation (left). The histograms of the number of operations for each tree-depth is also shown (right). The average is marked by a bullet. These graphs are obtained with one instance of a population generation with default `OpenMLC` object properties except for the initialization method which is indicated in each graph. One indicator of diversity is the distribution of operations in the trees. If multiplying the different tree depth is a good factor to enforce diversity, the natural growth of the average tree-depth as new generations are evolved (a phenomenon known as *bloat*) indicates that it is better to keep low tree-depth in the first generation.



function J in Eq. (2.1) is computed for each individual. In MLC, this evaluation corresponds to the individual being used as the control law for the dynamical system under consideration. This evaluation results in a single value J, which is the result of Eq. (2.1) for the specific individual being evaluated.

A major complication encountered with experiments (or even some noisy numerical systems) is the fact that, contrary to deterministic systems, the re-evaluation of one individual does not necessarily return the same cost function value as a previous evaluation. A large error on the value, either due to a measurement error or an exceptional performance of a non-robust control law can lead to a non-representative grading of the control law. As explained throughout the book, the search space is primarily explored around the best-performing individuals. If an intrinsically low-performing individual gets a mistakenly good evaluation, this may be a significant setback. Thus, all individuals are evaluated even if they have already been evaluated in a previous generation, and the best individuals undergo a re-evaluation. By default in `OpenMLC`, the five best individuals are each re-evaluated five times, and their cost function is averaged. This procedure ensures that the best performing individuals are more carefully ranked so that the search process is not misdirected.

2.2.6 *Selecting Individuals for Genetic Operations*

After the evaluation of each individual, the population evolution starts. In order to fill the next generation, genetic operations (see Sect. 2.2.7) are performed on selected individuals. The selection procedure is at the heart of any evolutionary algorithm as it determines the genetic content of the following generation. The selection process employed by default is a tournament. Each time an individual needs to be selected for a genetic operation, N_p unique individuals are randomly chosen from the previous generation to enter the one-round tournament. From this set of individuals, the one with the smallest cost function value is selected. As the population size N_i is fixed, N_i selection tournaments are run each time a new generation is created. The population is ranked by decreasing cost function value. If we discard uniqueness of the individuals and consider $N_i \gg N_p > 1$, the probability of individual i winning a tournament is $((N_i - i)/(N_i - 1))^{N_p - 1}$. On average, each individual will enter N_p tournaments. Each individual is sorted by their ranking, so that individual i has the ith lowest cost function value; we define $x = i/N_i$ for each individual so that $x \in [0, 1])$, where $x = 0$ is the best individual and $x = 1$ is the worst individual. If an individual is ranked $i = x \times N_i$, then its genetic content will contribute on average roughly to $N_p \times (1 - x)^{N_p - 1}$ new individuals. A standard selection parameter sets $N_p = 7$ and ensures that only the first half of the ranked generation contributes consistently to the next generation. Lower-performing individuals can still contribute, but these are rare events. In this selection procedure, choosing N_p sets the harshness of the selection. On the other hand, this selection procedure does not take into account the global distribution of the cost function values, as in other selection processes such as a fitness proportional selection. If an individual performs much better than the rest

of the population it will not have a better selection chance than an individual that barely performs better than the rest of the population. This choice ensures that the diversity in the population does not disappear too fast with an associated trade-off in convergence speed.

2.2.7 Selecting Genetic Operations

Four kinds of genetic operations are implemented: *elitism*, *replication*, *mutation* and *crossover*.

Elitism: The N_e best individuals of the evaluated population are copied directly to the next generation. This operation does not go through a selection procedure and ensures that the best control laws stay in the population. Once the elitism process is finished, $N_i - N_e$ individuals have to be generated through replication, mutation and crossover. The probability of each of these operations are P_r, P_m and P_c, respectively, with $P_r + P_m + P_c = 1$.

Replication: The selected individual is copied to the next generation.

Mutation: There are four mutation operations: *Cut and grow* replaces an arbitrarily chosen subtree by a randomly generated new subtree. For that, the same procedure is used as in the creation of the first generation. *Shrink* replaces an entire randomly chosen subtree by a randomly chosen leaf (constant or sensor). *Hoist* replaces the tree by a randomly chosen subtree. If this tree corresponds to a MIMO control law with N_b actuation inputs, each control law has a $1/N_b$ chance to be mutated if the hoist mutation is chosen. *Reparametrization* sets a 50 % chance for each constant to be randomly replaced with a new value. All of these mutations are displayed in Figs. 2.9 and 2.10.

Advanced material 2.4 Fitness proportional selection.
Fitness proportional selection is another popular process for selecting individuals for genetic operations. The inverse of an individual's cost $J_i = J(K_i)$ is a natural measure of its desirability. It goes to infinity as the optimal value zero is reached. The probability of the section of the ith individual is set proportional to this desirability

$$P_i = \frac{J_i^{-1}}{\sum_{j=1}^{N_i} J_j^{-1}}. \tag{2.4}$$

If one individual performs much better than the rest of the population, it will be selected more often in the same proportion, thus encouraging optimization around the best performing individuals while still allowing sub-optimal individuals to be selected. However, diversity can disappear rapidly if an individual performs one or several orders of magnitude better than the rest of the population. This is an undesirable feature which we avoid by a selection based on the relative ordering.

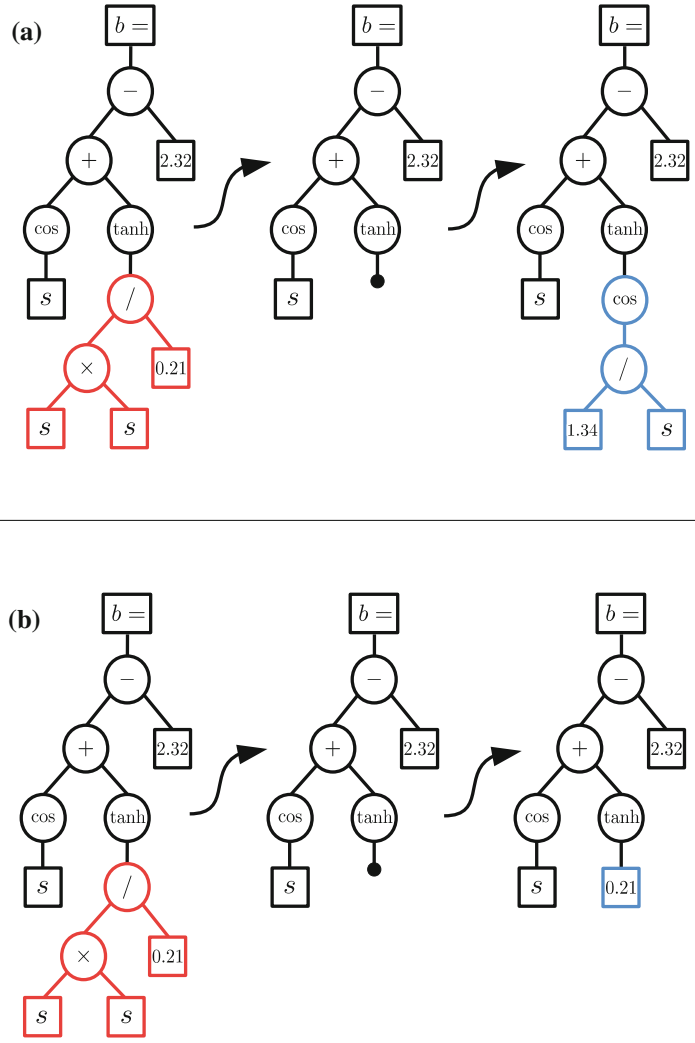


Fig. 2.9 Two of the four types of mutations implemented in MLC: (**a**) cut and grow, (**b**) shrink

Crossover: uses two selected individuals and exchanges one randomly chosen subtree between them (see Fig. 2.11).

Replication ensures some stability of the convergence process: it guarantees that a part of the population stays in the vicinity of explored local minima of the search space, keeping potentially useful mechanisms in the population and further exploiting them before they are discarded. Crossover and mutation are responsible for the exploitation and exploration of the search space, respectively. As we progress among the generations, the probability to cross similar individuals increases: the best indi-

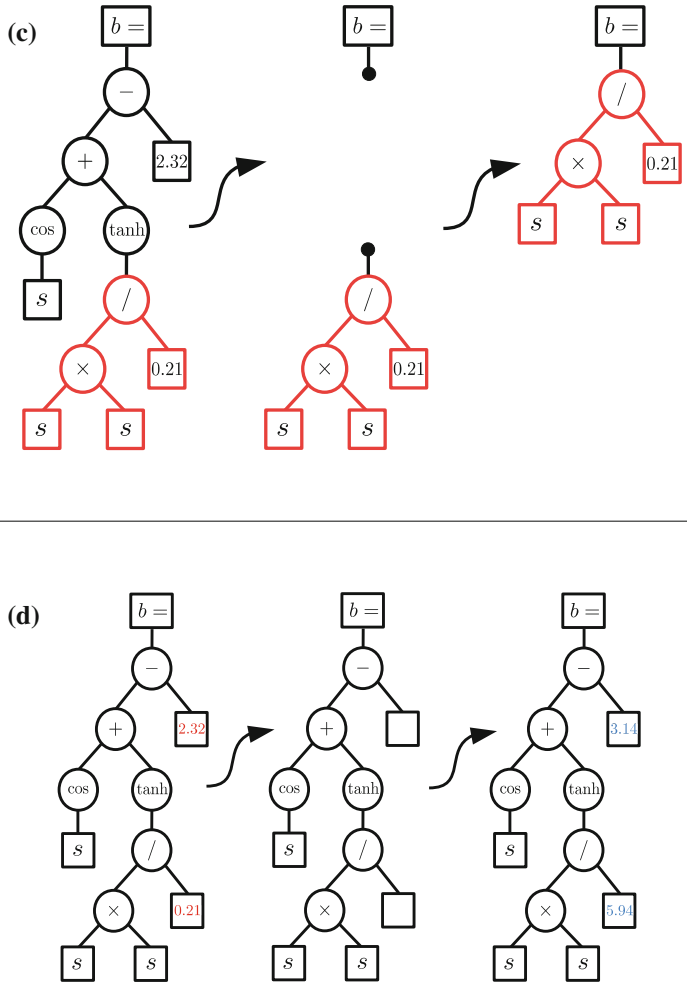


Fig. 2.10 Two of the four types of mutations implemented in MLC: (**c**) Hoist and (**d**) re-parameterization

vidual will propagate its genetic content N_p times on average. If this successful genetic content allows the concerned individuals to stay in the first positions of the ranking, it will be replicated about $N_p^k \times P_c$ times after k generations. Then crossovers of individuals selected in the top of the ranking will soon cross similar individuals and explore the vicinity of the dominant genetic content. On the other hand, mutations introduce new genetic content in the population, hence allowing large-scale exploration of the search space. Figure 2.12 illustrates how an evolutionary algorithm explores the search space of a two-dimensional problem with local minima:

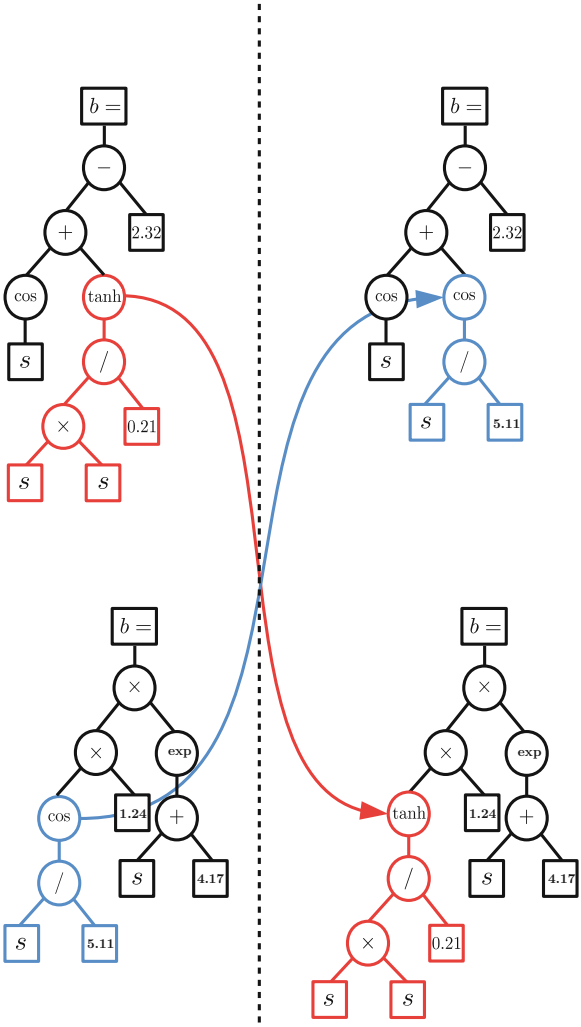


Fig. 2.11 Crossover example. The selected individuals of a considered generation (*left*) exchange subtrees to form a new pair in the next generation (*right*)

the association of these operations enables exploitation around the local minima while still exploring the search space for better solutions.

2.2.8 Advancing Generations and Stopping Criteria

There are no fool-proof general rules to choose optimal parameters for evolutionary algorithms. A common practice is to check the optimality of the solution offered by genetic programming by reproducing the process a number of times using different

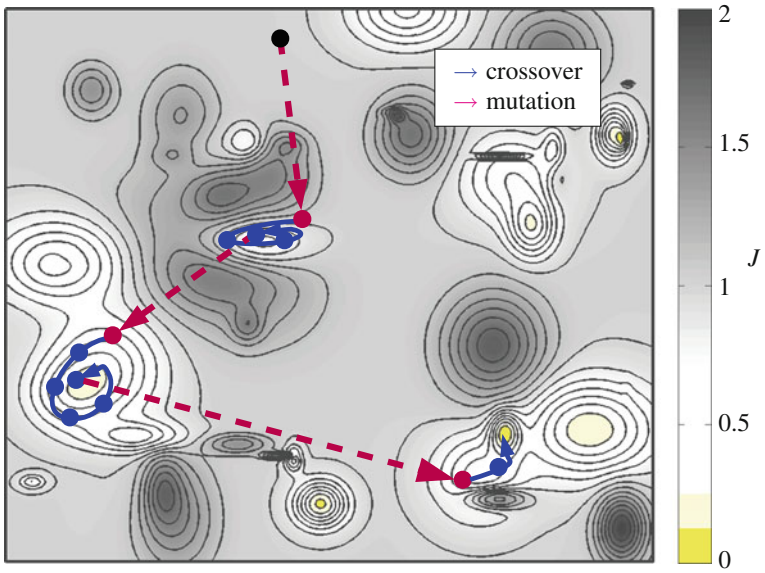


Fig. 2.12 Conceptual 2-dimensional representation of the search process of evolutionary algorithms. The level curves display the geometry of the cost function and its local minima and maxima. The *arrows* follow one branch of the lineage of one individual (displayed by a black point) that leads to the global minimum of the cost function. The exploitation of the local minima is achieved by crossovers while the large-scale exploration of the search space is achieved by the mutations

sets of parameters. This way, guiding statistics can be obtained. In the case of experimental turbulence control, one is more interested in finding an effective control law than in determining an optimal search algorithm. The main impact of modifying parameters is on the ratio between exploitation (i.e. convergence) and exploration of the search space. Monitoring the evolution of the evaluated populations is the best way to fine-tune the MLC process. Now, we discuss the role of the MLC parameters:

- Population size N_i: more individuals in the first generation will result in more exploration of the search space. On the other hand, a large initial population requires more evaluation time without any evolutionary convergence. Let us consider 1000 evaluations. If only the first generation is evaluated, then it is equivalent to a Monte Carlo process. Alternatively, one could devote 1000 evaluations to 20 generations with 50 individuals in each generation. This implies 20 iterative refinements of the best individuals through evolutionary operations.
- Genetic operation probabilities (N_e/N_i, P_r, P_m, P_c): Elitism is encouraged as it ensures that the N_e best performing individuals will remain in the population. Replication, mutation and crossover probabilities parametrize the relative importance between exploration and exploitation (Fig. 2.13). A large mutation rate P_m implies large-scale exploration and thus population diversity. If all individuals in the population share a similar expression and have similar costs then the muta-

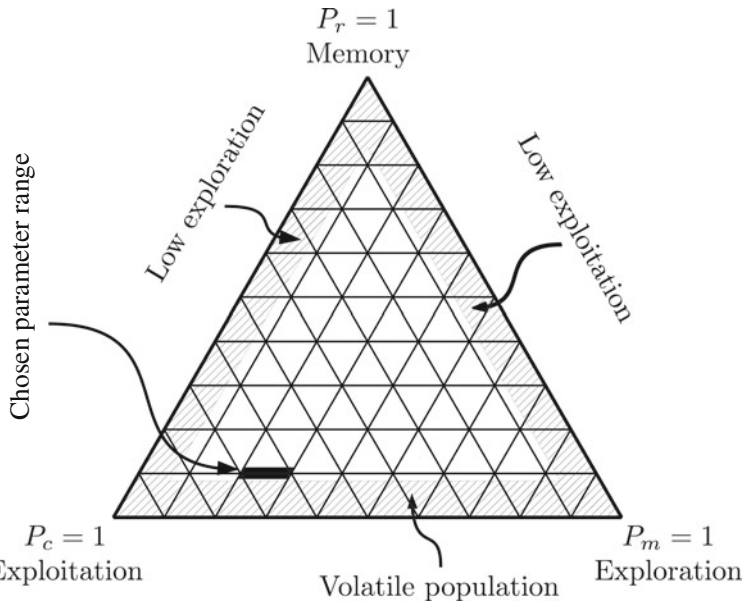


Fig. 2.13 Probability selection for the genetic operations. There is no globally optimal parameter selection. Depending on the problem, diversity or convergence needs to be modified. The range of parameters used in the present study is represented by the *black area*. *Hatched areas* represent the parametrical space where essential aspects of the evolutionary algorithm are endangered with volatile population translating in previous winning options to be forgotten once a better solution is found (not enough replications), low convergence (not enough crossovers) or low exploration (not enough mutations)

tion rate should be increased. On the other hand, the crossover rate will impact the convergence speed. If the individuals are different and the cost function value histogram does not show a peak around the lowest value, then the convergence is not sufficiently rapid and the crossover probability should be increased. Finally, replication ensures that a given search space area will be explored during a certain number of generations. This parameter will at the same time ensure diversity and exploration of different areas.

- The harshness of the selection procedure also influences the diversity of the next generation. The number of individuals N_p that enter a tournament directly influences the number of individuals that will contribute to the next generation. Reducing N_p increases the diversity while increasing it will accelerate the convergence.
- The choice of the elementary functions is intimately linked to the regression problem. This choice should be determined in concordance with the choice of the sensors and actuators.
- The maximum number of generations is eventually determined by the available testing time. A stopping criterion can end the iterations prematurely, for instance

if the optimal solution is reached ($J = 0$) or if the average and minimum of the cost function distribution converge.

GP experts usually recommend a high rate of crossover and a low rate of mutation with a large initial population, though specific values depend on the problem. These choices, clearly aimed to obtain a fast convergence, are possible when the evaluations can be parallelized. In experiments, however, the total time to evaluate a generation of individuals is critical and one cannot afford a large population. The convergence of the cost function evaluation is also affected by the measurement error. A good compromise in experiments is to deal with reduced populations (on the order of 50–500 individuals) associated with a high mutation rate (from 25 to 50 %). It is simpler to keep these values constant during the course of each experiment, but further performance improvement can be achieved by adapting them with respect to the phase (exploration or exploitation) of the learning process.

Throughout the book many examples of applications of MLC to different control problems can be found, from numerical dynamical systems to experiments, and their specific implementation using `OpenMLC` is discussed. Chapter 7 is dedicated to providing the reader with best practices for the use of MLC.

2.3 Examples

As outlined above, Machine Learning Control (MLC) formulates control design as a regression problem: Find a law which minimizes a given cost function. Genetic Programming (GP) is a regression tool and key enabler of MLC. In the following, we illustrate genetic programming for a simple two-dimensional data fit (Sect. 2.3.1) and and a control problem (Sect. 2.3.2). We ease the replication of the results by employing `OpenMLC`, a freely available Matlab® toolbox designed for MLC (see the Appendix for instructions about how to download and install). `OpenMLC` has been used for most numerical and experimental control problems of this book.

2.3.1 Fitting a Function Through Data Points

In the first example, we search for a one-dimensional function passing as close as possible to given data points.

Problem Formulation

Let us consider the 201 points in the plane:

$$s_i = i/10 \tag{2.5a}$$

$$b_i = \tanh(1.256\, s_i) + 1.2, \quad i = -100 \ldots 100. \tag{2.5b}$$

The goal is to find a function $s \to b = K(s)$ that goes through these points as close as possible using GP. A canonical cost function reads

$$J = \frac{1}{201} \sum_{i=-100}^{100} (b_i - K(s_i))^2 . \tag{2.6}$$

We do not assume a particular structure of the function K, like an affine, constant-linear-quadratic or polynomial structure. This excludes the least-mean-square approach for parameter identification.

In the following, we describe how GP solves the regression problem

$$K(s) = \underset{K'(s)}{\operatorname{argmin}}\, J\left[K'(s)\right], \tag{2.7}$$

i.e. finding a function $b = K(s)$ which minimizes the cost function (2.6).

This example is implemented as the default toy problem for `OpenMLC` which will be used throughout the section.

Code 2.1 Creation of the default regression problem

```
clear all
close all

mlc=MLC % Creates a MLC object with default values that
        % implements the simple regression problem.
```

Problem Solution

We take typical parameters for GP. These are also the default parameters of `OpenMLC` (Table 2.1).

Table 2.1 Parameters for MLC with genetic programming for example of fitting a function through data points

Parameter	Value
N_i	1000
P_r	0.1
P_m	0.2
P_c	0.7
N_p	7
N_e	10
Node functions	$+, -, \times, /$, exp, log, tanh

Our 'plant' is then written as follows:

Code 2.2 Evaluation function for the first regression problem (2.7)

```
function J=toy_problem(ind,parameters,i,fig)

%% Creation of the points to fit.
s=-10:0.1:10;
b=tanh(1.256*s)+1.2;

%% Initialisation of b_hat as a vector.
b_hat=b*0;

%% Evaluation.
% Evaluation is always encapsulated in try/catch.
% Structure to account for the unpredictible.

try
   % Translation from LISP.
idv_foraml=readmylisp_to_formal_MLC(ind,parameters);
idv_formal=strrep(m,'S0','s'); % Replace S0 leaf with
    variable s
   % Obtain the estimated s.
eval(['b_hat=' idv_formal ';'])
   % Obtain the cost function value.
J=sum((b-b_hat).^2)/length(b);
catch err
    % If something goes wrong, asign a bad value.
    J=parameters.badvalue;
    fprintf(err.message);
end
```

The code 2.2 comprises all the necessary steps needed in order to build the problem with corresponding cost function in `OpenMLC`. The evaluation function takes an individual

$$\hat{b} = \hat{K}(s) \tag{2.8}$$

as argument and returns a cost function value J. In addition, GP parameters enter as arguments.

One critical step is the translation of the individual from a LISP expression to something Matlab® can use as a function. This is realized through the function `readmylisp_to_formal_MLC` which recursively interprets each element of the tree. Generically all inputs are numbered $S0$, $S1$ to SN_s. Here, only one input is declared for the function that is to be learned. Thus only functions involving $S0$ are created. For the sake of readability, we replace $S0$ by s, so that the strings created are functions of s. Once the individual is interpreted, it can be used to approximate the data with the mapping (2.8) and compute the corresponding cost function.

MLC Run and Results

Here GP is launched for 50 generations using

```
mlc.go(50);
```

or

```
mlc.go(50,1);
```

to have graphical output. At the end of the 50th generation, typing

```
mlc
```

returns the best individual, its cost function value and other statistics:

```
After generation 50:
Best individual:
(+ (sin (+ (sin (sin (tanh (/ (/ S0 -2.511) (sin -8.741)))))
(sin (tanh (/ (/ S0 -2.629) (sin -8.741)))))) (log (log (+
(tanh (+ (sin (tanh (/ (sin (/ S0 -2.511)) (* -3.802 7.167))))
(tanh (tanh (+ (sin (+ (sin -8.741) (sin -8.741))) (sin 6.774))))))
(* -3.802 7.167)))))

Best J (average over 3 occurrence(s)):    2.380030e-06
```

This implies that the returned function is much more complicated than the actual function and produces an average error of 2.38×10^{-6}. Note that we did not penalize complexity of the individuals.

Typing

```
mlc.show_best_indiv;
```

will additionally display the original and learned relationship between the dataset used for regression (Fig. 2.14).

2.3.2 MLC Applied to Control a Dynamical System

The second study exemplifies MLC for control design of a noise-free, linear, one-dimensional ordinary differential equation, arguably the most simple example of Eq. (1.1).

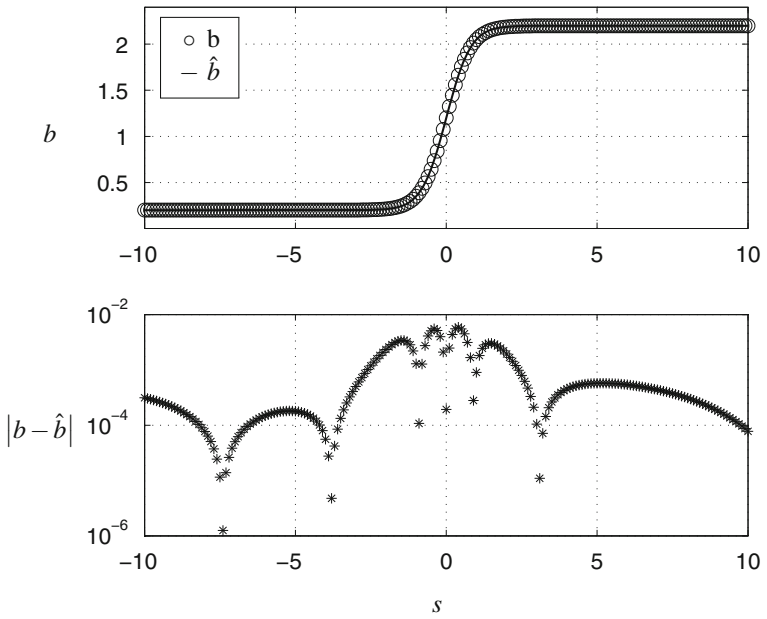


Fig. 2.14 Best individual of the regression problem (2.7) after 50 generations, obtained with '`mlc.show_best_indiv`' on the '`toy_problem`' regression problem

Problem Formulation

We investigate the ordinary differential equation

$$\frac{da}{dt} = a + b\,, \tag{2.9a}$$

$$s = a\,, \tag{2.9b}$$

$$b = K\,(s)\ , \tag{2.9c}$$

with the initial condition

$$a(0) = 1. \tag{2.10}$$

The cost function to be minimized penalizes a deviation from desired state $a_0 = 0$ and actuation,

$$J = \frac{1}{T}\int_0^T \left[a^2 + \gamma b^2\right] dt. \tag{2.11}$$

Here, T is the evaluation time and γ is a penalization coefficient for the cost of actuation.

Table 2.2 Parameters for MLC with genetic programing for simple control design example

Parameter	Value
N_i	50
P_r	0.1
P_m	0.4
P_c	0.5
N_p	7
N_e	1
Node functions	$+, -, \times$, tanh

This problem is set-up in the `OpenMLC` script `unidim_DS_script.m` and the associated object can be instantiated by calling:

```
mlc=MLC('unidim_DS_script');
```

This command implements the new control regression problem

$$K(s) = \underset{K'(s)}{\operatorname{argmin}}\, J\left[K'(s)\right], \tag{2.12}$$

i.e. finding a control law which minimizes the cost function (2.11).

The search space for the control law contains compositions using operations taken from $(+, -, \times, \tanh)$. This includes arbitrary polynomials and sigmoidal functions. A full list of GP parameters used (Table 2.2) can be obtained by typing:

```
mlc.paramters
```

The most important parameters are given in Table 2.2.

Problem Implementation

The evaluation function for this simple dynamical system control problem is provided under the name `unidim_DS_evaluator.m`.

Code 2.3 Evaluation function for control regression problem (2.12)

```
function J=unidim_DS_evaluator(ind,mlc_parameters,i,fig
    )
%% Obtaining parameters from MLC object.
Tf=mlc_parameters.problem_variables.Tf;
objective=mlc_parameters.problem_variables.objective;
gamma=mlc_parameters.problem_variables.gamma;
Tevmax=mlc_parameters.problem_variables.Tevmax;

%% Interpret individual.
m=readmylisp_to_formal_MLC(ind);
m=strrep(m,'S0','y');
K=@(y)(y);
eval(['K=@(y)(' m ');']);
```

```
f=@(t,y)(y+K(y)+testt(toc,Tevmax));

%% Evaluation
try                  % Encapsulation in try/catch.
tic
[T,Y]=ode45(f,[0 Tf],1);   % Integration.
if T(end)==Tf              % Check if Tf is reached.
   b=Y*0+K(Y);            % Computes b.
   Jt=1/Tf*cumtrapz(T,(Y-objective).^2+gamma*b.^2);%
       Computes J.
   J=Jt(end);
else
    J=mlc_parameters.badvalue;  % Return high value if
        Tf is not reached.
end
catch err
   J=mlc_parameters.badvalue    % Return high value if
       ode45 fails.
end

if nargin>3    % If a fourth argument is provided, plot
    the result
  subplot(3,1,1)
  plot(T,Y,'-k','linewidth',1.2)
  ylabel('$a$','interpreter','latex','fontsize',20)
  subplot(3,1,2)
  plot(T,b,'-k','linewidth',1.2)
  ylabel('$b$','interpreter','latex','fontsize',20)
  subplot(3,1,3)
  plot(T,Jt,'-k','linewidth',1.2)
  ylabel('$(a-a_0)^2+\gamma b^2$','interpreter','latex'
      ,'fontsize',20)
  xlabel('$t$','interpreter','latex','fontsize',20)
end
```

First of all we retrieve the parameters which are stored in the structure

`mlc.parameters.problem_variables`

This structure is empty by default and can be used to specify variables that are specific to the problem. Here, the evaluation time T, the penalization coefficient γ, the desired state $a_0 = 0$ and the integration time T_{max} are available from the structure `problem_variables` and retrieved in the first lines of the evaluation function. This allows one to parametrize the problem, so that running the same problem with different values for the parameters can be implemented in a loop.

The individual is interpreted as a control law using the `OpenMLC` toolbox function `readmylisp_to_formal_MLC`. This transforms the LISP string to a Matlab® expression string if copied in the console. By default, sensors are named $S0$, $S1$, ..., Sn_a. As in the first example, we replace $S0$ by the sensor symbol s, with a `strrep` (string replacement) parsing. Finally, a symbolic function is created inside a call to

`eval` that allows one to use the formal expression in the individual string to define the control function.

The dynamical system is then written as a actuated dynamics F defining the derivative of the state a, including the control law, and a `OpenMLC` toolbox function `testt(toc,Tevmax)` which returns an error when the time elapsed for this evaluation is higher than the time specified in `Tevmax`. This works with the placement of the `tic` function at the beginning of the evaluation function.

The integration of the controlled dynamical system is realized in a try/catch structure which allows error management. As `OpenMLC` will allow any combination of the elementary functions to be tested, it is likely that many systems will diverge or cause integration errors. The integration of the controlled dynamics becomes expensive with complex control laws. Here, a generation contains 1000 individuals to test. Hence, a `testt` function is provided for numerical problems, so that an error is automatically generated when `Tevmax` seconds have passed in the real world. When an error is generated, the program continues in the catch section, where a high value specified in `parameters.badvalue` is attributed to the individual and the evaluation is stopped.

If no error is generated, the J value is computed according to Eq. (2.11) and is stored in the variable J, which will be sent back to the `OpenMLC` object.

This could be the end of the evaluation function, but `OpenMLC` allows the use of an optional fourth argument to force a graphic output. This can be used to observe one specific individual. For instance,

```
unidim_DS_evaluator(mlc.population(4).individuals{5},
    mlc.parameters,1,1)
```

will display the fourth individual of the fifth generation. If the problem is launched using:

```
mlc.go(15,1)
```

A graphical output will be generated for the best individual of each generation. The best individual is represented according to the section included in the `if nargin>4` structure (if the **n**umber of **arg**uments **in**puted is strictly larger than 3). Here, the state, the control and the cost function integration are plotted against time.

MLC Run and Results

We choose 15 generations to solve the control problem. MLC is launched using

```
mlc.go(15);
```

Typing:

```
mlc
```

returns the best individual and statistics such as its cost function value:

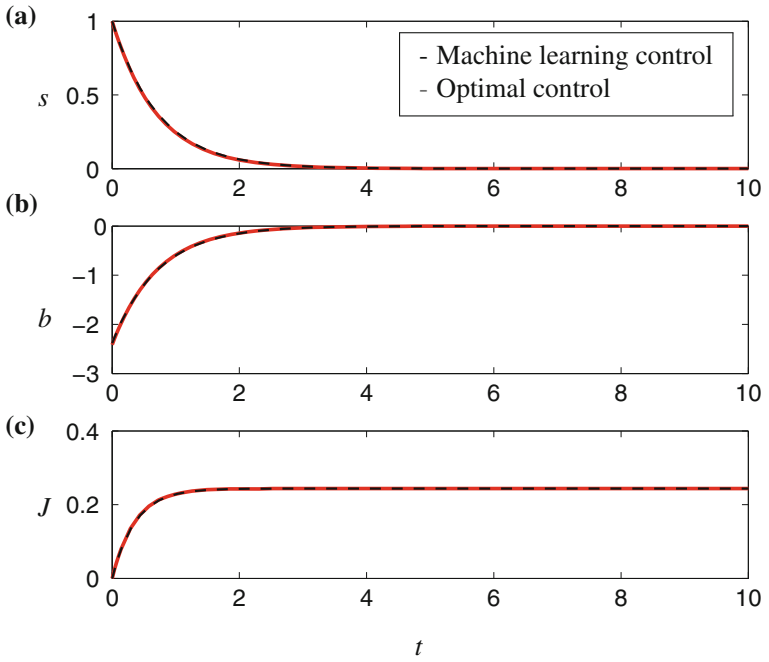


Fig. 2.15 Best individual example after 15 generations, obtained with 'mlc._show_best_indiv' on the 'unidim_DS_evaluator' control problem. The *red continuous line* shows the optimal control (see Chap. 3) for this control problem

```
After generation 15:
Best individual:
(* (* 3.128 S0) (tanh (tanh -6.613)))
Best J (average over 7 occurrence(s)):    2.437028e-01
```

Once again, typing:

```
mlc.show_best_indiv
```

will provide the graphs specified in the evaluation function for the best individual (Fig. 2.15). The time-dependent cost function

$$J(t) = \frac{1}{T}\int_0^t \left[a^2(\tau) + \gamma\, b^2(\tau)\right] d\tau \tag{2.13}$$

quantifies the contribution of its integrand during time $[0, t]$ to Eq. (2.11). Note that $J(t)$ must be monotonically increasing from 0 to $J(T) = J$, as the normalization with $1/T$ is fixed and Eq. (2.13) does not define a sliding-window average.

`OpenMLC` also contains methods to analyze the whole evolution process. For instance,

```
mlc.show_convergence
```

displays a succession of histograms of the cost function values for each generation (see Fig. 2.16). For the default values (Fig. 2.16a, b), 1000 bins are created with a logarithmic progression from the minimal to the maximal value of J within all generations (excluding values equal or above `mlc.parameters.badvalue`). Section 7.4.1 provides a detailed description. The colormap reflects the ordering of individuals by cost function, not by the quantitative values. Thus, the 2D-view (Fig. 2.16a, c) illustrates the J values of the individuals while a 3D-view (Fig. 2.16b, d) reveals also the population density associated with J-values. The detailed map is obtained by the command:

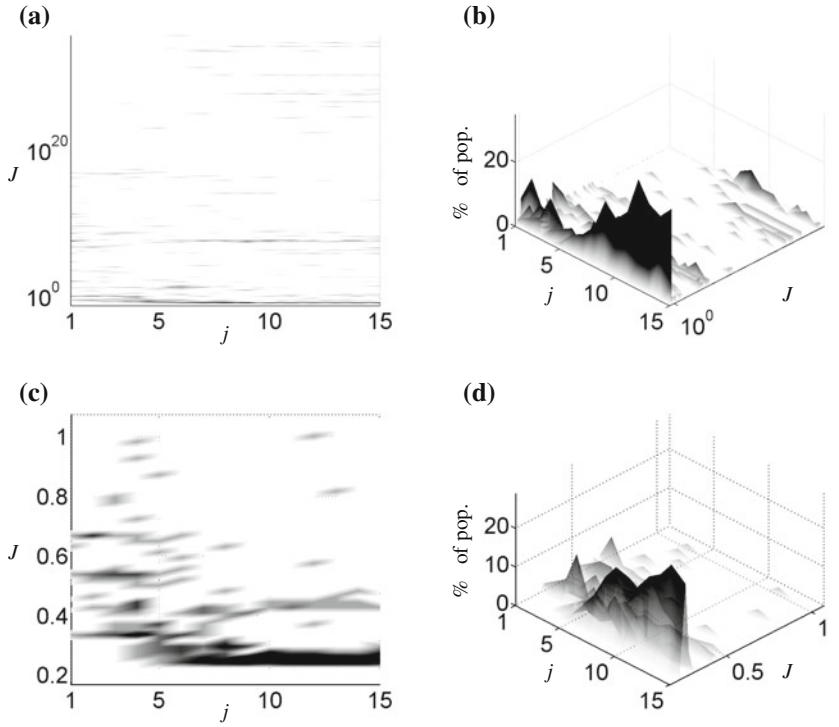


Fig. 2.16 Successive histograms of the population repartition in the cost-function value space. (**a**, **c**) (detail): *Top view*. Well performing individuals were obtained in the first generation and no significant progress has been achieved after the 4th generation. (**b**, **d**) (detail): *3D view*. Progressively the population sees a large proportion around the best individuals (**d**) while it can be observed that some diversity is kept (**b**). Generated using `mlc.show_convergence`, parameters in text

```
mlc.show_convergence(50,0,1,1,3)
```

Most evolutive processes of dynamical systems will feature similar behaviors and properties:

- The progression is achieved by successive jumps. Once a better individual is found, it is used many times for the evolution and rapidly an island around its J-value is formed. Then, a better individual is found and the population gradually shifts toward the better one.
- Some diversity is kept and some individuals and/or other islands are found in the graph for every generation.
- There is one streak which is present for all generations, though it is far from optimal: these are the many incarnations of the zero-individual. The zero can be created in many ways: subtraction of the same subtree, multiplication by zero, etc. Each generation can be expected to create such zero-individuals.
- Potentially other far-from-optimal islands will be present for other generations: it can be saturated individuals if you impose, for instance, too narrow bounds on the actuation command, or other problem specific phenomena.

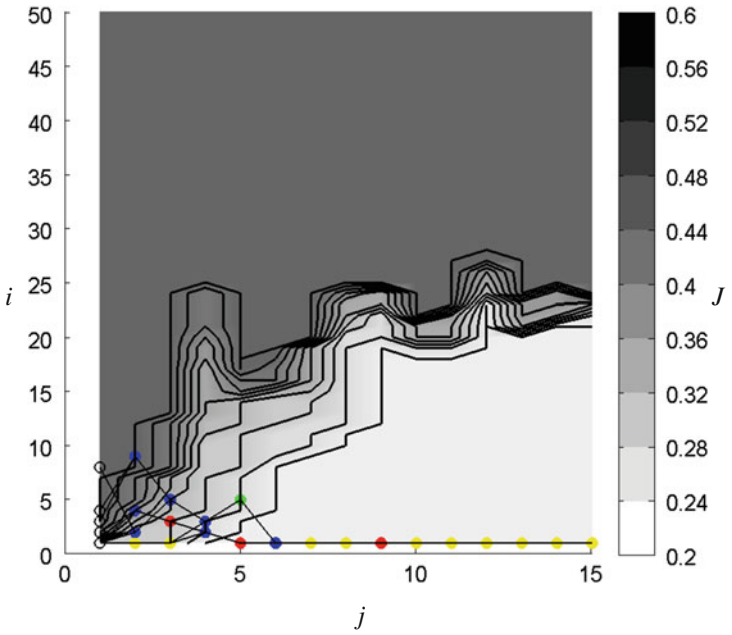


Fig. 2.17 Genealogy of the best individual. All individuals are ranked by performance. The best individual is linked to its parent(s). Color indicates the process: *yellow* is elitism, *red* is mutation, *green* is replication and *blue* is crossover. The background colormap indicates the J-values for the individuals of index i in generation j. It appears that the best individual after 15 generations has been found from the 10th generation, and that the global population is not evolving any more

Another way to check the convergence process is by typing:

```
mlc.genealogy(15,1)
```

which shows the ancestors of the best individual of the 15th generation (Fig. 2.17). Individuals are linked by parenthood and colors show the genetic operation that resulted in the next individual: yellow for elitism, red for mutation, blue for crossover and green for replication. More importantly, the background shows which proportion of the population is in a given order of magnitude of the cost function.

`mlc.show_convergence` and `mlc.genealogy` both are customizable. Full options can be found by typing:

```
help MLC/show_convergence
help MLC/genealogy
```

2.4 Exercises

Exercise 2–1: Transform the example from Sect. 2.3.1 in order to achieve a surface regression:

$$\begin{aligned} r_i &= i/10 \\ s_j &= j/10 \\ b_{i,j} &= \tanh(1.256\, r_i\, s_i) + 1.2\sin(s_i), \quad i, j \in \{-100 \ldots 100\}. \end{aligned}$$

Exercise 2–2: Transform the example from Sect. 2.3.2 in order to learn $b = K(a)$ so that the following dynamical system is stabilized to a fixed point:

$$\begin{aligned} \frac{da}{dt} &= a\left[\frac{a}{10} - \frac{a^2}{10000}\right] + b \\ b &= K(a). \end{aligned}$$

Exercise 2–3: Stabilize the following Lorenz system to each of its three fixed points using MLC:

$$\begin{aligned} \frac{da_1}{dt} &= \sigma\,(a_2 - a_1) \\ \frac{da_2}{dt} &= a_1\,(\rho - a_3) - a_2 \\ \frac{da_3}{dt} &= a_1 a_2 - \beta a_3 + b \\ b &= K(a_1, a_2, a_3), \end{aligned}$$

with $\sigma = 10$, $\beta = 8/3$ and $\rho = 28$.

(a) Determine the three fixedpoints.
(b) Write the three cost function associated to the stabilization of each of the fixed points.
(c) Run MLC for each of the cases.

2.5 Suggested Reading

Texts

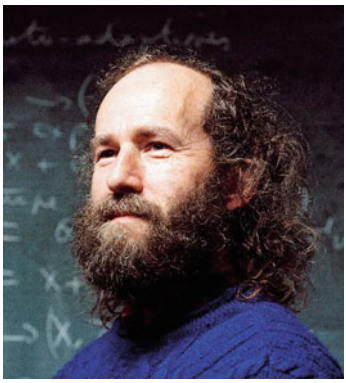
(1) **Learning from Data**, by Y. S. Abu-Mostafa, M. Magndon-Ismail, H.-T. Lin, 2012 [2].
This book is an exquisite introduction into the principles of learning from data and is highly recommended as a first reading.
(2) **Pattern Classification**, by R. O. Duda, P. E. Hart, and D. G. Stork, 2000 [92].
This classic text provides a serious introduction to machine learning and classification from the probabilistic perspective.
(3) **Pattern Recognition and Machine Learning**, by C. Bishop, 2006 [30].
This text provides a complete overview of machine learning with a self-contained prior on probability theory. Bayes' theory is highlighted in this text.
(4) **Machine Learning: a Probabilistic Perspective**, by K. P. Murphy, 2012 [194].
This comprehensive text describes automated methods in machine learning that may be applied to increasingly big data. There are numerous examples using methods that are computationally available.
(5) **Genetic Programming: On the Programming of Computers by Means of Natural Selection**, by J. R. Koza, 1992 [164].
This seminal text provides a complete overview of the theory of genetic programming. This material has since become the gold standard evolutionary algorithm.
(6) **Genetic Programming: An Introduction**, by W. Banzhaf, P. Nordin, R. E. Keller, and R. D. Francone, 1998 [17].
This text provides an excellent introduction to genetic programming and its applications.

Seminal Papers

(1) **Top 10 algorithms in data mining**, by X. Wu *et al.*, *Knowledge and Information Systems*, 2008 [280].
This paper provides an excellent description of ten of the most ubiquitous and powerful techniques in machine learning and data mining that are in use today.
(2) **A tutorial on support vector regression**, by A. J. Smola and B. Schölkopf, *Statistics and Computing*, 2004 [253].
This paper describes the support vector regression, which has become one of the highest performing classifiers in machine learning.

(3) **Random forests**, by L. Breiman, *Machine learning*, 2001 [33].
This paper describes the natural generalization of decision trees: random forests. In this framework, an ensemble of decision trees is used to improve classification performance.

2.6 Interview with Professor Marc Schoenauer



Professor Marc Schoenauer is Principal Senior Researcher (Directeur de Recherche 1ére classe) at INRIA, the French National Institute for Research in Computer Science and Control. He graduated at Ecole Normale Supérieure in Paris, and obtained a PhD in Numerical Analysis at Université Paris 6 in 1980. From 1980 until Aug. 2001 he has been a full time researcher with CNRS (the French National Research Center), working at CMAP (the Applied Maths Laboratory) at Ecole Polytechnique. He then joined INRIA, and later founded the TAO team at INRIA Saclay in September 2003 together with Michèle Sebag. Marc Schoenauer has been working in the field of Evolutionary Computation (EC) since the early 90s, more particularly at the interface between EC and Machine Learning (ML). He is author of more than 130 papers in journals and major conferences of these fields. He is or has been advisor to 30 PhD students. He has also been part-time Associate Professor at Ecole Polytechnique in the Applied Maths Department from 1990 to 2004.

Marc Schoenauer is chair of the Executive Board of SIGEVO, the ACM Special Interest Group for Evolutionary Computation. He was Senior Fellow and member of the Board of the late ISGEC (International Society of Genetic and Evolutionary Computation), that has become ACM-SIGEVO in 2005. He has served in the IEEE Technical Committee on Evolutionary Computation from 1995 to 1999, and is a member of the PPSN Steering Committee. He was the founding president (1995–2002) of Evolution Artificielle, the French Society for Evolutionary Computation, and has been president of the French Association for Artificial Intelligence (2002–2004). Marc Schoenauer has been Editor in Chief of Evolutionary Computation Journal (2002–2009), is or has been Associate Editor of IEEE Transactions on Evolutionary Computation (1996–2004), Theoretical Computer Science—Theory of

Natural Computing (TCS-C) (2001–2006), Genetic Programming and Evolvable Machines Journal (1999–now), and the Journal of Applied Soft Computing (2000–now), and is Acting Editor of Journal of Machine Learning Research (JMLR) since 2013. He serves or has served on the Program Committees of many major conferences in the fields of Evolutionary Computation and Machine Learning.

Authors: Dear Marc, you have pushed the frontiers of evolutionary algorithms by numerous enablers. You proposed one of the first mechanical engineering applications of genetic programming, namely identifying the behavioral law of materials. Many readers of our book are from fluid mechanics with little background in evolutionary algorithms. Could you describe the need for evolutionary algorithms and the particular role of genetic programming? How did you get attracted to genetic programming shortly after Koza's discovery in 1992?

Prof. Schoenauer: First of all, let me state that the work you refer to dates back 20 years now. Furthermore, it was a team work, in collaboration with Michèle Sebag on the algorithmic side, François Jouve on the numerical side, and Habibou Maitournam on the mechanical side.

To come back to your question, the Genetic Programming motto—write the program that writes the program—would be appealing to any programmer who wouldn't call it crazy. I got interested in Evolutionary Computation after hearing a talk by Hugo de Garis in 1992 [77], who evolved a controller for walking humanoid stick-legs. I realized the potentialities of evolution as a model for optimization for problems which more classical methods could not address. Remember that I was trained as an applied mathematician.

The identification of behavioral laws, like many inverse problems, was one such problem. However, it should be made clear that applied mathematicians also make continuous progresses, solving more and more problems of that kind—though also unveiling new applications domains for evolutionary methods.

Authors: How would you compare genetic programming with other regression techniques for estimation, prediction and control, e.g. linear or linear-quadratic regression, neural networks, genetic algorithms, Monte-Carlo methods and the like.

Prof. Schoenauer: This is a tricky question, especially in these days of Deep Neural Networks triumphs in image and video processing, in games, etc.

One usual argument for genetic programming compared to other regression techniques is the understandability of the results. It is true that there are several examples of such understandable results, starting with Koza's work on digital circuit design [165] (and including our findings in behavioral laws). However, many other applications in genetic programming result in rather large trees, and their interpretation is more problematic. Some interesting work by Hod Lipson [180] and Tonda et al. [110] demonstrate that it should be possible to preserve this advantage. Nevertheless, in most cases, the user has to choose how to handle the trade-off between precision and concision—and if precision is preferred, then genetic programming might not be the best choice (keeping in mind that as of today, all such claims are problem-dependent).

Authors: Which advice would you give a fluid dynamicist for shortening the learning time with machine learning control? Which enablers and show-stoppers do you see?

Prof. Schoenauer: My first advice would be to start with what they know (from fluid mechanics and applied maths), and clearly state the limits of the standard methods. From there on, Machine Learning or Evolutionary Computation might be the solution they are looking for. Then they should consider different approaches, at least genetic programming and neural network-based approaches (note that there are very nice results too that hybridize both, as mentioned). And overall, if choosing such non-traditional approach, they should consider all simplifying hypotheses that they have made when trying to solve the problem using standard methods, as these hypotheses might not be necessary any more when using ML/EC methods, opening wide news areas of alternative solutions. In any case, they should not look for a nail for their genetic programming hammer.

Authors: We thank you for this interview!

Chapter 3
Methods of Linear Control Theory

Guaranteed Margins for LQG Regulators. Abstract—There are none.
John Doyle IEEE Transactions on Automatic Control, 1978 [86]

The most well-developed theory of control generally applies to a linear system or to the linearization of a nonlinear system about a fixed point or a periodic orbit. Linear control theory has many applications in fluid dynamics, such as the stabilization of unstable laminar boundary layers. Although the governing equations may be nonlinear, successful stabilizing controllers will regulate the system to a neighborhood where the linearization is increasingly valid.

In this chapter we introduce linear systems (Sect. 3.1) and explore $\mathscr{H}_2$ optimal control problems, including the linear quadratic regulator (LQR) in Sect. 3.2 and the Kalman filter in Sect. 3.3. These problems are chosen because of their simplicity, ubiquitous application, well-defined quadratic cost-functions, and the existence of known optimal solutions. Next, linear quadratic Gaussian (LQG) control is introduced for sensor-based feedback in Sect. 3.4. Finally, methods of linear system identification are provided in Sect. 3.5.

This chapter is not meant to be an exhaustive primer on linear control theory, although key concepts from optimal control are introduced as needed to build intuition. Note that none of the linear system theory below is required to implement the machine learning control strategies in the remainder of the book, but they are instead included to provide context and demonstrate known optimal solutions to linear control problems. In many situations, $\mathscr{H}_\infty$ robust control may be more desirable to balance the trade-off between robustness and performance in systems with uncertainty and unmodeled dynamics, and the MLC methods developed here may be generalized to other cost functions. For a more complete discussion of linear control theory, excellent books include [93, 252].

T. Duriez et al., *Machine Learning Control – Taming Nonlinear Dynamics and Turbulence*, Fluid Mechanics and Its Applications 116,
DOI 10.1007/978-3-319-40624-4_3

3.1 Linear Systems

Many systems of interest are either linear, or correspond to the linearization of a nonlinear system, such as Eq. (1.1), about a fixed point or periodic orbit. The most complete theory of control applies to linear systems. Consider the following state-space system:

$$\frac{d}{dt}\mathbf{a} = \mathbf{A}\mathbf{a} + \mathbf{B}\mathbf{b} \tag{3.1a}$$

$$\mathbf{s} = \mathbf{C}\mathbf{a} + \mathbf{D}\mathbf{b}. \tag{3.1b}$$

The matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, and $\mathbf{D}$ arise from the linearization of Eq. (1.1) about an equilibrium state $\mathbf{a}_0$; in Eq. (3.1), the state $\mathbf{a}$ is the deviation from the equilibrium $\mathbf{a}_0$. In the absence of an actuation input $\mathbf{b}$, the solution to Eq. (3.1a) is:

$$\mathbf{a}(t) = e^{\mathbf{A}(t-t_0)}\mathbf{a}(t_0), \tag{3.2}$$

where the matrix exponential $e^{\mathbf{A}t}$ is given by the infinite series:

$$e^{\mathbf{A}t} = \mathbf{I} + \mathbf{A}t + \frac{1}{2!}\mathbf{A}^2t^2 + \frac{1}{3!}\mathbf{A}^3t^3 + \cdots. \tag{3.3}$$

The stability of this system is determined by the eigenvalues of $\mathbf{A}$, and eigenvalues with positive real part are unstable. The corresponding eigenvectors represent unstable state directions, where perturbations will either grow without bound, or grow until unmodeled nonlinear dynamics become important.

In the case of an actuation input $\mathbf{b}$, the solution to Eq. (3.1a) becomes:

$$\mathbf{a}(t) = e^{\mathbf{A}t}\mathbf{a}(0) + \int_0^t e^{\mathbf{A}(t-\tau)}\mathbf{B}\mathbf{b}(\tau)\,d\tau. \tag{3.4}$$

The system in Eq. (3.1a) is *controllable* if it is possible to navigate the system to an arbitrary state $\mathbf{a}$ from the origin in finite time with a finite actuation signal $\mathbf{b}(t)$. Mathematically, this relies on the controllability matrix

$$\mathscr{C} = \left[\mathbf{B}\ \mathbf{A}\mathbf{B}\ \mathbf{A}^2\mathbf{B}\ \cdots\ \mathbf{A}^{N_a-1}\mathbf{B}\right] \tag{3.5}$$

having full column rank. In practice, the *degree of controllability*, characterized by the singular value decomposition of the controllability matrix in Eq. (3.5), or equivalently, by the eigen-decomposition of the controllability Gramian, is often more useful. Note that if Eq. (3.1a) is the linearization of a nonlinear system about a fixed point, then it may be controllable with a nonlinear controller $\mathbf{b} = \mathbf{K}(\mathbf{a})$, even if Eq. (3.1a) is linearly uncontrollable. As long as all unstable state directions are in the span of $\mathscr{C}$, then the system is *stabilizable*; these unstable directions correspond to eigenvectors of $\mathbf{A}$ with eigenvalues having positive real part.

Similarly, the system in Eq. (3.1b) is *observable* if any state $\mathbf{a}$ may be estimated from a time-history of sensor measurements $\mathbf{s}$. Mathematically, this corresponds to the observability matrix

$$\mathscr{O} = \begin{bmatrix} \mathbf{C} \\ \mathbf{CA} \\ \mathbf{CA}^2 \\ \vdots \\ \mathbf{CA}^{N_a-1} \end{bmatrix} \tag{3.6}$$

having full row rank. A system is *detectable* if all unstable states are *observable*, so that they are in the row-space of $\mathscr{O}$.

As in Eq. (1.1), the matrices in the linearization in Eq. (3.1) may depend on the specific bifurcation parameters $\boldsymbol{\mu}$. The linear theory above may also be generalized for linear parameter varying (LPV) systems, where the matrices in Eq. (3.1) depend on a time-varying parameter [16, 247]. For example, when linearizing about a periodic orbit, the matrices are parameterized by the phase ϕ of the trajectory on the orbit. In this case, gain-scheduling allows different controllers to be applied depending on the parameter values [233, 247].

3.2 Full-State Feedback

If measurements of the full state $\mathbf{a}$ are available, then $\mathbf{D} = \mathbf{0}$ and $\mathbf{C} = \mathbf{I}$, where $\mathbf{I}$ is the $N_a \times N_a$ identity matrix. We may then consider full-state feedback control $\mathbf{b} = \mathbf{K}(\mathbf{a})$ based on measurements of the state, $\mathbf{s} = \mathbf{a}$. Although full-state feedback may be unrealistic, especially for high-dimensional systems, it is often possible to estimate the full state from limited sensor measurements, using a Kalman filter, as discussed in Sect. 3.3. Remarkably, it is possible to design an optimal full-state feedback controller and an optimal state-estimator separately, and the combined sensor-based feedback controller will also be optimal, as we will show in Sect. 3.4.

Linear Quadratic Regulator (LQR)

If the system in Eq. (3.1a) is controllable, then it is possible to design a proportional controller

$$\mathbf{b} = -\mathbf{K}_r \mathbf{a} \tag{3.7}$$

to arbitrarily place the eigenvalues of the closed-loop system

$$\frac{d}{dt}\mathbf{a} = \mathbf{A}\mathbf{a} + \mathbf{B}\mathbf{b} = (\mathbf{A} - \mathbf{B}\mathbf{K}_r)\,\mathbf{a}. \tag{3.8}$$

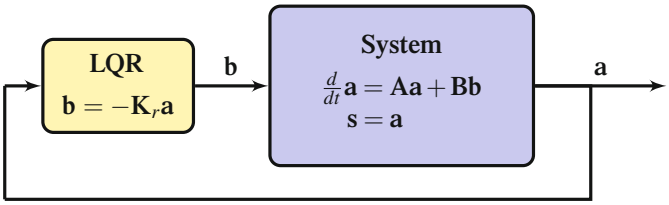


Fig. 3.1 LQR stabilization problem. The optimal control for a linear system with full-state feedback $\mathbf{s} = \mathbf{a}$ is given by proportional control $\mathbf{b} = -\mathbf{K}_r\mathbf{a}$ where $\mathbf{K}_r$ is a gain matrix obtained by solving an algebraic Riccati equation

A natural goal in control theory is to stabilize the system so that the state $\mathbf{a}$ converges quickly to $\mathbf{0}$, but without expending too much control effort. We may construct a quadratic cost function J that balances the aggressive regulation of $\mathbf{a}$ with the cost of control:

$$J(t) = \int_0^t \left[\mathbf{a}^T(\tau)\mathbf{Q}\,\mathbf{a}(\tau) + \mathbf{b}^T(\tau)\mathbf{R}\mathbf{b}(\tau)\right] d\tau. \tag{3.9}$$

The goal is to develop a control strategy $\mathbf{b} = -\mathbf{K}_r\mathbf{a}$ to minimize $J = \lim_{t\to\infty} J(t)$. The matrices $\mathbf{Q}$ and $\mathbf{R}$ weight the cost of deviations of the state from zero and the cost of actuation, respectively. $\mathbf{R}$ is positive definite and $\mathbf{Q}$ is positive semi-definite. These matrices are often diagonal, and the diagonal elements may be tuned to change the relative importance of the control objectives. For example, if we increase the entries of $\mathbf{Q}$ by a factor of 10 and keep $\mathbf{R}$ the same, then accurate state regulation is more heavily weighted, and more aggressive control may be permitted. Typically, the ratios of elements in $\mathbf{Q}$ and $\mathbf{R}$ are increased or decreased by powers of 10.

Because of the well-defined quadratic cost function in Eq. (3.9), the optimal controller $\mathbf{K}_r$ may be solved for analytically. In particular, the controller $\mathbf{K}$ that minimizes the cost in Eq. (3.9) is given by

$$\mathbf{K}_r = \mathbf{R}^{-1}\mathbf{B}^T\mathbf{X}, \tag{3.10}$$

where $\mathbf{X}$ is the solution to the algebraic Riccati equation:

$$\mathbf{A}^T\mathbf{X} + \mathbf{X}\mathbf{A} - \mathbf{X}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{X} + \mathbf{Q} = \mathbf{0}. \tag{3.11}$$

The resulting full-state feedback controller is called a linear quadratic regulator (LQR), since it is a linear control law that minimizes a quadratic cost function to regulate the system. This is shown schematically in Fig. 3.1. Solving for the LQR controller $\mathbf{K}_r$ in Eq. (3.10) is computationally robust, and it is a built-in routine in many computational packages. However, the computational cost of solving the Riccati equation in Eq. (3.11) scales with the cube of the state dimension, making it prohibitively expensive for large systems, except as an off-line calculation.

3.3 Sensor-Based State Estimation

The optimal LQR controller above relies on access to the full state of the system. However, in many applications full-state measurements of a high-dimensional system are either technologically infeasible or prohibitively expensive to collect and process. When full measurements are available, as with the use of particle image velocimetry (PIV) [127, 278] to measure fluid velocity fields in experiments, these measurements are typically only available in controlled experimental settings, and are not practical for in-field applications such as monitoring flow over a wing in flight. The computational burden of collecting, transferring and processing full-state measurements may also limit the temporal resolution of the measurements and introduce unacceptable time-delays which degrade robust performance.

In practice, it is often necessary to *estimate* the full state $\mathbf{a}$ from limited noisy sensor measurements $\mathbf{s}$. This process balances information from a model prediction of the state with the sensor measurements. Under a set of well-defined conditions it is possible to obtain a stable estimator that converges to an estimate of the full state $\mathbf{a}$, which can then be used in conjunction with the optimal full-state feedback LQR control law described above.

Kalman Filtering

The Kalman filter [156] is perhaps the most often applied algorithm to estimate the full-state of a system from noisy sensor measurements and an uncertain model of the system. Kalman filters have been used in myriad applications, including guidance and tracking of vehicles, airplane autopilots, modeling climate and weather, seismology, and satellite navigation, to name only a few. An excellent and complete derivation of the Kalman filter may be found in [255].

In the dynamic state-estimation framework, the linear dynamics from Eq. (3.1) are generalized to include stochastic disturbances $\mathbf{w}_d$, also known as *process noise*, and sensor noise $\mathbf{w}_n$:

$$\frac{d}{dt}\mathbf{a} = \mathbf{A}\mathbf{a} + \mathbf{B}\mathbf{b} + \mathbf{w}_d \tag{3.12a}$$

$$\mathbf{s} = \mathbf{C}\mathbf{a} + \mathbf{D}\mathbf{b} + \mathbf{w}_n. \tag{3.12b}$$

Both the disturbance and noise terms are assumed to be zero-mean Gaussian white-noise processes, although generalizations exist to handle correlated and biased noise terms. We assume that the disturbance and noise covariances are known:

$$\mathbb{E}\left(\mathbf{w}_d(t)\mathbf{w}_d(\tau)^T\right) = \mathbf{V}_d\delta(t-\tau) \tag{3.13a}$$

$$\mathbb{E}\left(\mathbf{w}_n(t)\mathbf{w}_n(\tau)^T\right) = \mathbf{V}_n\delta(t-\tau) \tag{3.13b}$$

where $\mathbb{E}$ is the expectation operator, and $\delta(\cdot)$ is the Dirac delta function. The matrices $\mathbf{V}_d$ and $\mathbf{V}_n$ are diagonal matrices whose entries contain the variances of the corresponding disturbance or noise term.

A full-state estimator is a dynamical system that produces an estimate $\hat{\mathbf{a}}$ for the full-state $\mathbf{a}$ using only knowledge of the noisy sensor measurements $\mathbf{s}$, the actuation input $\mathbf{b}$, and a model of the process dynamics. If the system is observable, it is possible to construct an estimator with a filter gain $\mathbf{K}_f$ as follows:

$$\frac{d}{dt}\hat{\mathbf{a}} = \mathbf{A}\hat{\mathbf{a}} + \mathbf{B}\mathbf{b} + \mathbf{K}_f\left(\mathbf{s} - \hat{\mathbf{s}}\right) \tag{3.14a}$$

$$\hat{\mathbf{s}} = \mathbf{C}\hat{\mathbf{a}} + \mathbf{D}\mathbf{b}. \tag{3.14b}$$

The output $\hat{\mathbf{s}}$ is a prediction of the expected sensor output based on the full-state estimate $\hat{\mathbf{a}}$. Substituting the expression for $\hat{\mathbf{s}}$ from Eq. (3.14b) into Eq. (3.14a) yields a dynamical system for $\hat{\mathbf{a}}$ with $\mathbf{b}$ and $\mathbf{s}$ as inputs:

$$\frac{d}{dt}\hat{\mathbf{a}} = \left(\mathbf{A} - \mathbf{K}_f\mathbf{C}\right)\hat{\mathbf{a}} + \mathbf{K}_f\mathbf{s} + \left(\mathbf{B} - \mathbf{K}_f\mathbf{D}\right)\mathbf{b} \tag{3.15a}$$

$$= \left(\mathbf{A} - \mathbf{K}_f\mathbf{C}\right)\hat{\mathbf{a}} + \left[\mathbf{K}_f,\ \left(\mathbf{B} - \mathbf{K}_f\mathbf{D}\right)\right]\begin{bmatrix}\mathbf{s}\\ \mathbf{b}\end{bmatrix}. \tag{3.15b}$$

This is shown schematically in Fig. 3.2 for $\mathbf{D} = \mathbf{0}$.

For observable systems in Eq. (3.1), it is possible to arbitrarily place the eigenvalues of the estimator dynamics $\mathbf{A} - \mathbf{K}_f\mathbf{C}$, resulting in stable convergence of the estimate $\hat{\mathbf{a}}$ to the true state $\mathbf{a}$. To see that stable dynamics $\mathbf{A} - \mathbf{K}_f\mathbf{C}$ result in a stable estimator that converges to the full-state $\mathbf{a}$, consider the time dynamics of the estimation error $\boldsymbol{\varepsilon} = \mathbf{a} - \hat{\mathbf{a}}$:

$$\begin{aligned}
\frac{d}{dt}\boldsymbol{\varepsilon} &= \frac{d}{dt}\mathbf{a} - \frac{d}{dt}\hat{\mathbf{a}} \\
&= [\mathbf{A}\mathbf{a} + \mathbf{B}\mathbf{b} + \mathbf{w}_d] - \left[\left(\mathbf{A} - \mathbf{K}_f\mathbf{C}\right)\hat{\mathbf{a}} + \mathbf{K}_f\mathbf{s} + \left(\mathbf{B} - \mathbf{K}_f\mathbf{D}\right)\mathbf{b}\right] \\
&= \mathbf{A}\boldsymbol{\varepsilon} + \mathbf{w}_d + \mathbf{K}_f\mathbf{C}\hat{\mathbf{a}} - \mathbf{K}_f\mathbf{s} + \mathbf{K}_f\mathbf{D}\mathbf{b} \\
&= \mathbf{A}\boldsymbol{\varepsilon} + \mathbf{w}_d + \mathbf{K}_f\mathbf{C}\hat{\mathbf{a}} - \mathbf{K}_f\underbrace{[\mathbf{C}\mathbf{a} + \mathbf{D}\mathbf{b} + \mathbf{w}_n]}_{\mathbf{s}} + \mathbf{K}_f\mathbf{D}\mathbf{b} \\
&= \left(\mathbf{A} - \mathbf{K}_f\mathbf{C}\right)\boldsymbol{\varepsilon} + \mathbf{w}_d - \mathbf{K}_f\mathbf{w}_n.
\end{aligned}$$

Therefore, the estimate $\hat{\mathbf{a}}$ will converge to the true state $\mathbf{a}$ as long as $\mathbf{A} - \mathbf{K}_f\mathbf{C}$ is stable. Analogous to the case of LQR, there is a balance between over-stabilization and the amplification of noise. An analogy is often made with an inexperienced automobile driver who holds the wheel too tightly and reacts to every bump and disturbance on the road.

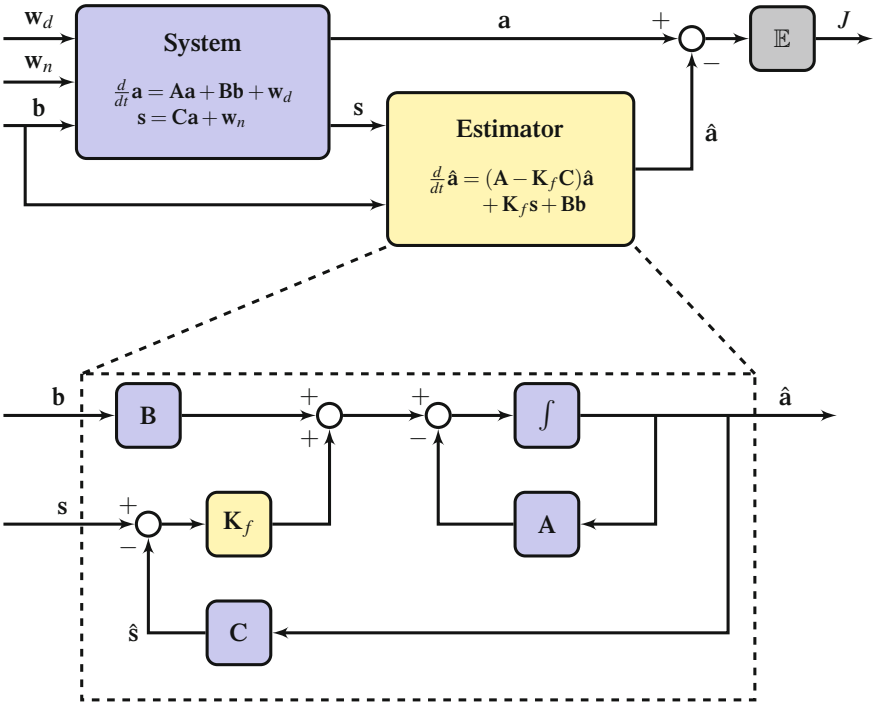


Fig. 3.2 Schematic of the Kalman filter for state estimation from noisy measurements $\mathbf{s} = \mathbf{C}\mathbf{a} + \mathbf{w}_n$ with process noise (disturbance) $\mathbf{w}_d$. Note that there is no feedthrough term $\mathbf{D}$ in this diagram

The Kalman filter is an *optimal* full-state estimator that minimizes the following cost function:

$$J = \lim_{t \to \infty} \mathbb{E}\left((\mathbf{a}(t) - \hat{\mathbf{a}}(t))^T (\mathbf{a}(t) - \hat{\mathbf{a}}(t)) \right). \tag{3.16}$$

Implicit in this cost function are the noise and disturbance covariances, which determine the optimal balance between aggressive estimation and noise attenuation. The mathematical derivation of an optimal solution is nearly identical to that of LQR, and this problem is often called linear quadratic estimation (LQE) because of the dual formulation. The optimal Kalman filter gain $\mathbf{K}_f$ is given by

$$\mathbf{K}_f = \mathbf{Y}\mathbf{C}^T\mathbf{V}_n \tag{3.17}$$

where $\mathbf{Y}$ is the solution to another algebraic Riccati equation:

$$\mathbf{Y}\mathbf{A}^T + \mathbf{A}\mathbf{Y} - \mathbf{Y}\mathbf{C}^T\mathbf{V}_n^{-1}\mathbf{C}\mathbf{Y} + \mathbf{V}_d = \mathbf{0}. \tag{3.18}$$

3.4 Sensor-Based Feedback

In practice, the full-state estimate from sensor-based estimation is used in conjunction with a full-state feedback control law, resulting in sensor-based feedback. The separation principle in control theory states that for linear systems it is possible to design optimal full-state feedback and full-state estimator gain matrices separately, and the resulting sensor-based feedback will remain optimal when combined. There are numerous techniques to develop sensor-based control that optimize different quantities. For instance, combining the LQR and Kalman filter solutions results in what are known as $\mathscr{H}_2$ optimal control laws, while other controllers, known as $\mathscr{H}_\infty$ controllers, may be designed to provide robustness.

In the case of model-free machine learning control, the controller dynamical system, which estimates relevant states from sensors and feeds this state estimate back into an actuation signal, must be designed and optimized as a single unit. Realistically, we may not have access to full-state data to train an estimator, even during an expensive off-line optimization. Moreover, the system under investigation may be unstable, so that a sensor-based controller must first be applied before training an estimator is even feasible. However, it will be possible to design sensor-based feedback controllers in one shot with MLC using generalized transfer function blocks, as will be explored in the following chapter.

Linear Quadratic Gaussian (LQG)

The linear quadratic Gaussian (LQG) controller is the optimal sensor-based feedback control law that minimizes the cost function in Eq. (3.9) using sensors $\mathbf{s}$ from the linear model in Eq. (3.12) with sensor and process noise. Remarkably, the optimal LQG solution is obtained by combining the optimal LQR feedback gain $\mathbf{K}_r$

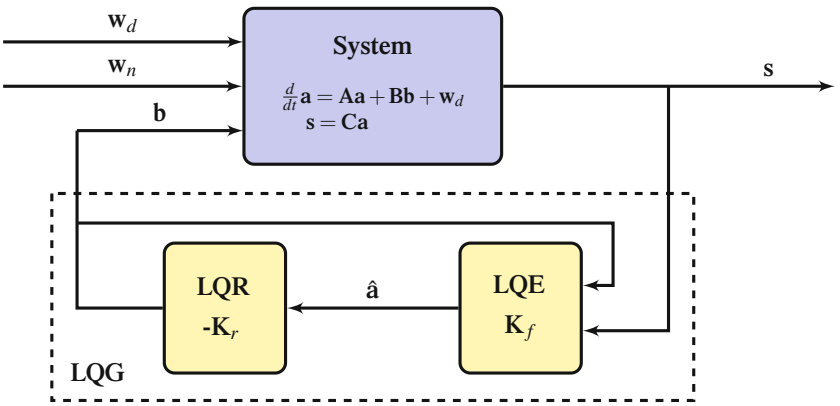


Fig. 3.3 Schematic diagram for linear quadratic Gaussian (LQG) controller. The optimal LQR and LQE gain matrices $\mathbf{K}_r$ and $\mathbf{K}_f$ are designed separately based on the solutions of two different algebraic Riccati equations. When combined, the resulting sensor-based feedback is optimal

with the estimated state $\hat{\mathbf{a}}$ obtained by the optimal Kalman filter $\mathbf{K}_f$, as shown in Fig. 3.3. Thus, it is possible to design $\mathbf{K}_r$ and $\mathbf{K}_f$ separately by solving the respective Riccati equations in Eqs. (3.11) and (3.18) and then combine to form an optimal LQG controller; this is known as the *separation principle*.

Combining the full-state LQR control in Eq. (3.8) with the full-state Kalman filter estimate in Eq. (3.15), we obtain a dynamical system for the LQG controller where $\mathbf{s}$ is the input to the controller, $\mathbf{b}$ is the output of the controller, and the internal controller state is the full-state estimate $\hat{\mathbf{a}}$:

$$\frac{d}{dt}\hat{\mathbf{a}} = \left(\mathbf{A} - \mathbf{K}_f\mathbf{C} - \mathbf{B}\mathbf{K}_r\right)\hat{\mathbf{a}} + \mathbf{K}_f\mathbf{s} \tag{3.19a}$$

$$\mathbf{b} = -\mathbf{K}_r\hat{\mathbf{a}}. \tag{3.19b}$$

The LQG cost function is the ensemble-averaged value of Eq. (3.9),

$$J(t) = \left\langle \int_0^t \left[\mathbf{a}^T(\tau)\,\mathbf{Q}\,\mathbf{a}(\tau) + \mathbf{b}^T(\tau)\mathbf{R}\mathbf{b}(\tau)\right] d\tau \right\rangle, \tag{3.20}$$

where the angular brackets denote the average over many noise realizations.

Applying LQR to the full-state estimate $\hat{\mathbf{a}}$ results in the following state dynamics:

$$\frac{d}{dt}\mathbf{a} = \mathbf{A}\mathbf{a} - \mathbf{B}\mathbf{K}_r\hat{\mathbf{a}} + \mathbf{w}_d \tag{3.21a}$$

$$= \mathbf{A}\mathbf{a} - \mathbf{B}\mathbf{K}_r\mathbf{a} + \mathbf{B}\mathbf{K}_r\left(\mathbf{a} - \hat{\mathbf{a}}\right) + \mathbf{w}_d \tag{3.21b}$$

$$= \mathbf{A}\mathbf{a} - \mathbf{B}\mathbf{K}_r\mathbf{a} + \mathbf{B}\mathbf{K}_r\boldsymbol{\varepsilon} + \mathbf{w}_d, \tag{3.21c}$$

where $\boldsymbol{\varepsilon} = \mathbf{a} - \hat{\mathbf{a}}$ as before. We may finally write the closed-loop system as

$$\frac{d}{dt}\begin{bmatrix}\mathbf{a}\\ \boldsymbol{\varepsilon}\end{bmatrix} = \begin{bmatrix}\mathbf{A} - \mathbf{B}\mathbf{K}_r & \mathbf{B}\mathbf{K}_r\\ \mathbf{0} & \mathbf{A} - \mathbf{K}_f\mathbf{C}\end{bmatrix}\begin{bmatrix}\mathbf{a}\\ \boldsymbol{\varepsilon}\end{bmatrix} + \begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{I} & -\mathbf{K}_f\end{bmatrix}\begin{bmatrix}\mathbf{w}_d\\ \mathbf{w}_n\end{bmatrix}. \tag{3.22}$$

It is clear that if $\mathbf{K}_r$ and $\mathbf{K}_f$ were chosen to optimally place the closed-loop eigenvalues of $\mathbf{A} - \mathbf{B}\mathbf{K}_r$ and $\mathbf{A} - \mathbf{K}_f\mathbf{C}$ in the respective LQR and Kalman filter problems, then these are still the eigenvalues of the sensor-based closed-loop LQG controller.

The LQG framework assumes an accurate system model and knowledge of the measurement and process noise magnitude; moreover, the *Gaussian* in the title refers to the assumption that these noise terms are Gaussian white noise processes. In practice, all of these assumptions are dubious for many real-world systems, and even small amounts of model uncertainty can destroy the LQG performance and cause instability [86]. The entire optimization process above is often referred to as $\mathscr{H}_2$ optimal control. The optimization problem may be modified to promote robust controllers for systems that have model uncertainty [88, 89, 119], and these controllers are often referred to as $\mathscr{H}_\infty$ robust control laws. Intuitively, robust control penalizes the *worst-case* performance of a system, so that robustness is promoted. Often, an

LQG controller may be *robustified* through a process called loop transfer recovery, although this is beyond the scope of this book. An excellent treatment of robust control may be found in [93].

3.5 System Identification and Model Reduction

In many high-dimensional fluid problems, it is still possible to use linear control techniques, despite nonlinear equations of motion. For example, in fluid dynamics there are numerous success stories of linear model-based flow control, including transition delay in a spatially developing boundary layer on a flat plate and in channel flow [11–13, 29, 63, 101, 135, 136, 142, 245, 246], reducing skin-friction drag in wall turbulence [71, 72, 102, 159, 160], and stabilization of the cavity flow [55–57, 144, 228, 230, 232, 234, 235]. However, many of the linear control approaches do not scale well to large state spaces, and they may be prohibitively expensive to enact for real-time control on short timescales. It is therefore often necessary to first develop low-dimensional approximations of the full-state system for use with feedback control. There are two broad approaches to this problem: First, it is possible to start with a high-dimensional dynamical system, such as the discretized Navier-Stokes equations, and project the dynamics onto a low-dimensional subspace identified, for example, using proper orthogonal decomposition (POD) [28,138] and Galerkin projection. This results in a reduced-order model (ROM) [24, 221]. There are many approaches to this problem, including discrete empirical interpolation methods (DEIM) [60, 211], gappy POD [100], balanced proper orthogonal decomposition (BPOD) [231, 277], and many more. The second approach is to collect data from a simulation or an experiment and try to identify a low-dimensional model using data-driven techniques. This approach is typically called system identification, and is often preferred for control design because of the relative ease of implementation. Examples include the dynamic mode decomposition (DMD) [238, 270] and related Koopman analysis [187, 188, 229], the eigensystem realization algorithm (ERA) [151, 181], and the observer–Kalman filter identification (OKID) [150, 152, 214].

After a linear model has been identified, either by model reduction or system identification, it may then be used for model-based control design, as described above. However, there are a number of issues that may arise in practice, as linear model-based control might not work for a large class of problems. First, the system being modeled may be strongly nonlinear, in which case the linear approximation might only capture a small portion of the dynamic effects. Next, the system may be stochastically driven, so that the linear model will average out the relevant fluctuations. Finally, when control is applied to the full system, the attractor dynamics may change, rendering the linearized model invalid. Exceptions include the stabilization of laminar solutions in fluid mechanics, where feedback control rejects nonlinear disturbances and keeps the system close to the fixed point where linearization is useful.

There are certainly alternative methods for system identification and model reduction that are nonlinear, involve stochasticity, and change with the attractor. However, these methods are typically advanced and they also may limit the available machinery from control theory.

3.5.1 System Identification

System identification may be thought of as a form of machine learning, where an input–output map of a system is learned from training data in a representation that generalizes to data that was not in the training set. There is a vast literature on methods for system identification [150,174], which is beyond the scope of this treatment, although many of the leading methods are based on a form of dynamic regression that fits models based on data. For this section, we consider the eigensystem realization algorithm (ERA) and observer-Kalman filter identification (OKID) methods because of their connection to balanced model reduction [181, 192, 231, 270] and their recent successful application in closed-loop flow control [11, 13, 143]. The ERA/OKID procedure is also applicable to multiple-input, multiple-output (MIMO) systems.

3.5.2 Eigensystem Realization Algorithm (ERA)

The eigensystem realization algorithm produces low-dimensional linear input–output models from sensor measurements of an impulse response experiment, and it is based on the "minimal realization" theory of Ho and Kalman [133]. The modern theory was developed to identify structural models for various spacecraft [151], and it has been shown by Ma et al. [181] that ERA models are equivalent to BPOD models.[1] ERA is based entirely on impulse response measurements and does not require prior knowledge of a model.

Given a linear system, as in Eq. (3.1), it is possible to obtain a discrete-time version:

$$\mathbf{a}_{k+1} = \mathbf{A}_d \mathbf{a}_k + \mathbf{B}_d \mathbf{b}_k \tag{3.23a}$$

$$\mathbf{s}_k = \mathbf{C}_d \mathbf{a}_k + \mathbf{D}_d \mathbf{b}_k, \tag{3.23b}$$

where subscript k denotes the time and Δt the corresponding timestep, so that $\mathbf{a}_k = \mathbf{a}(t_k) = \mathbf{a}(k\Delta t)$. The matrices in the discrete-time system are denoted with a subscript d and are related to the original continuous-time system matrices as:

[1] BPOD and ERA models both balance the empirical Gramians and approximate balanced truncation [192] for high-dimensional systems.

$$\mathbf{A}_d = \exp(\mathbf{A}\Delta t) \tag{3.24a}$$

$$\mathbf{B}_d = \int_0^{\Delta t} \exp(\mathbf{A}\tau)\mathbf{B}d\tau \tag{3.24b}$$

$$\mathbf{C}_d = \mathbf{C} \tag{3.24c}$$

$$\mathbf{D}_d = \mathbf{D}. \tag{3.24d}$$

Now, a discrete-time delta function input in the actuation $\mathbf{b}$:

$$\mathbf{b}_k^\delta \triangleq \mathbf{b}^\delta(k\Delta t) = \begin{cases} \mathbf{I}, \ k = 0 \\ \mathbf{0}, \ k = 1, 2, 3, \cdots \end{cases} \tag{3.25}$$

gives rise to a discrete-time impulse response in the sensors $\mathbf{s}$:

$$\mathbf{s}_k^\delta \triangleq \mathbf{s}^\delta(k\Delta t) = \begin{cases} \mathbf{D}_d, \ k = 0 \\ \mathbf{C}_d\mathbf{A}_d^{k-1}\mathbf{B}_d, \ k = 1, 2, 3, \cdots . \end{cases} \tag{3.26}$$

In an experiment or simulation, typically N_b impulse responses are performed, one for each of the N_b separate input channels. The output responses are collected for each impulsive input, and at a given time-step k, the output vector in response to the jth impulsive input will form the jth column of $\mathbf{s}_k^\delta$. Thus, each of the $\mathbf{s}_k^\delta$ is a $N_s \times N_b$ matrix.

A Hankel matrix $\mathbf{H}$ is formed by stacking shifted time-series of impulse-response measurements into a matrix:

$$\mathbf{H} = \begin{bmatrix} \mathbf{s}_1^\delta & \mathbf{s}_2^\delta & \cdots & \mathbf{s}_{m_c}^\delta \\ \mathbf{s}_2^\delta & \mathbf{s}_3^\delta & \cdots & \mathbf{s}_{m_c+1}^\delta \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{s}_{m_o}^\delta & \mathbf{s}_{m_o+1}^\delta & \cdots & \mathbf{s}_{m_c+m_o-1}^\delta \end{bmatrix}. \tag{3.27}$$

This matrix is closely related to the empirical discrete-time observability and controllability Gramians, $\mathbf{W}_{\mathscr{O}}^d = \mathscr{O}_{\boldsymbol{d}}{}^*\mathscr{O}_{\boldsymbol{d}}$ and $\mathbf{W}_{\mathscr{C}}^d = \mathscr{C}_{\boldsymbol{d}}\mathscr{C}_{\boldsymbol{d}}{}^*$. Substituting the expression from Eq. (3.26) into Eq. (3.27) yields:

$$\mathbf{H} = \begin{bmatrix} \mathbf{C}_d\mathbf{B}_d & \mathbf{C}_d\mathbf{A}_d\mathbf{B}_d & \cdots & \mathbf{C}_d\mathbf{A}_d^{m_c-1}\mathbf{B}_d \\ \mathbf{C}_d\mathbf{A}_d\mathbf{B}_d & \mathbf{C}_d\mathbf{A}_d^2\mathbf{B}_d & \cdots & \mathbf{C}_d\mathbf{A}_d^{m_c}\mathbf{B}_d \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{C}_d\mathbf{A}_d^{m_o-1}\mathbf{B}_d & \mathbf{C}_d\mathbf{A}_d^{m_o}\mathbf{B}_d & \cdots & \mathbf{C}_d\mathbf{A}_d^{m_c+m_o-2}\mathbf{B}_d \end{bmatrix} \tag{3.28a}$$

$$= \begin{bmatrix} \mathbf{C}_d \\ \mathbf{C}_d\mathbf{A}_d \\ \vdots \\ \mathbf{C}_d\mathbf{A}_d^{m_o-1} \end{bmatrix} \begin{bmatrix} \mathbf{B}_d & \mathbf{A}_d\mathbf{B}_d & \cdots & \mathbf{A}_d^{m_c-1}\mathbf{B}_d \end{bmatrix} = \mathscr{O}_{\boldsymbol{d}}\mathscr{C}_{\boldsymbol{d}}. \tag{3.28b}$$

Taking the singular value decomposition (SVD) of this Hankel matrix yields the dominant temporal patterns in this time-series:

$$\mathbf{H} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^* = \begin{bmatrix}\mathbf{U}_r & \mathbf{U}_s\end{bmatrix}\begin{bmatrix}\boldsymbol{\Sigma}_r & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_s\end{bmatrix}\begin{bmatrix}\mathbf{V}_r^* \\ \mathbf{V}_s^*\end{bmatrix} \approx \mathbf{U}_r\boldsymbol{\Sigma}_r\mathbf{V}_r^*. \tag{3.29}$$

Notice that we may truncate all small singular values in $\boldsymbol{\Sigma}_s$ and only keep the first r singular values in $\boldsymbol{\Sigma}_r$. The columns of $\mathbf{U}_r$ and $\mathbf{V}_r$ may be thought of as *eigen*-time-delay coordinates.

With sensor measurements from an impulse-response experiment, it is also possible to create a second, shifted Hankel matrix $\mathbf{H}'$:

$$\mathbf{H}' = \begin{bmatrix} \mathbf{s}_2^\delta & \mathbf{s}_3^\delta & \cdots & \mathbf{s}_{m_c+1}^\delta \\ \mathbf{s}_3^\delta & \mathbf{s}_4^\delta & \cdots & \mathbf{s}_{m_c+2}^\delta \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{s}_{m_o+1}^\delta & \mathbf{s}_{m_o+2}^\delta & \cdots & \mathbf{s}_{m_c+m_o}^\delta \end{bmatrix} \tag{3.30a}$$

$$= \begin{bmatrix} \mathbf{C}_d\mathbf{A}_d\mathbf{B}_d & \mathbf{C}_d\mathbf{A}_d^2\mathbf{B}_d & \cdots & \mathbf{C}_d\mathbf{A}_d^{m_c}\mathbf{B}_d \\ \mathbf{C}_d\mathbf{A}_d^2\mathbf{B}_d & \mathbf{C}_d\mathbf{A}_d^3\mathbf{B}_d & \cdots & \mathbf{C}_d\mathbf{A}_d^{m_c+1}\mathbf{B}_d \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{C}_d\mathbf{A}_d^{m_o}\mathbf{B}_d & \mathbf{C}_d\mathbf{A}_d^{m_o+1}\mathbf{B}_d & \cdots & \mathbf{C}_d\mathbf{A}_d^{m_c+m_o-1}\mathbf{B}_d \end{bmatrix} = \mathscr{O}_d\mathbf{A}\mathscr{C}_d. \tag{3.30b}$$

Based on the matrices $\mathbf{H}$ and $\mathbf{H}'$, we are able to construct a reduced-order model as follows:

$$\mathbf{A}_r = \boldsymbol{\Sigma}_r^{-1/2}\mathbf{U}_r^*\mathbf{H}'\mathbf{V}_r\boldsymbol{\Sigma}_r^{-1/2}; \tag{3.31a}$$

$$\mathbf{B}_r = \text{ first } N_b \text{ columns of } \boldsymbol{\Sigma}_r^{1/2}\mathbf{V}^*; \tag{3.31b}$$

$$\mathbf{C}_r = \text{ first } N_s \text{ columns of } \mathbf{U}\boldsymbol{\Sigma}_r^{1/2}. \tag{3.31c}$$

Thus, we express the input–output dynamics in terms of a reduced system with a low-dimensional state:

$$\tilde{\mathbf{a}}_{k+1} = \mathbf{A}_r\tilde{\mathbf{a}}_k + \mathbf{B}_r\mathbf{b} \tag{3.32a}$$

$$\mathbf{s} = \mathbf{C}_r\tilde{\mathbf{a}}_k. \tag{3.32b}$$

$\mathbf{H}$ and $\mathbf{H}'$ are constructed from impulse response simulations/experiments, without the need for storing direct or adjoint snapshots, as in other balanced model reduction techniques. However, if full-state snapshots are available (for example, by collecting velocity fields in simulations or PIV experiments), it is then possible to construct direct modes. These full-state snapshots form $\mathscr{C}_d$, and modes can be constructed by:

$$\boldsymbol{\Phi}_r = \mathscr{C}_d\mathbf{V}_r\boldsymbol{\Sigma}_r^{-1/2}. \tag{3.33}$$

These modes may then be used to approximate the full state of the high-dimensional system from the low-dimensional model in Eq. (3.32) by:

$$\mathbf{a} \approx \boldsymbol{\Phi}_r \tilde{\mathbf{a}}. \tag{3.34}$$

ERA balances the empirical controllability and observability Gramians, $\mathscr{O}_d \mathscr{O}_d^*$ and $\mathscr{C}_d^* \mathscr{C}_d$. Unless we collect a very large amount of data, the true Gramians are only approximately balanced. Instead of collecting long tails of data, it is possible to collect data until the Hankel matrix is full rank, balance the full-rank identified model, and then truncate. This is more efficient than collecting snapshots until all transients have decayed; this idea is developed in [177, 269].

3.5.3 *Observer Kalman Filter Identification (OKID)*

OKID was developed to compliment the ERA for lightly damped experimental systems with noise [152]. In practice, performing isolated impulse response experiments is challenging, and the effect of measurement noise can contaminate results. Moreover, if there is a large separation of timescales, then a tremendous amount of data must be collected to use ERA. This section poses the general problem of approximating the impulse response from arbitrary input/output data. Typically, one would identify reduced-order models according to the following general procedure, shown in Fig. 3.4:

1. Collect the output response to a pseudo-random input.
2. This information is passed through the OKID algorithm to obtain the de-noised linear impulse response.
3. The impulse response is passed through the ERA to obtain a reduced-order state-space system.

The output $\mathbf{s}_k$ in response to a general input signal $\mathbf{b}_k$, for zero initial condition $\mathbf{x}_0 = \mathbf{0}$, is given by:

$$\begin{aligned}
\mathbf{s}_0 &= \mathbf{D}_d \mathbf{b}_0 && \text{(3.35a)}\\
\mathbf{s}_1 &= \mathbf{C}_d \mathbf{B}_d \mathbf{b}_0 + \mathbf{D}_d \mathbf{b}_1 && \text{(3.35b)}\\
\mathbf{s}_2 &= \mathbf{C}_d \mathbf{A}_d \mathbf{B}_d \mathbf{b}_0 + \mathbf{C}_d \mathbf{B}_d \mathbf{b}_1 + \mathbf{D}_d \mathbf{b}_2 && \text{(3.35c)}\\
&\cdots \\
\mathbf{s}_k &= \mathbf{C}_d \mathbf{A}_d^{k-1} \mathbf{B}_d \mathbf{b}_0 + \mathbf{C}_d \mathbf{A}_d^{k-2} \mathbf{B}_d \mathbf{b}_1 + \cdots + \mathbf{C}_d \mathbf{B}_d \mathbf{b}_{k-1} + \mathbf{D}_d \mathbf{b}_k. && \text{(3.35d)}
\end{aligned}$$

Note that there is no $\mathbf{C}$ term in the expression for $\mathbf{s}_0$ since there is zero initial condition $\mathbf{x}_0 = \mathbf{0}$. This progression of measurements $\mathbf{s}_k$ may be further simplified and expressed in terms of impulse-response measurements $\mathbf{s}_k^\delta$:

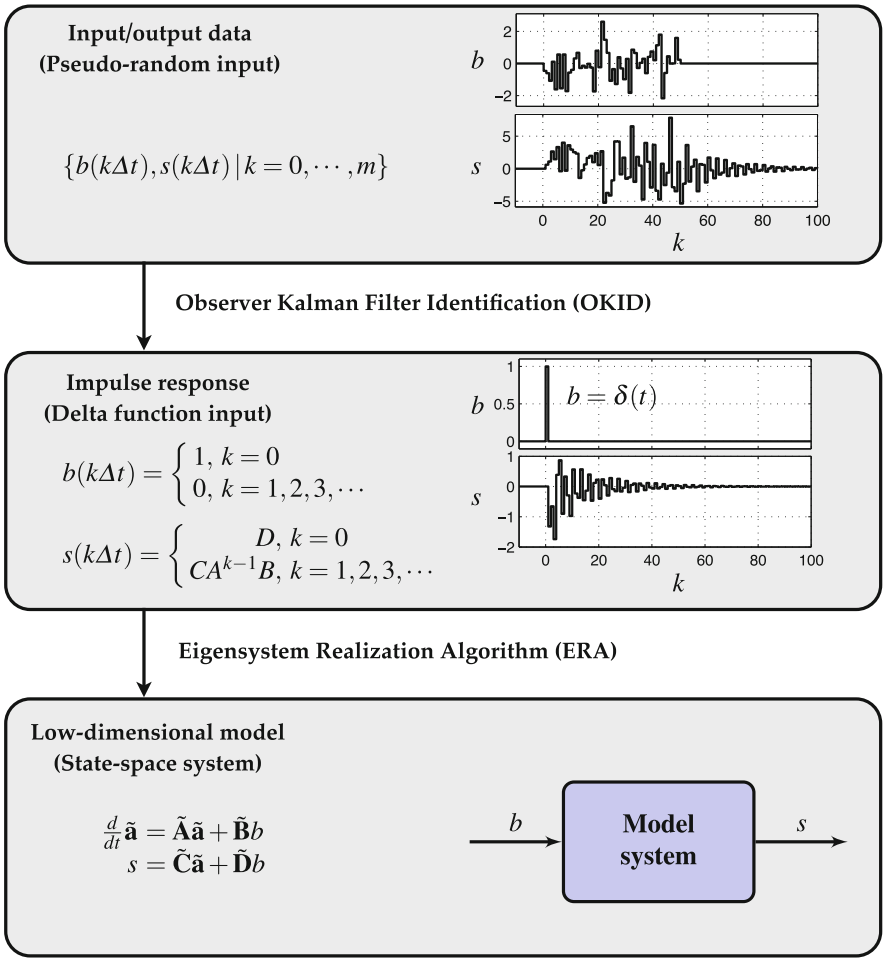


Fig. 3.4 Schematic illustrating the use of OKID followed by ERA to identify a low-dimensional state-space model, based on measurement data. The schematic illustrates the single-input single-output (SISO) case, although both methods are general and handle multiple-input multiple-output (MIMO) systems

$$\underbrace{\begin{bmatrix}\mathbf{s}_0 & \mathbf{s}_1 & \cdots & \mathbf{s}_N\end{bmatrix}}_{\mathscr{S}} = \underbrace{\begin{bmatrix}\mathbf{s}_0^\delta & \mathbf{s}_1^\delta & \cdots & \mathbf{s}_N^\delta\end{bmatrix}}_{\mathscr{S}^\delta} \underbrace{\begin{bmatrix}\mathbf{b}_0 & \mathbf{b}_1 & \cdots & \mathbf{b}_N \\ \mathbf{0} & \mathbf{b}_0 & \cdots & \mathbf{b}_{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{b}_0\end{bmatrix}}_{\mathscr{B}}. \tag{3.36}$$

It is often possible to invert the matrix of control inputs, $\mathscr{B}$, to solve for the Markov parameters $\mathscr{S}^\delta$. However, $\mathscr{B}$ may be sparsely populated, so that either it is

un-invertible, or inversion is ill-conditioned. In addition, $\mathscr{B}$ is large for lightly damped systems, making inversion computationally expensive. Finally, noise is not optimally filtered by simply inverting $\mathscr{B}$ to solve for the Markov parameters.

The OKID method addresses each of these issues. Instead of the original discrete-time system, we now introduce an optimal observer system:

$$\hat{\mathbf{a}}_{k+1} = \mathbf{A}_d \hat{\mathbf{a}}_k + \mathbf{K}_f \left(\hat{\mathbf{s}}_k - \mathbf{s}_k\right) + \mathbf{B}_d \mathbf{b}_k \tag{3.37a}$$

$$\hat{\mathbf{s}}_k = \mathbf{C}_d \hat{\mathbf{a}}_k + \mathbf{D}_d \mathbf{b}_k, \tag{3.37b}$$

which may be re-written as:

$$\hat{\mathbf{a}}_{k+1} = \underbrace{\left(\mathbf{A}_d + \mathbf{K}_f \mathbf{C}_d\right)}_{\bar{\mathbf{A}}_d} \hat{\mathbf{a}}_k + \underbrace{\left[\mathbf{B}_d + \mathbf{K}_f \mathbf{D}_d, -\mathbf{K}_f\right]}_{\bar{\mathbf{B}}_d} \begin{bmatrix} \mathbf{b}_k \\ \mathbf{s}_k \end{bmatrix}. \tag{3.38}$$

Recall from above that if the system is observable, it is possible to place the poles of $\mathbf{A}_d + \mathbf{K}_f \mathbf{C}_d$ anywhere we like. However, depending on the amount of noise in the measurements and structural disturbance in our model, there are *optimal* pole locations that are given by the *Kalman filter* (recall Sect. 3.3). We may now solve for the *observer Markov parameters* $\bar{\mathscr{Y}}^\delta$ of this system in terms of measured inputs and outputs according to the following algorithm from [152]:

1. Choose the number of observer Markov parameters to identify, p.
2. Construct the data matrices below:

$$\mathscr{S} = \begin{bmatrix} \mathbf{s}_0 & \mathbf{s}_1 & \cdots & \mathbf{s}_p & \cdots & \mathbf{s}_M \end{bmatrix} \tag{3.39}$$

$$\mathscr{C} = \begin{bmatrix} \mathbf{b}_0 & \mathbf{b}_1 & \cdots & \mathbf{b}_p & \cdots & \mathbf{b}_M \\ \mathbf{0} & \mathbf{v}_0 & \cdots & \mathbf{v}_{p-1} & \cdots & \mathbf{v}_{M-1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{v}_0 & \cdots & \mathbf{v}_{M-p} \end{bmatrix} \tag{3.40}$$

   where $\mathbf{v}_i = \begin{bmatrix} \mathbf{b}_i^T & \mathbf{s}_i^T \end{bmatrix}^T$.

   The matrix $\mathscr{C}$ resembles $\mathscr{B}$, except that is has been augmented with the outputs $\mathbf{s}_i$. In this way, we are working with a system that is augmented to include a Kalman filter. We are now identifying the observer Markov parameters of the *augmented* system, $\bar{\mathscr{Y}}^\delta$, using the equation $\mathscr{S} = \bar{\mathscr{Y}}^\delta \mathscr{C}$.
3. Identify the matrix $\bar{\mathscr{Y}}^\delta$ of observer Markov parameters by solving $\mathscr{S} = \bar{\mathscr{Y}}^\delta \mathscr{C}$ for $\bar{\mathscr{Y}}^\delta$ using the right pseudo-inverse of $\mathscr{C}$ (i.e., SVD).
4. Recover system Markov parameters, $\mathscr{Y}^\delta$, from the observer Markov parameters, $\bar{\mathscr{Y}}^\delta$.

(a) Order the observer Markov parameters $\bar{\mathscr{Y}}^{\delta}$ as:

$$\bar{\mathscr{Y}}_0^{\delta} = \mathbf{D}, \tag{3.41}$$

$$\bar{\mathscr{Y}}_k^{\delta} = \left[(\bar{\mathscr{Y}}^{\delta})_k^{(1)} \ (\bar{\mathscr{Y}}^{\delta})_k^{(2)} \right] \text{ for } k \geq 1, \tag{3.42}$$

where $(\bar{\mathscr{Y}}^{\delta})_k^{(1)} \in \mathbb{R}^{q \times p}$, $(\bar{\mathscr{Y}}^{\delta})_k^{(2)} \in \mathbb{R}^{q \times q}$, and $\mathbf{s}_0^{\delta} = \bar{\mathscr{Y}}_0^{\delta} = \mathbf{D}$.

(b) Reconstruct system Markov parameters:

$$\mathbf{s}_k^{\delta} = (\bar{\mathscr{Y}}^{\delta})_k^{(1)} + \sum_{i=1}^{k} (\bar{\mathscr{Y}}^{\delta})_i^{(2)} \mathbf{s}_{k-i}^{\delta} \text{ for } k \geq 1. \tag{3.43}$$

Thus, the OKID method identifies the Markov parameters of a system augmented with an asymptotically stable Kalman filter. The system Markov parameters are extracted from the observer Markov parameters by Eq. (3.43). These system Markov parameters approximate the impulse response of the system, and may be used directly as inputs to the ERA algorithm.

There are numerous extensions of the ERA/OKID methods. For example, there are generalizations for linear parameter varying (LPV) systems and systems linearized about a limit cycle. We will implement the ERA/OKID method in Sect. 6.4 on a turbulent mixing layer experiment. This example will demonstrate the limited usefulness of linear system identification for strongly nonlinear systems.

3.6 Exercises

Exercise 3–1: Show that the following system is controllable but not observable:

$$\frac{d}{dt} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 0 & -2 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} b \tag{3.44a}$$

$$s = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}. \tag{3.44b}$$

How might we change the matrix $\mathbf{C} = \begin{bmatrix} 0 & 1 \end{bmatrix}$ to make the system observable?

Exercise 3–2: Develop an optimal LQR controller for the following system:

$$\frac{d}{dt} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} b. \tag{3.45}$$

(a) In particular, solve for the gain matrix $\mathbf{K}_r$ so that $b = -\mathbf{K}_r \mathbf{a}$ minimizes the cost function:

$$J = \int_0^\infty \left[\mathbf{a}^T(\tau)\, \mathbf{Q}\, \mathbf{a}(\tau) + R\, b^2(\tau) \right] d\tau, \quad \mathbf{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1. \tag{3.46}$$

(b) Now show that nearby controllers with controller gain $1.1\mathbf{K}_r$ and $.9\mathbf{K}_r$ are suboptimal.
(c) Finally, solve for the optimal $\mathbf{K}_r$ using a genetic algorithm. You will likely need to approximate the cost function J by integrating to a finite but long time until transients decay.

Exercise 3–3: Develop a Kalman filter for the following system:

$$\frac{d}{dt} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} -.01 & 1 \\ -1 & -.01 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} b + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} w_{d_1} \\ w_{d_2} \end{bmatrix} \tag{3.47a}$$

$$s = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} + w_n. \tag{3.47b}$$

(a) Simulate the system with measurement and process noise with forcing $b = \sin(t)$ and plot the Kalman filter prediction of the state. You can compare this to the full-state of the true system by using the same $\mathbf{A}$ and $\mathbf{B}$ matrices above but using $\mathbf{C} = \mathbf{I}$ to output the full state $\mathbf{a}$.
(b) Now, using the same Kalman filter above, increase the process noise (disturbance) by a factor of 5. How does this change the full-state prediction?
(c) For a range of process and measurement noise magnitudes, compute the Kalman filter. How do the eigenvalues of the full-state estimator change with the various noise magnitudes? Is there a relationship?

Exercise 3–4: Consider the following system

$$\frac{d}{dt} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} -2 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} b \tag{3.48a}$$

$$s = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}. \tag{3.48b}$$

(a) Compute an LQR controller for the matrix pair $\mathbf{A}$ and $\mathbf{B}$.
(b) Compute a Kalman filter for the matrix pair $\mathbf{A}$ and $\mathbf{C}$.
(c) Now, compute the closed-loop system in Matlab® by implementing LQG control. Show that the closed-loop eigenvalues are the same as the LQR and Kalman filter eigenvalues from above.

Exercise 3–5: Consider the following linear system:

$$\frac{d}{dt} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} -1.0 & 0.001 \\ 0 & -0.99 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} b \tag{3.49a}$$

$$s = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}. \tag{3.49b}$$

(a) Construct a time-series of impulse-response data from this system using a sufficiently small Δt to resolve the dynamics.
(b) How many terms of the time series are required before the rank of the Hankel matrix in Eq. (3.27) saturates? Why is this true?
(c) Use the eigensystem realization algorithm to determine a model from the time-series.
(d) Now, add a small amount of measurement noise to the time-series. How many terms are required to capture the system dynamics now? How does this number change as the noise magnitude is increased?

3.7 Suggested Reading

Texts

(1) **Feedback Systems: An Introduction for Scientists and Engineers**, by K.J. Aström and R.M. Murray, 2010 [223].
This is an excellent introductory text that provides a number of motivating examples. The control problem is formulated in state-space, which is beneficial for students with a strong mathematical background.
(2) **Feedback Control Theory**, by J.C. Doyle, B.A. Francis, and A.R. Tannenbaum, 2013 [87].
This text strikes a delicate balance between simple introductory concepts and advanced topics in robust control. The authors largely defined this field, and this book is essential reading.
(3) **Multivariable Feedback Control: Analysis and Design**, by S. Skogestad and I. Postlethwaite, 2005 [252].
This is perhaps the most complete and practically useful guide for real-world engineering control. It strikes a delicate balance between historical, theoretical, and practical advice for the advanced control practitioner.
(4) **A Course in Robust Control Theory: A Convex Approach**, by G.E. Dullerud and F. Paganini, 2000 [93].
This text provides an excellent treatment of the mathematical foundations of linear control theory. There is a considerable focus on computational aspects, including the use of methods from convex optimization.
(5) **Optimal Control and Estimation**, by R.F. Stengel, 2012 [255].
This book provides a comprehensive overview and derivation of optimal control, including advanced methods such as neighboring optimal control. This text covers estimation and forecasting with a subtle balance between dynamical systems and probability.

Seminal Papers

(1) **Guaranteed margins for LQG regulators**, by J.C. Doyle, *IEEE Transactions on Automatic Control*, 1978 [86].
This paper turned the world of control upside down. With a simple counterexample, Doyle showed the possible lack of robustness of LQG regulators.

(2) **Principal component analysis in linear systems: Controllability, observability, and model reduction**, by B.C. Moore, *IEEE Transactions on Automatic Control*, 1981 [192].
This paper connects dimensionality reduction with controllability and observability, paving the way towards modern techniques in model reduction.

(3) **Identification of linear parameter varying models**, by B. Bamieh and L. Giarré, *International Journal of Robust and Nonlinear Control*, 2002 [16].
This paper describes how to identify a parameterized family of locally linear models that may be used for gain-scheduled control.

Chapter 4
Benchmarking MLC Against Linear Control

All stable processes we shall predict. All unstable processes we shall control.

John von Neumann

We have now developed two powerful approaches to design control laws: the machine learning control (MLC) approach from Chap. 2, and the classical optimal linear control theory from Chap. 3. Both approaches have benefits and trade-offs. Linear control theory yields concise control laws that are solutions to optimization problems, providing the best possible control laws for linear systems that are well characterized by accurate input–output models. In fact, we may view the MLC approach as an alternative optimization procedure to determine these classical controllers that generalizes naturally to nonlinear problems in a model-free context.

In this chapter, we demonstrate the use of genetic programming for MLC on linear systems where optimal control laws are known. In particular, we benchmark MLC against the linear quadratic regulator (LQR) for full-state feedback in Sect. 4.1, the Kalman filter for noisy state estimation in Sect. 4.2, and linear quadratic Gaussian (LQG) for optimal sensor-based feedback in Sect. 4.3. As an example system, we consider an unstable linear oscillator, which mimics many instabilities that occur in fluid dynamics. Next, we compare MLC with linear optimal control on systems with increasing nonlinearity in Sect. 4.4. Exercises are provided in Sect. 4.5. We conclude the chapter in Sect. 4.6 with an interview of Professor Shervin Bagheri who is a pioneer and a leading scholar in model-based closed-loop flow control.

T. Duriez et al., *Machine Learning Control – Taming Nonlinear Dynamics and Turbulence*, Fluid Mechanics and Its Applications 116,
DOI 10.1007/978-3-319-40624-4_4

4.1 Comparison of MLC with LQR on a Linear Oscillator

As discussed in Chap. 3, control design often begins with the implementation of a full-state feedback regulator. The assumption of full-state measurements will later be relaxed, necessitating dynamic estimation from limited sensor measurements.

Machine learning control using genetic programming provides an approach to optimize controller design given a well-posed cost function. Thus, it is possible to test the ability of MLC to discover the known optimal linear quadratic regulator (LQR) solution from Sect. 3.2. In particular, we consider an unstable linear oscillator with two states:

$$\frac{d}{dt}a_1 = \sigma a_1 - \omega a_2 \tag{4.1a}$$

$$\frac{d}{dt}a_2 = \omega a_1 + \sigma a_2 + b. \tag{4.1b}$$

We assume full-state measurements, $\mathbf{s} = \mathbf{a}$, although we will relax this assumption in the following sections. In the state space form of Eq. (3.1), this system is given by the following system matrices:

$$\mathbf{A} = \begin{bmatrix} \sigma & -\omega \\ \omega & \sigma \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \tag{4.2a}$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \mathbf{D} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{4.2b}$$

For this example $\sigma = \omega = 1$, corresponding to an unstable linear growth rate. The following LQR cost function weights are used:

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1. \tag{4.3}$$

The LQR optimal controller from Eq. (3.10) is given by $b = -\mathbf{K}_r\mathbf{a}$ with

$$\mathbf{K}_r = \begin{bmatrix} -4.8783 & 4.4288 \end{bmatrix}. \tag{4.4}$$

The cost is monitored as a function of time, as in Chap. 2:

$$J(t) = \int_0^t \left[\mathbf{a}^T(\tau)\,\mathbf{Q}\,\mathbf{a}(\tau) + R\,b^2(\tau)\right] d\tau. \tag{4.5}$$

If time is omitted as an argument, the converged value $\lim_{t\to\infty} J(t)$ is used.

The Matlab® implementation is shown in Code 4.1, and the closed-loop LQR response is shown in Fig. 4.1 (solid curves). Notice that there are two large periods of growth in the cost function J corresponding to large magnitude of control

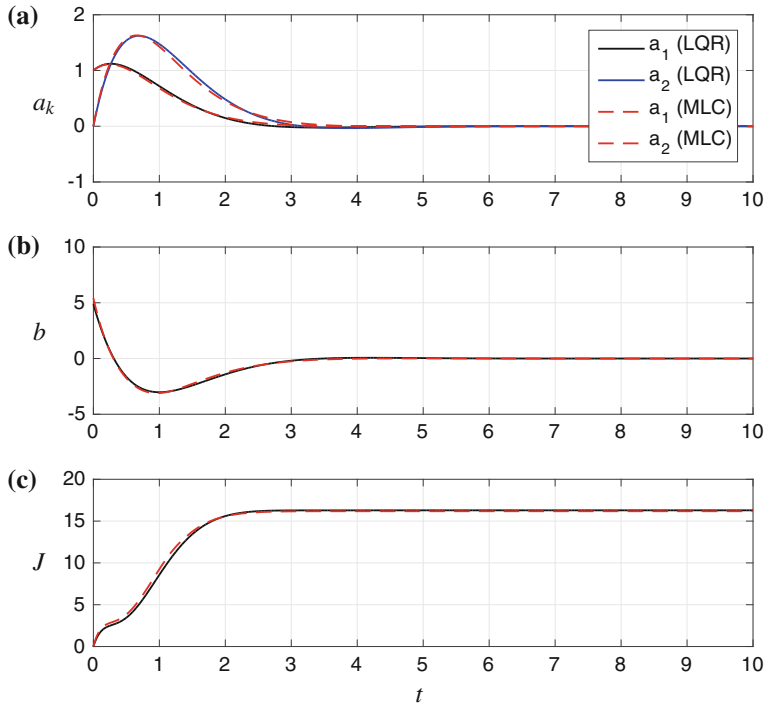


Fig. 4.1 Response of the unstable oscillator system in Eq. (4.2) with LQR control (*solid lines*). The MLC response is also shown (*dashed red lines*), and it agrees with the LQR controller. (**a**) The state initially grows and is eventually stabilized. (**b**) The actuation signal b is aggressive at the beginning until the system is driven closer to the origin. (**c**) The cost J (4.5) initially rises because of a large actuation expenditure b, and then continues to rise until the state decays

Table 4.1 Main parameters used for MLC solution of LQR problem

Parameter	N_i	N_s	N_b	P_r	P_m	P_c	N_p	N_e	Node functions
Value	1000	2	1	0.1	0.4	0.5	7	10	$+, -, \times$

expenditure. Eventually, the cost function converges to a final value after the state has been stabilized and the actuation input shrinks to zero.

An MLC controller is constructed using genetic programming with simple operations $(+, -, \times)$ and the same cost function as in Eq. (4.5). The optimal controller is simply a proportional feedback on the state measurements $\mathbf{a}$, so this presents a simple test case for the MLC architecture. The MLC controller is also shown in Fig. 4.1 (dashed curves), and the response is extremely close to that of the LQR controller.

The MLC implementation is given below with the parameters of Table 4.1.

Code 4.1 LQR implementation in Matlab®

```
clear all, close all, clc

% Define system matrices
sigma=1, omega=1;    % Unstable oscillator parameters
A = [sigma -omega;   % Dynamics
    omega sigma];
B = [0; 1];          % Actuation on second state
C = [1 0;            % Full-state measurements
    0 1];
D = [0; 0];          % No feedthrough term
sys = ss(A,B,C,D);   % Continuous-time state-space system

% Compute LQR controller
Q = eye(2);          % State cost is 2x2 identity matrix
R = 1;               % Actuation cost is 1
[K,S,e] = lqr(A,B,Q,R);  % Optimal gain matrix K from LQR

% Simulate closed-loop system
dt = 0.001;
Acl = A-B*K;          % Closed-loop dynamics
Bcl = [0; 0];         % No input after closing-loop
sysK = ss(Acl,Bcl,C,D);  % Closed-loop system
[s,t] = initial(sysK,[1; 0],0:dt:10);  % Intial condition
    response

% Compute cost function
b = -K*s';            % Actuator signal
J(1) = 0;             % Initialize cost J=0
% For each dt, integrate cost function
for k=2:length(t)
    J(k) = J(k-1)+dt*(s(k-1,:)*Q*s(k-1,:)' + R*b(k-1)^2);
end
```

The following code initializes the MLC problem, runs 50 generations, and displays the results.

```
mlc=MLC('MLC_ex_LQE_problem'); % Creates the MLC problem
mlc.go(50);                    % Runs MLC for 50 generations
mlc.show_best_individual       % Displays the results
```

The evaluation function can be displayed by using the command:

```
open('MLC_evaluator_LQR')
```

As in Sect. 2.3.2, the evaluation function implements a dynamical system by representing the time derivatives as functions of the states and the control law representing the individual. Then `ode45` is used to integrate the dynamical system and the cost function value is computed and returned. In case the numerical integration diverges, a predefined high value is returned for J.

Although MLC achieves near-optimal performance in the LQR problem, we had the advantage of full state measurements. To explore the case with limited mea-

surements, we must explore an extension of MLC that includes temporal filters as function blocks to estimate the full state from a time-history of sensor measurements.

4.2 Comparison of MLC with Kalman Filter on a Noisy Linear Oscillator

In practice, full-state measurements of the system are often unavailable or may be prohibitively expensive to collect and process in real-time. Instead, it is typically necessary to collect limited sensor measurements and *reconstruct* the relevant state through dynamic estimation, for example using the Kalman filter from Sect. 3.3.

In a high-dimensional fluid, even reconstructing the state through estimation may be computationally expensive, introducing unacceptable time delays in the control loop. Instead reduced-order models are generally used to describe the few states that are most controllable and observable. A more challenging test of MLC involves full-state estimation from limited noisy sensor measurements. As an illustrative example, consider a neutrally stable oscillator with no forcing:

$$\frac{d}{dt}a_1 = \sigma a_1 - \omega a_2 + w_{d,1} \tag{4.6a}$$

$$\frac{d}{dt}a_2 = \omega a_1 + \sigma a_2 + w_{d,2} \tag{4.6b}$$

$$s = a_1 + w_n. \tag{4.6c}$$

Again, this corresponds to a linear system with the following system matrices

$$\mathbf{A} = \begin{bmatrix} \sigma & -\omega \\ \omega & \sigma \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \tag{4.7a}$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix}, \qquad \mathbf{D} = \begin{bmatrix} 0 \end{bmatrix}. \tag{4.7b}$$

In this example, we will consider $\sigma = 0$ and $\omega = 1$, corresponding to a neutrally stable oscillator. Finally, the disturbance and noise covariance are given by:

$$\mathbf{V}_d = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{V}_n = 0.1. \tag{4.8}$$

The cost function quantifies the accuracy of the estimation, as in Eq. (3.16).

The Matlab® implementation is shown in Code 4.2, and the full-state Kalman filter estimate is shown in Fig. 4.2. The noisy sensor measurement s is shown in the middle panel for a single noise realization. An ensemble of square-summed errors are shown in the bottom panel, and the ensemble average is the cost function J, as in Eq. (3.16). Because there is constantly error introduced through noisy measurements, the cost function continues to increase for all time.

To compare the MLC solution, it is necessary to first generalize the function tree representation beyond a static input–output map. In particular, we envision two

methods of generalizing a function tree to achieve dynamic estimation: First, it is possible to have nodes that accumulate information by integration (see Fig. 4.3), and second, it is possible to have function expressions for the numerator and denominator of a transfer function (see Fig. 4.4).

Code 4.2 LQE implementation in Matlab®

```
% Define system matrices
sigma = 0, omega = 1;     % Neutrally stable oscillator
A = [sigma -omega;        % Dynamics
    omega sigma];
B = [eye(2) [0; 0] ];     % Disturbance plus actuation
C = [1 0];                % Measure first state
D = 0;                    % No feedthrough term
sys = ss(A,B,C,D);        % Continuous state-space system

% Disturbance and noise covariance matrices
Vd = eye(2);              % Disturbance covariance
Vn = .1;                  % Noise covariance
Vdn = [0; 0];             % No cross-terms

% Compute Kalman filter using LQE
[L,P,E] = lqe(A,eye(2),C,Vd,Vn,Vdn);  % L is gain matrix
Aest = A-L*C;               % Estimator dynamics
Best = L;                   % Input to estimator
Cest = eye(2);              % Estimator outputs both states
Dest = [0; 0];              % No feedthrough
sysK = ss(Aest,Best,Cest,Dest);  % Estimator system

% Loop through 50 noise realizations for average
for count = 1:50
    t = 0:0.01:20;            % Duration of simulation
    d1 = 1*randn(size(t));    % Disturbance to state a1
    d2 = 1*randn(size(t));    % Disturbance to state a2
    n = .1*randn(size(t));    % Noise
    b = zeros(size(d1));      % No actuation

    % Simulate noisy system with disturbance
    [s,tout,a] = lsim(sys,[d1; d2; b],t,[1;0]);

    % Simulate clean system for "truth" baseline
    [sclean,tout,aclean] = lsim(sys,[0*d1; 0*d2; b],t,[1;0]);

    % Simulate Kalman filter to estimate ahat
    sn = s+n';
    [ahat,tout] = lsim(sysK,sn,t,[1; 0]);

    % Compute cost function
    for k=1:size(a,1)
        err = a(k,:)-ahat(k,:);  % Choice: use 'a' or 'aclean'
        Jlong(k) = err*err';
    end
    Jindiv = cumtrapz(t,Jlong);  % Trapezoidal integration
    Jall(count,:) = Jindiv; % Store current realization
end
J = mean(Jall,1);           % Average cost across realizations
```

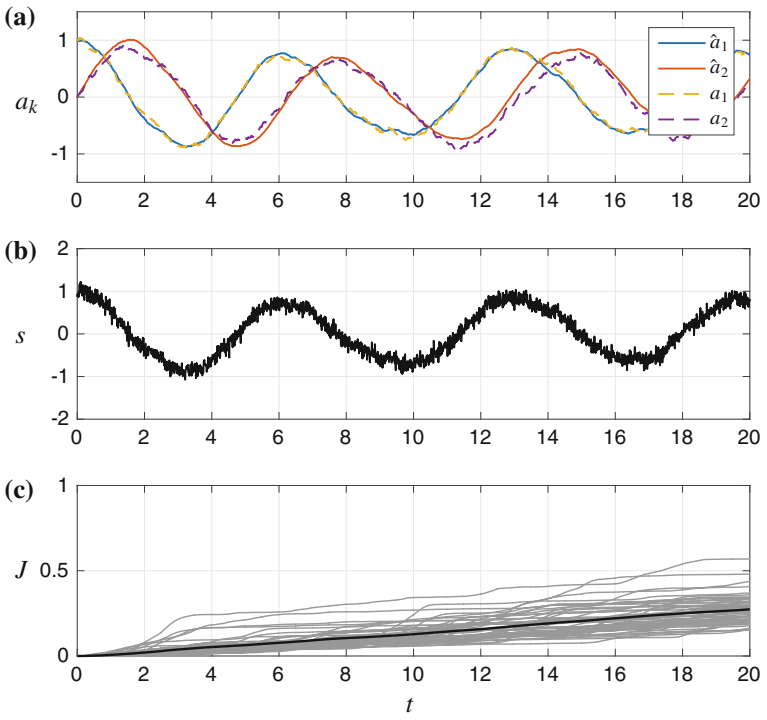


Fig. 4.2 Dynamic full-state estimation of the neutrally stable oscillator system in Eq. (4.6) with a Kalman filter (i.e., LQE). The cost J (3.16) rises because disturbances and noise constantly introduce discrepancies between the estimate $\hat{\mathbf{a}}$ and the true state $\mathbf{a}$. An ensemble average cost function is shown in *black*, and the individual cost functions for fifty instances are shown in *gray*

Table 4.2 Main parameters used for MLC solution of LQE problem. $N_b = 4$ indicates that 4 subtrees are generated for each individual

Parameter	N_i	N_s	N_b	P_r	P_m	P_c	N_p	N_e	Node functions
Value	1000	1	4	0.1	0.4	0.5	7	10	$+, -, \times$

The performance of MLC for state estimation is shown in Fig. 4.5 for the same conditions as used in Fig. 4.2. The state is estimated despite large disturbances and sensor noise. MLC results in a sum-square error that is about twice as large as the optimal Kalman filter solution.

The MLC implementation is given below with the parameters of Table 4.2.

The following code initializes MLC, runs 50 generations, and displays the results.

```
mlc=MLC('MLC_ex_LQE_problem'); %creates the MLC problem
mlc.go(50);                    %runs MLC for 50 generations
mlc.show_best_individual       %displays the results
```

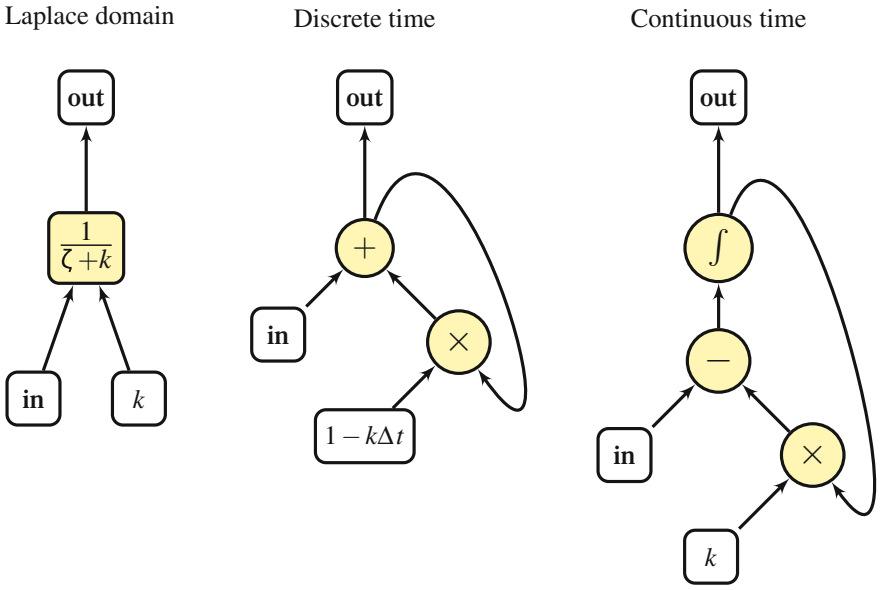


Fig. 4.3 Illustration of a generalized genetic programming transfer function operator in the frequency domain with Laplace variable ζ and filter frequency k. The discrete-time and continuous-time implementations are also shown

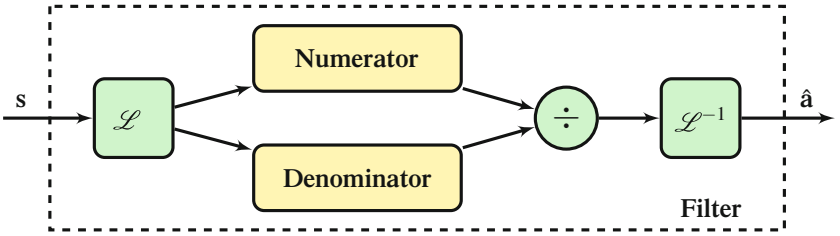


Fig. 4.4 General filter block used in genetic programming for MLC. The inputs **s** are Laplace transformed, and a transfer function is constructed in the frequency domain. This transfer function is achieved by allowing GP to select numerator and denominator polynomials, which are then divided. The output signal is inverse Laplace transformed to bring it back to the time domain

Each filter is the quotient of two polynomials in frequency domain, requiring one tree for the numerator and one tree for the denominator, as in Fig. 4.4. In this example, two filters are identified for each of a_1 and a_2, resulting in 4 polynomials. All individuals are initialized to contain 4 subtrees, and can be written as the following LISP string:

$$(\text{root } (\text{poly}_1)\ (\text{poly}_2)\ (\text{poly}_3)\ (\text{poly}_4))$$

where each 'poly_i' is a LISP polynomial such as '$(\times\ S0\ (+S0\ 2.23))$', where $S0$ represents the argument. In order to only obtain polynomials, $+$, $-$, and $\times$ are the

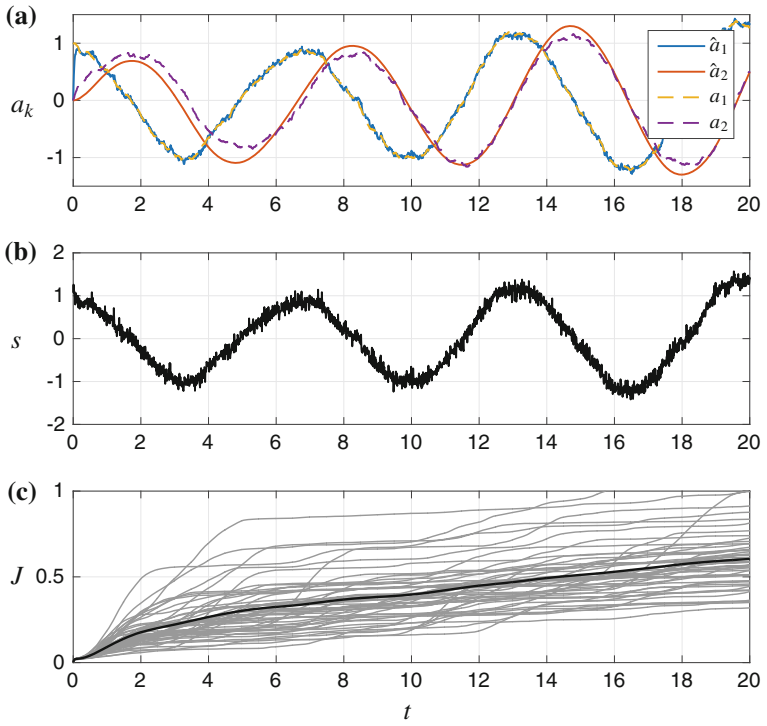


Fig. 4.5 Dynamic full-state estimation of the neutrally stable oscillator system in Eq. (4.6) using genetic programming for MLC. The cost J (3.16) continually rises because disturbances and noise constantly introduce discrepancies between the estimate $\hat{\mathbf{a}}$ and the true state $\mathbf{a}$. An ensemble average cost function is shown in *black*, and the individual cost functions for fifty random instances are shown in *gray*

only operations allowed. The first two subtrees define the denominator and numerator of the first filter. The latter two subtrees describe the same quantities for the second filter.

The evaluation function can be displayed by using the command:

```
open('MLC_evaluator_LQE')
```

The evaluation function is set up in a Simulink® model with the parameters (initial conditions, noise level etc.) and the polynomials for the transfer functions. The frequency-domain filters need the denominator to be of higher order than the numerator. This is why a pre-evaluation function is called at individual initialization, so that proper transfer functions are enforced. The Simulink® implementation of MLC is shown in Figs. 4.6 and 4.7.

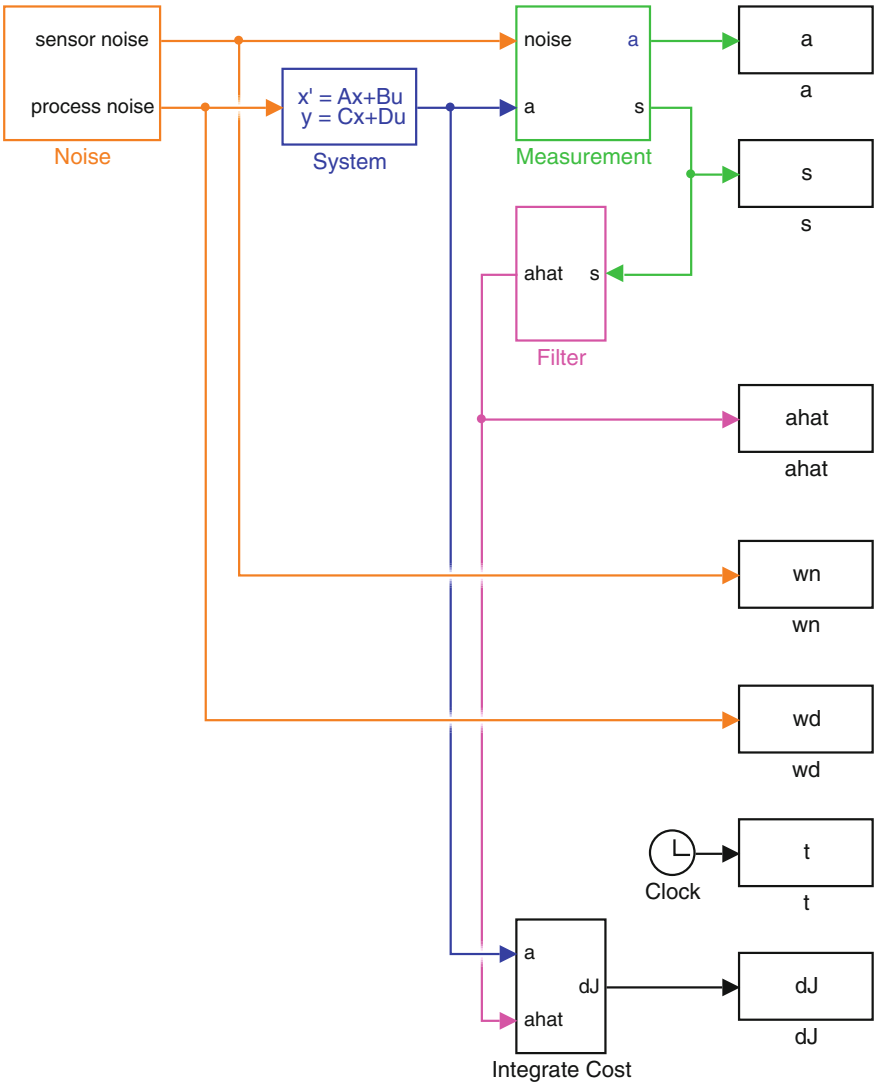


Fig. 4.6 Simulink® model to implement MLC for state estimation based on limited noisy measurements. Individual blocks are shown in Fig. 4.7

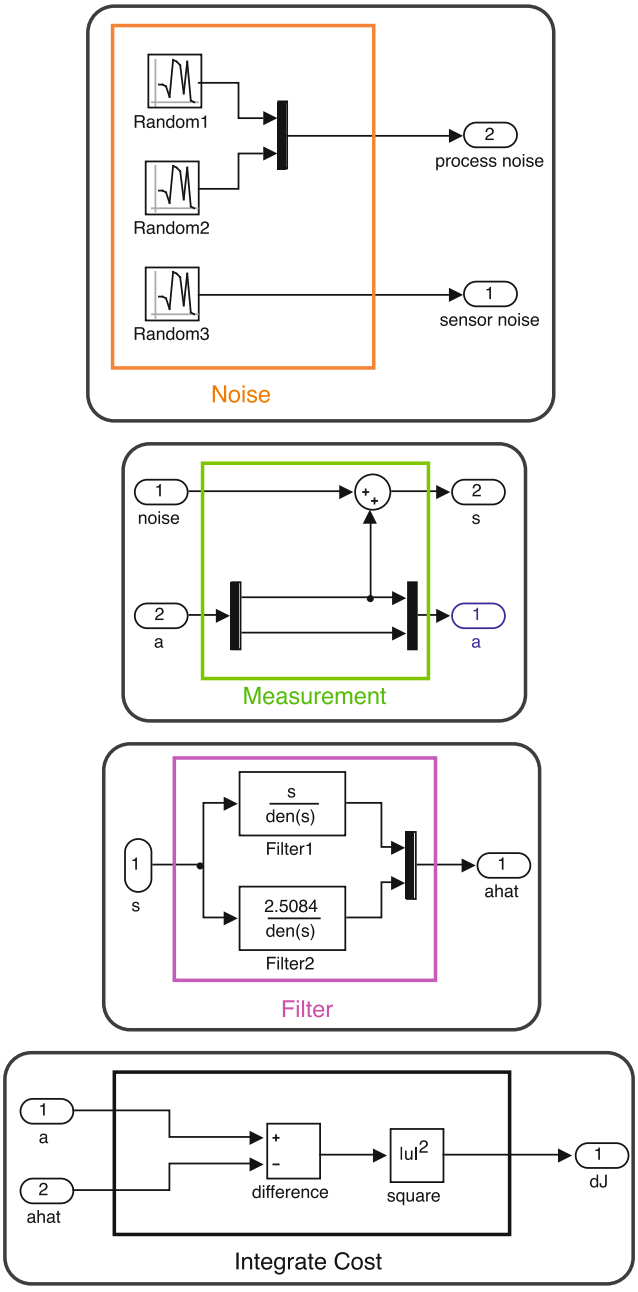


Fig. 4.7 Simulink® blocks for model in Fig. 4.6. This implements MLC for state estimation based on limited noisy measurements

Time-Delay Coordinates Through Spatial Measurements of Convective Flow

Interestingly, it has been shown in convective flow experiments that a finite but large set of spatial sensors have resulted in effective closed-loop control performance, even without dynamic estimation. It is likely that a rake of hot wires is effectively establishing time-delay coordinates, which may be acting as a proxy for a dynamic state estimation. This is a vibrant area of research, with many promising connections to dynamical systems via the Koopman operator [41, 47, 162, 163, 170, 187, 188], although this is beyond the scope of this book.

4.3 Comparison of MLC with LQG for Sensor-Based Feedback

In model-based control design, it is possible to separately design an optimal regulator using LQR and an optimal Kalman filter, and then combine the two to obtain an optimal sensor-based LQG regulator. In practice, we may not have a model of the dynamics, and if the dynamics are unstable it is difficult to run experiments to collect data for system identification of a model. In this section, we demonstrate the ability of MLC to generate a sensor-based stabilizing feedback controller for an unstable system.

Again we consider the unstable system

$$\frac{d}{dt}a_1 = \sigma a_1 - \omega a_2 + w_{d,1} \tag{4.9a}$$

$$\frac{d}{dt}a_2 = \omega a_1 + \sigma a_2 + b + w_{d,2}, \tag{4.9b}$$

and we assume that the sensor only measures the first component:

$$s = a_1 + w_n. \tag{4.10}$$

This corresponds to a linear system with the following system matrices

$$\mathbf{A} = \begin{bmatrix} \sigma & -\omega \\ \omega & \sigma \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \tag{4.11a}$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix}, \qquad \mathbf{D} = \begin{bmatrix} 0 \end{bmatrix}. \tag{4.11b}$$

In this example we choose $\sigma = \omega = 1$, corresponding to an unstable oscillator. The ensemble-averaged cost function from Eq. (3.20) is used.

The result of classic linear-quadratic-Gaussian (LQG) control is shown in Fig. 4.8. Note that the state $\mathbf{a}$ is rapidly stabilized despite a single noisy measurement s.

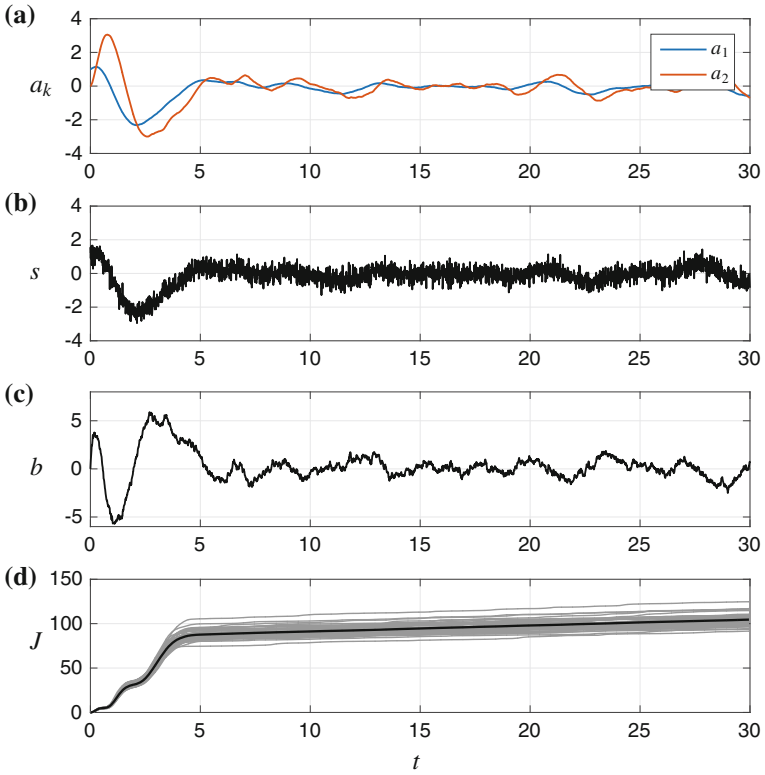


Fig. 4.8 Sensor-based feedback for the unstable oscillator system in Eq. (4.9) using LQG control. The cost J (3.20) continually rises because disturbances and noise constantly push the system away from equilibrium. An ensemble average cost function is shown in *black*, and the individual cost functions for fifty random instances are shown in *gray*

However, because of continual noise and disturbances, the state has small fluctuations about zero for all time. These fluctuations contribute to a slowly increasing cost. The most significant increase in cost J is experienced from $t = 0$ to $t = 5$, where the state is large and there is significant control expenditure required to stabilize the system. The LQG controller represents the optimal sensor-based feedback regulator given a model of the system and of the noise and disturbance covariances.

Figure 4.9 shows the performance of the machine learning control approach on the same unstable system (parameters in Table 4.3). In this example, there is no model of the system dynamics or of the noise and disturbance covariances. Instead, candidate sensor-based control schemes are formulated, tested, evaluated, and evolved to converge on a stabilizing controller. This represents a more challenging and general approach to control design. It is seen that this control also stabilizes the system, although the performance is not optimal as in the LQG case. However, this is reasonable, since MLC is not based on a model, but instead relies on trial-and-error

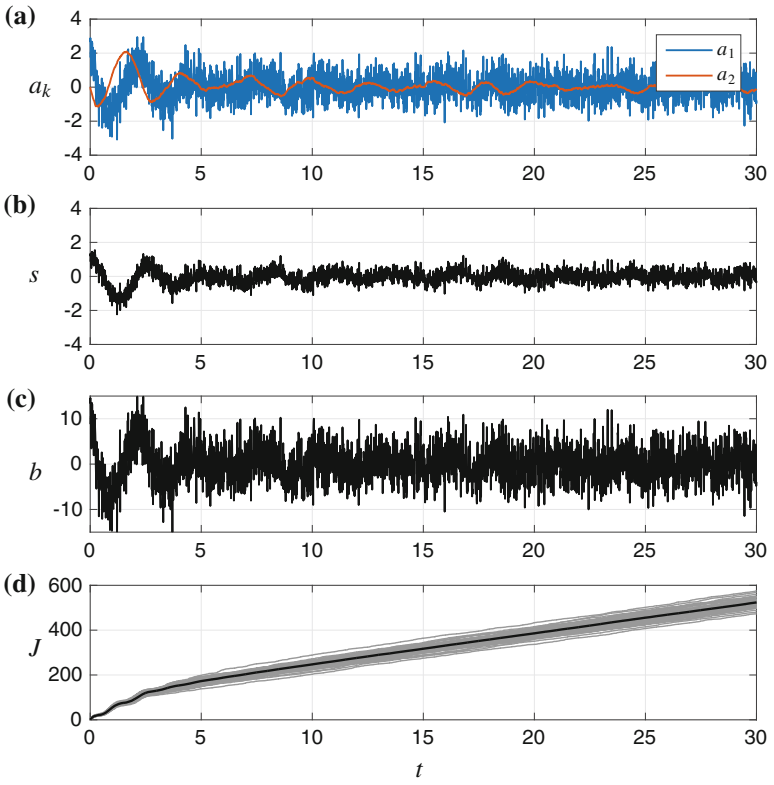


Fig. 4.9 Sensor-based feedback for the unstable oscillator system in Eq. (4.9) using MLC. Parameters and plotted quantities are the same as in Fig. 4.8

Table 4.3 Main parameters used for MLC solution of LQG problem

Parameter	N_i	N_s	N_b	P_r	P_m	P_c	N_p	N_e	Node functions
Value	1000	2	5	0.1	0.4	0.5	7	10	$+, -, \times$

to stabilize the system. Moreover, since the system is unstable originally, it is quite challenging to find stabilizing control expressions. With more optimization and generations, it is likely that MLC will converge to a solution where less noise propagates through the control and into the state, resulting in a lower cost function.

The following code initializes the MLC problem, runs 50 generations, and displays the results.

```
mlc=MLC('MLC_ex_LQG_problem'); % Creates the MLC problem
mlc.go(25);                    % Runs MLC for 25 generations
mlc.show_best_individual       % Displays the results
```

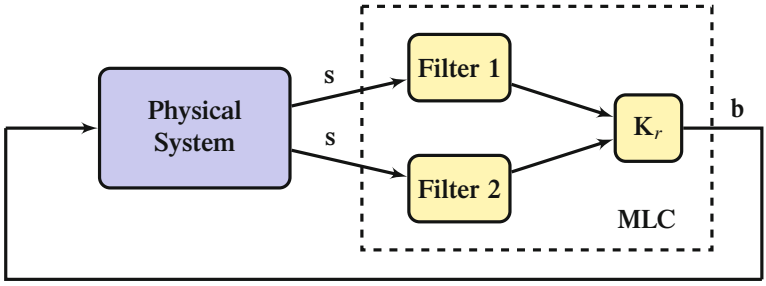


Fig. 4.10 Schematic illustrating the use of filter blocks in genetic programming for MLC on LQG problem. Each filter block is constructed as in Fig. 4.4

In the MLC case, two filters are constructed to estimate the two states a_1 and a_2 from a single noisy sensor (Fig. 4.10). Each filter is composed in the frequency domain as the ratio of a numerator and denominator polynomial, as in Fig. 4.4. Thus, each filter is represented by two expression trees. The controller gain matrix $\mathbf{K}_r$ is also represented as an expression tree, resulting in five trees overall. This can be written as the following LISP string:

$$(\text{root } (\text{poly}_1)\ (\text{poly}_2)\ (\text{poly}_3)\ (\text{poly}_4)\ (\text{poly}_5)),$$

where each 'poly_i' is a LISP polynomial such as '$(\times\ S0\ (+S1\ 2.23))$'. In order to only obtain polynomials, $+$, $-$, and $\times$ are the only operations allowed. For the first four subtrees, all sensors ($S0$ or $S1$) are interpreted as the same variable so that they can be used in polynomials as in the LQE implementation. The fifth subtree uses the two estimated states as its sensor input.

The evaluation function can be displayed by using the command:

```
open('MLC_evaluator_LQG')
```

The evaluation function is set up in a Simulink® model with the parameters (initial conditions, noise level etc.), the polynomials for the Laplace space filters and finally the control law in the controller box. As the Laplace space filters need the denominator to be of higher order than the numerator, for both filters, 75 % of the individuals randomly generated this way cannot describe correctly such a filter. This is why a pre-evaluation function is called at the individual creation, so that only the individuals that meet the requirement that both denominator polynomials are of higher order than their respective numerator are kept.

4.4 Modifications for Small Nonlinearity

Often, linear control can be applied to nonlinear systems. Here, we compare the performance of MLC and LQR when nonlinear terms are added to a linear system:

$$\frac{d}{dt}\mathbf{a} = \mathbf{A}\mathbf{a} + \varepsilon\,\mathbf{F}(\mathbf{a}) + \mathbf{B}\mathbf{b}. \tag{4.12}$$

In this case, the variable ε modulates the strength of the nonlinearity, with $\varepsilon = 0$ corresponding to the linear system in Eq. (3.1). As an example, consider the Hopf system, which adds a cubic nonlinearity to the linear oscillator in Eq. (4.2):

$$\frac{d}{dt}a_1 = \sigma a_1 - \omega a_2 - \varepsilon a_1(a_1^2 + a_2^2) \tag{4.13a}$$

$$\frac{d}{dt}a_2 = \omega a_1 + \sigma a_2 - \varepsilon a_2(a_1^2 + a_2^2) + b. \tag{4.13b}$$

For unstable linear dynamics with $\sigma > 0$, this system has a stable limit cycle with radius $r = \sqrt{a_1^2 + a_2^2} = \sqrt{\sigma/\varepsilon}$. The smaller ε is, the larger the limit cycle radius, and the larger the domain where the linear model is valid. As ε approaches zero, then the limit cycle radius increases to infinity, and the linear system is recovered (Fig. 4.11).

Again, we assume full-state feedback, so that $\mathbf{s} = \mathbf{a}$, and we use the same linear dynamics $\sigma = \omega = 1$, the same LQR controller $b = -\mathbf{K}_r\mathbf{a}$, and the same LQR cost function weights $\mathbf{Q}$ and R from Sect. 4.1.

Figure 4.12 shows the LQR controller response for increasing nonlinearity ε. The system is stabilized for all values, and as ε increases, the nonlinear dynamics actually help the controller, so that the overall cost J decreases. Figure 4.13 shows the LQR performance when ε is negative. In this case, the increasing nonlinearity makes the system more difficult to control, and after a point the LQR controller fails to stabilize the system; these parameter values are marked as diverging solutions.

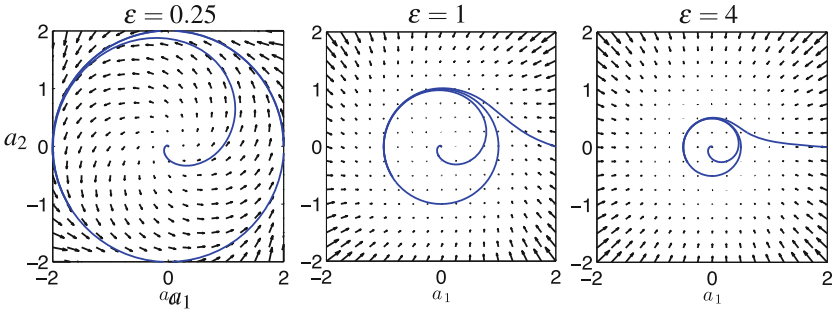


Fig. 4.11 Vector field and trajectories attracting onto limit cycle for the Hopf normal form in Eq. (4.13) with various values of ε

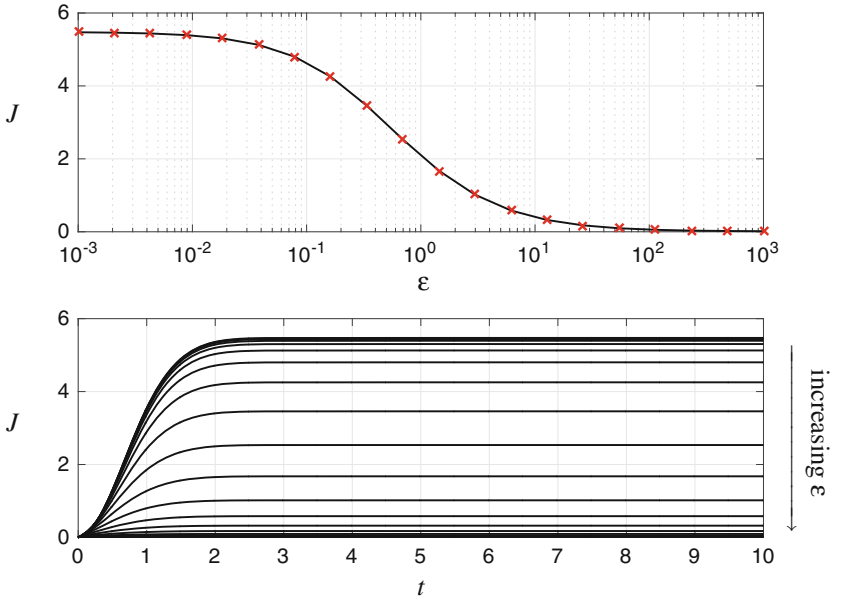


Fig. 4.12 Cost of LQR for Hopf normal form with varying nonlinearity strength ε The initial condition for each case is $a_1 = a_2 = \sqrt{2/2}$

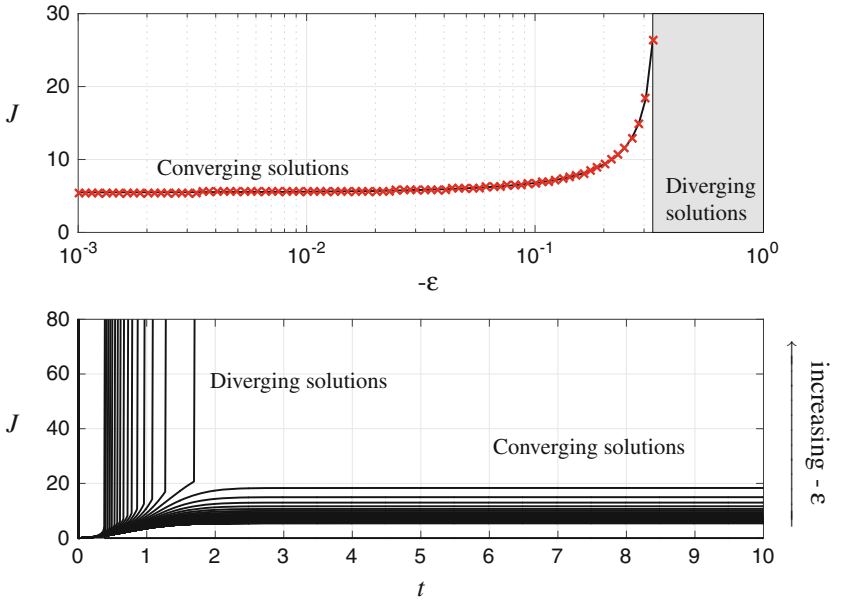


Fig. 4.13 Cost of LQR for unstable Hopf normal form with various magnitude of nonlinearity, $-\varepsilon$. The initial conditions are given by $a_1 = a_2 = \sqrt{2}/2$

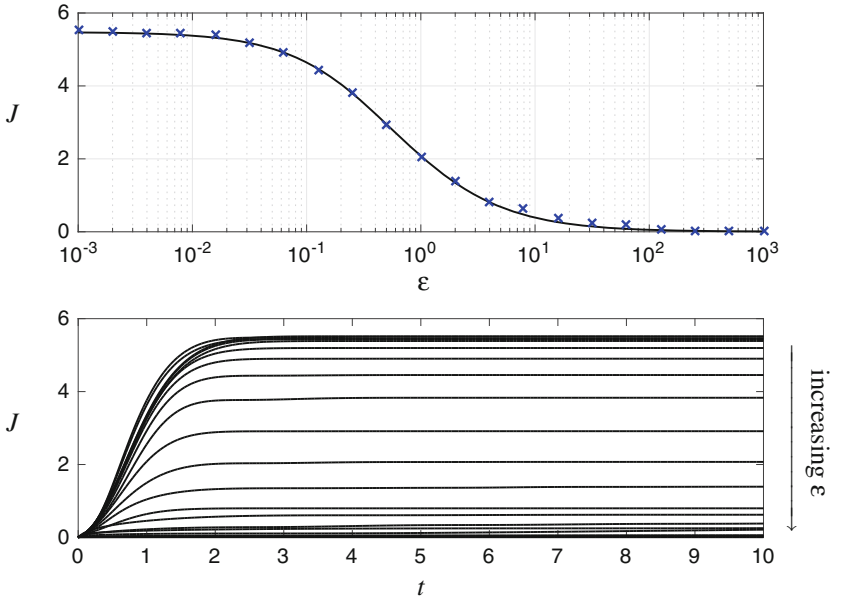


Fig. 4.14 Cost of MLC using $(+, -, *)$ for Hopf normal form with varying nonlinearity strength ε. The initial condition for each case is $a_1 = a_2 = \sqrt{2}/2$. The *blue crosses* indicate MLC results and the *solid black line* shows the corresponding LQR cost

Figures 4.14 and 4.15 show the machine learning control performance on the same system. In the case of positive ε, the performance is quite similar to the LQR solution, whereas in the case of negative ε, MLC performs with lower cost and over a larger range of ε corresponding to stronger nonlinearities.

4.5 Exercises

Exercise 4-1: Consider the same linear system from **Exercise 3-1**:

$$\frac{d}{dt}\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} b. \tag{4.14}$$

(a) Now, use genetic programming to solve for the optimal controller $b = \mathbf{K}(\mathbf{s})$ assuming full state measurements $\mathbf{s} = \mathbf{a}$. This controller should minimize the LQR cost function:

$$J = \int_0^\infty \left[\mathbf{a}^T(\tau)\,\mathbf{Q}\,\mathbf{a}(\tau) + R\,b^2(\tau)\right] d\tau, \quad \mathbf{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1. \tag{4.15}$$

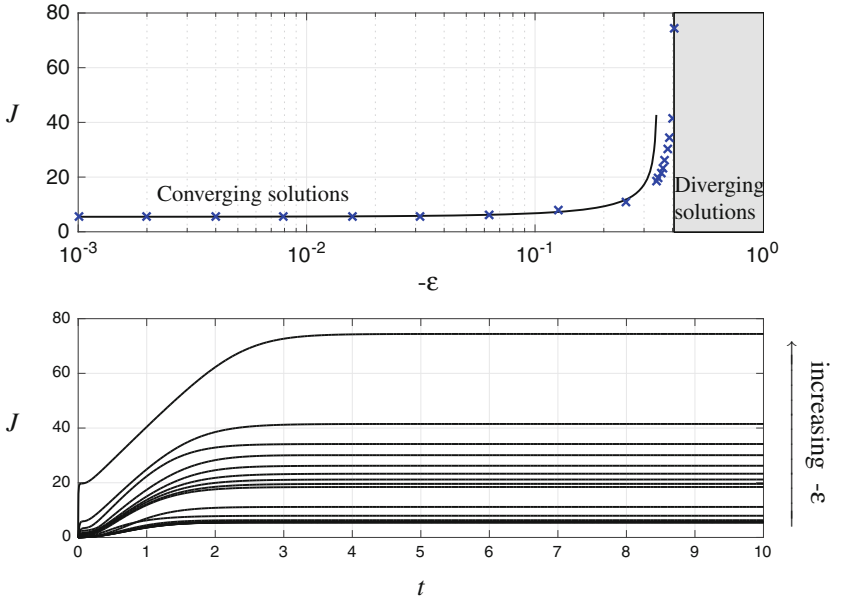


Fig. 4.15 Cost of MLC using $(+, -, *)$ for unstable Hopf normal form with various magnitude of nonlinearity, $-\varepsilon$. The initial conditions are given by $a_1 = a_2 = \sqrt{2}/2$. The *blue crosses* indicate MLC results while the *solid black line* shows the corresponding LQR cost

(b) Check your MLC expression and compare with the optimal LQR solution. Implement a refinement to select genetic programming expressions with good performance but also with the added constraint of a simple expression. This may be added as a penalty in the cost function, or you may alternatively plot a Pareto front of complexity vs performance for numerous candidate high-performance controllers from the generations.

Exercise 4–2: Consider the neutrally stable system:

$$\frac{d}{dt}\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} + \mathbf{B}b + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} w_{d_1} \\ w_{d_2} \end{bmatrix}. \tag{4.16a}$$

$$s = \mathbf{C}\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} + w_n. \tag{4.16b}$$

(a) For the case with no actuation, $\mathbf{B} = \mathbf{0}$, and a measurement of the first state $\mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix}$, develop a genetic programming expression to estimate the state from noisy measurements using full-state training data. Construct an expression to estimate the state with and without the use of filter blocks as discussed in this chapter. How does the static function perform in the presence of noise?

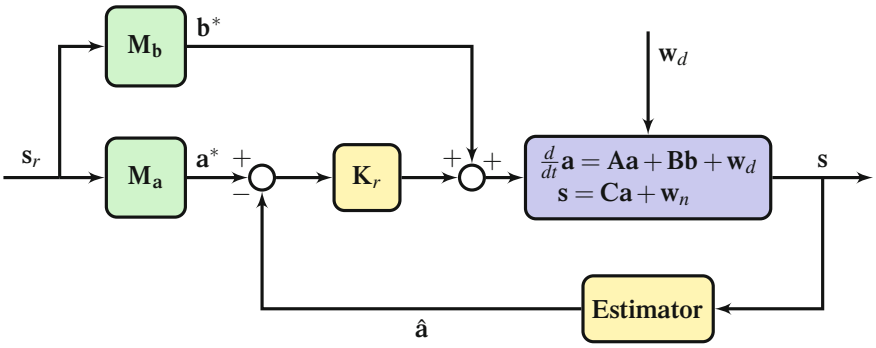


Fig. 4.16 LQG reference tracking schematic for Exercise 4–3

(b) Now consider the case with actuation $\mathbf{B} = \begin{bmatrix} 0 & 1 \end{bmatrix}^T$, and develop a GP expression to estimate the state with forcing. Use the same cost function as in the formulation of a Kalman filter.

(c) Finally, develop an optimal sensor based feedback control law using genetic programming for MLC that optimizes the LQR cost function.

Exercise 4–3: In many cases, it is desirable to track a reference trajectory with feedback control, as opposed to stabilizing a fixed point. For instance, we may implement a control law to design and track a limit cycle.

With a working LQG controller, it is possible to command reference trajectories, as depicted schematically in Fig. 4.16. In the general case, it is necessary to translate the reference sensor signal $\mathbf{s}_r$ into a reference actuation $\mathbf{b}^*$ and reference state $\mathbf{a}^*$ according to the following formula:

$$\begin{bmatrix} \mathbf{0} \\ \mathbf{s}_r \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{a}^* \\ \mathbf{b}^* \end{bmatrix} \implies \begin{bmatrix} \mathbf{a}^* \\ \mathbf{b}^* \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}^\dagger \begin{bmatrix} \mathbf{0} \\ \mathbf{s}_r \end{bmatrix}, \tag{4.17}$$

where the superscript '†' denotes the Moore-Penrose pseudo inverse. In the case of full-state measurements, so that $\mathbf{s} = \mathbf{a}$, then $\mathbf{a}_r = \mathbf{a}^*$, so that $\mathbf{M_a} = \mathbf{I}$ and

$$\mathbf{b}^* = \mathbf{B}^\dagger \mathbf{A} \mathbf{a}_r. \tag{4.18}$$

Thus, $\mathbf{M_b} = \mathbf{B}^\dagger \mathbf{A}$.

Use MLC to design a controller to force a stable linear system into a limit cycle behavior shown in Fig. 4.17. Implement the controller with actuation $\mathbf{B} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. Compare with the full-state LQR controller response in Fig. 4.17.

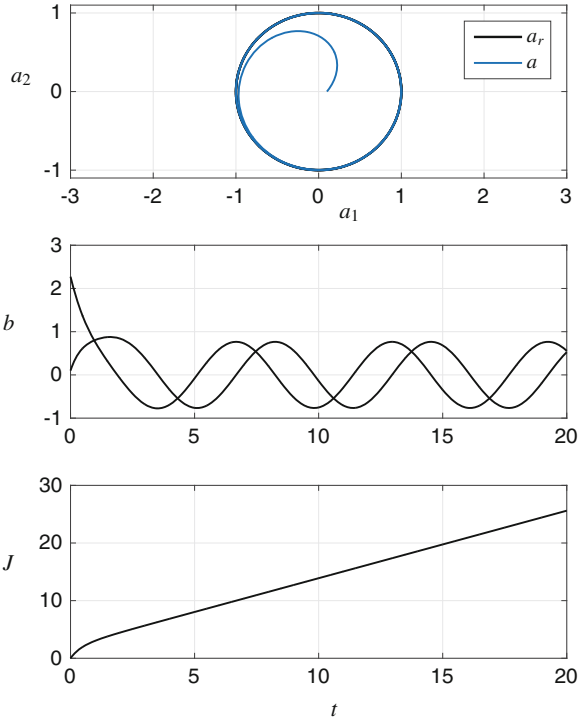


Fig. 4.17 LQR stabilization of a limit cycle with full control authority in Exercise 4–3

4.6 Interview with Professor Shervin Bagheri

Shervin Bagheri is Associate Professor in fluid mechanics at the Royal Institute of Technology (KTH) and the Linné Flow Centre. He has pioneered model-based closed-

loop control with numerical and experimental demonstrations. He is a co-inventor of DMD modes, also known as Koopman modes, one of most used ingredients of reduced-order models. In his research, he focuses on the mathematical foundations and physical principles that enable manipulation of fluid flows. These efforts include both passive and active means to decrease drag, to increase mixing and to enhance lift on bodies. His work was published in the leading fori of our field including Nature Communications, the Philosophical Transactions of the Royal Society London A and Physical Review Letters.

Authors: You are a leader in the field of model-based flow control with landmark contributions to linear control design. What are some of the major trends that you have observed in flow control over the past decade?

Prof. Bagheri: It was only 15 years ago that a systems theoretical approach (such as input-output analysis, state-space systems, controllability, observability etc.) to analyze and control shear flows became an active field on its own. Since then, we have had a decade of proof-of-concept work, focusing on accommodating and testing many of the powerful systems theoretical tools on fluid flows, and using model reduction as enabler to do this. Last years however, things have changed. We know that many control theoretical tools, albeit expensive, can be applied to control fluid flow instabilities both in convective and globally unstable flows at low Reynolds numbers. The use of these tools in nonlinear and turbulent flows is the next step, and in the last years several groups have made progress. For example, by treating the nonlinear terms of the Navier-Stokes equations as an appropriate stochastic forcing. Another emerging branch in flow control is the use of transfer operators (such as the Koopman operator). These methods can via an appropriate nonlinear transformation of the system provide a linear system, where analysis and control tools can be applied, followed by a transformation back. A third example is the use of online adaptive self-learning techniques, such as machine learning. In summary, the major current trends are to deal with nonlinearity and high-dimensionality at the same time in order to move from simple linear 2D systems at low Reynolds numbers towards more complex systems.

Authors: In recent years, you are moving to nonlinear modeling, statistical closures and machine learning methods. Can you sketch the need for the inclusion of nonlinear dynamics and noise in model-based flow control? How much can be gained?

Prof. Bagheri: Moving in this direction is necessary, since in nearly all classical engineering applications the Reynolds number is high and the flow physics is sensitive to the external disturbance environment. It is clear that a flow-control technology based on a systematic approach (in contrast to a trial-and-error approach) that can be used in applications, has to deal with turbulence and robustness.

However, we should also be realistic (and humble) for this task, since for the coming decades our computational capability is limited to academic complex problems, such as low-Reynolds number turbulent flows.

Authors: It is common for practitioners to collect data to characterize a system, develop a model, and then use this model for control. What are some of the challenges and benefits associated with the online learning and adaptation of controllers from data?

Prof. Bagheri: Indeed, data-driven methods are becoming increasingly important, as large-scale high-performance computations have now taken its rightful place in the community. When it comes to control of high-dimensional nonlinear chaotic systems, in my opinion, an attractive approach is adaptive algorithms that are able to adjust online to new dynamics in uncertain conditions. One of the challenges we have encountered when using adaptive algorithms is that although they may be fast enough to account for slow changes in conditions (e.g. variation in the Reynolds number), they are often not sufficiently quick learners to account for changes in the internal changes in the dynamics (e.g. emergence of new length scales during the transition process).

Authors: In the coming decades, what do you envision as the academically most rewarding grand-challenge problems of feedback control in fluid dynamics? You work not only on a model-based control logic but also on model-free bio-inspired actuators. Which evolutions do you foresee in experimental flow control on a hardware and a theoretical level?

Prof. Bagheri: The grand challenge is to efficiently and robustly control turbulence for Reynolds numbers that are relevant for everyday applications. Within the next decade, we will be able to reduce turbulent skin friction drag with 30 % using actuation/sensing at the wall at moderate Reynolds numbers. It will probably take another decade or two, to devise both efficient and robust controllers for high-Reynolds number turbulent flows, where the contribution to skin-friction is also significant from large scale structures. In order to achieve these goals we need a multi-disciplinary approach, where advances in fluid mechanics, material science and surface chemistry are combined with applied mathematics, algorithms and computer science.

For example, we are now looking into how soft, porous, lubricated, multi-scale hierarchical materials possibly treated chemically can be used to manipulate an overlying fluid. Although, mimicking biological surface coatings such as shark skin and lotus leaf has proven useful, I believe that active control techniques can provide the right guidance for using innovative surface materials for flow control.

Authors: We look forward to your next breakthroughs in flow control and thank you for this interview!

Prof. Bagheri: Thank you. It was a pleasure.

Chapter 5
Taming Nonlinear Dynamics with MLC

Prediction is very difficult, especially about the future.
Niels Bohr

Frequency crosstalk is a ubiquitous phenomenon of turbulence and is of pivotal importance in control. In the *normal turbulence cascade*, the coherent structures feed increasingly smaller scales corresponding to increasingly larger frequencies with energy via the transfer term. In the *inverse cascade*, the merging of coherent structures yield increasingly larger scales or lower frequencies. All frequencies change the base flow, i.e. low frequencies, via the Reynolds stress. Thus, interacting frequencies range from the zero frequency corresponding to the mean flow to large frequencies corresponding to the Kolmogorov scale.

Control design may exploit this frequency crosstalk. Numerous experiments have demonstrated how high-frequency forcing can stabilize the fluid flow. Examples include jets [236], mixing layers [206], wakes [264], car models [21], and the flow over a backward-facing step [272]. Low frequency forcing can have a similar effect [5, 210]. In both cases, the coherent structures at a characteristic frequency are mitigated by a different imposed frequency. In other words, frequency crosstalk is a control enabler!

In this chapter, we present a generalized mean-field model as arguably the most simple dynamical model for frequency crosstalk between unforced and forced frequency components (Sect. 5.1). The control goal is to stabilize the unstable natural frequency. While the control based on linearized dynamics (Chap. 3) is shown to fail, MLC detects and exploits the frequency crosstalk mechanism in an unsupervised manner (Sect. 5.2). In Sect. 5.3, the derivation of the investigated model is sketched. This derivation contains the underlying approximations for the nonlinear control approaches, against which MLC is benchmarked in Sect. 5.4. The last two sections are analytical supplements for improved understanding of the model and the MLC control. These sections require background in methods of nonlinear

T. Duriez et al., *Machine Learning Control – Taming Nonlinear Dynamics and Turbulence*, Fluid Mechanics and Its Applications 116,
DOI 10.1007/978-3-319-40624-4_5

oscillation [148] and nonlinear dynamics [125] and may be skipped during the first reading. Sect. 5.5 contains exercises for MLC control. A suggested reading (Sect. 5.6) and an interview (Sect. 5.7) with Professor Mark Glauser, a pioneer in nonlinear modeling and feedback turbulence control, conclude this chapter.

5.1 Generalized Mean-Field System

In this section, we review a generalized mean-field model which explains how low- or high-frequency forcing stabilizes a self-amplified natural instability. The model has been used to explain the stabilizing effect of low-frequency forcing of a wake [210] and a resulting control design [6]. Similarly, the model has been applied to high-frequency forcing of a high-lift airfoil [176] and model-based control design [178]. Another application is an actuated swirling jet [201].

We refer to Sect. 5.3 for the derivation of the generalized mean-field model. The result is a four-dimensional model which describes the evolution of the mode amplitudes a_i, $i = 1, \ldots, 4$ of a Galerkin expansion for the fluctuation. Here, a_1, a_2, is associated with the cosine and sine mode of natural vortex shedding and a_3, a_4 describe analogous quantities for the periodically forced coherent structures. The system of ordinary differential equation reads

$$\frac{da_1}{dt} = \sigma_\bullet a_1 - \omega_\bullet a_2 \tag{5.1a}$$

$$\frac{da_2}{dt} = \sigma_\bullet a_2 + \omega_\bullet a_1 \tag{5.1b}$$

$$\frac{da_3}{dt} = \sigma_\circ a_3 - \omega_\circ a_4 \tag{5.1c}$$

$$\frac{da_4}{dt} = \sigma_\circ a_4 + \omega_\circ a_3 + g\, b \tag{5.1d}$$

$$\sigma_\bullet = \sigma_{\bullet\star} - \beta_{\bullet\bullet} r_\bullet^2 - \beta_{\bullet\circ} r_\circ^2 \tag{5.1e}$$

$$\omega_\bullet = \omega_{\bullet\star} + \gamma_{\bullet\bullet} r_\bullet^2 + \gamma_{\bullet\circ} r_\circ^2 \tag{5.1f}$$

$$\sigma_\circ = \sigma_{\circ\star} - \beta_{\circ\bullet} r_\bullet^2 - \beta_{\circ\circ} r_\circ^2 \tag{5.1g}$$

$$\omega_\circ = \omega_{\circ\star} + \gamma_{\circ\bullet} r_\bullet^2 + \gamma_{\circ\circ} r_\circ^2. \tag{5.1h}$$

The symbols are explained in Table 5.1.

The nonlinearity of Eq. (5.1) has two important effects. First, without forcing, $b \equiv 0$, the second oscillator vanishes, $a_3 = a_4 = 0$. Thus, Eq. (5.1a), (5.1b), (5.1e), and (5.1f) represent a Landau oscillator with linear oscillatory instability ($\sigma_{\bullet\star} > 0$) and a cubic damping ($\beta_{\bullet\bullet} > 0$). In other words, the first oscillator has a globally stable limit cycle as discussed in Sect. 4.4 for $b \equiv 0$. Second, with forcing at the eigenfrequency of the stable oscillator, i.e. $b = B\cos(\omega_{\circ\star} t)$, the amplitude of the second oscillator $r_\circ$ grows in proportion to the forcing amplitude B, as the nonlinear terms of Eq. (5.1c) and (5.1d) are assumed to vanish (see Table 5.1). The stabilizing

Table 5.1 Symbols of Sect. 5.1

Quantities related to the amplitude of the unstable oscillator (a_1, a_2)	
$r_\bullet$	Amplitude of the first oscillator
$\sigma_\bullet$	Growth rate
$\sigma_{\bullet\star} = 0.1$	Initial growth rate near the fixed point $r_\bullet = 0$
$\beta_{\bullet\bullet} = 1$	Parameter for growth-rate change of $\sigma_\bullet$ due to $r_\bullet$
$\beta_{\bullet\circ} = 1$	Parameter for growth-rate change of $\sigma_\bullet$ due to $r_\circ$
Quantities related to the phase of the unstable oscillator (a_1, a_2)	
$\phi_\bullet$	Phase of the first oscillator
$\omega_\bullet$	Frequency (analog to $r_\bullet$ for the amplitude)
$\omega_{\bullet\star} = 1,$	Initial frequency (analog to $\sigma_{\bullet\star}$ for the amplitude)
$\gamma_{\bullet\bullet} = 0,$	Parameter for frequency change due to $r_\bullet$ (analog to $\beta_{\bullet\bullet}$)
$\gamma_{\bullet\circ} = 0,$	Parameter for frequency change due to $r_\circ$ (analog to $\beta_{\bullet\circ}$)
Quantities related to the amplitude of the stable oscillator (a_3, a_4)	
$r_\circ$	Amplitude of the second oscillator
$\sigma_\circ$	Growth rate
$\sigma_{\circ\star} = -0.1$	Initial growth rate near the fixed point $r_\circ = 0$
$\beta_{\circ\bullet} = 0$	Parameter for growth-rate change of $\sigma_\circ$ due to $r_\bullet$
$\beta_{\circ\circ} = 0$	Parameter for growthrate change of $\sigma_\circ$ due to $r_\circ$
$g = 1$	Gain of control command
Quantities related to the phase of the stable oscillator (a_3, a_4)	
$\phi_\circ$	Phase of the second oscillatory
$\omega_\circ$	Frequency (analog to $r_\circ$ for the amplitude)
$\omega_{\circ\star} = 10,$	Initial frequency (analog to $\sigma_{\circ\star}$ for the amplitude)
$\gamma_{\circ\bullet} = 0,$	Parameter for frequency change due to $r_\bullet$ (analog to $\beta_{\circ\bullet}$)
$\gamma_{\circ\circ} = 0,$	Parameter for frequency change due to $r_\circ$ (analog to $\beta_{\circ\circ}$)

The equations indicate the numerical values employed for the control problem

effect of the second oscillator on the first one requires $\beta_{\bullet\circ} > 0$. Thus, fluctuation of the second oscillator reduces the growth rate of the first oscillator. The minimum amplitude $r_\circ$ for complete stabilization $r_\bullet = 0$ is $r_\circ = \sqrt{\sigma_{\bullet\star}/\beta_{\bullet\circ}}$, as can be derived from $\sigma_\bullet = 0$ in Eq. (5.1e). Physically, the forcing changes the base flow such that the production of the unstable coherent structures is reduced below the dissipative term.

In the following, Eq. (5.1) is simplified for the purpose of defining a control problem in which a high-frequency stable oscillator stabilizes a self-amplified amplitude-limited one. All coefficients which are not needed to illustrate the frequency crosstalk are set to zero, e.g. the Landau coefficients for the nonlinear terms of the second stable oscillator ($\beta_{\circ\circ} = \beta_{\circ\bullet} = \gamma_{\circ\circ} = \gamma_{\circ\bullet} = 0$) and for the frequency variation of the second oscillator ($\gamma_{\bullet\bullet} = \gamma_{\bullet\circ} = 0$). The small growth or decay rates of the oscillators are set to ± 0.1, i.e. $\sigma_{\bullet\star} = 0.1$ and $\sigma_{\circ\star} = -0.1$. The large frequency of the second oscillator is set to $\omega_{\circ\star} = 10$. The remaining quantities, i.e. the frequency and Landau

coefficients of the first oscillator and the gain of the forcing term, are set to unity, $\omega_{\bullet\star} = \beta_{\bullet\bullet} = \beta_{\bullet\circ} = g = 1$. The parameter values are listed in Table 5.1.

The resulting system reads:

$$\frac{da_1}{dt} = \sigma\, a_1 - a_2 \tag{5.2a}$$

$$\frac{da_2}{dt} = \sigma\, a_2 + a_1 \tag{5.2b}$$

$$\frac{da_3}{dt} = -0.1\, a_3 - 10\, a_4 \tag{5.2c}$$

$$\frac{da_4}{dt} = -0.1\, a_4 + 10\, a_3 + b \tag{5.2d}$$

$$\sigma = 0.1 - a_1^2 - a_2^2 - a_3^2 - a_4^2. \tag{5.2e}$$

For the initial condition at $t = 0$ we choose a point close to the unstable fixed point,

$$\mathbf{a}(0) = [a_1, a_2, a_3, a_4]^T(0) = [0.01, 0, 0, 0]^T. \tag{5.3}$$

The superscript 'T' denotes the transpose of the row vector.

The transient and actuated dynamics of Eq. (5.2) are illustrated in Fig. 5.1. The initial period shows an unforced transient towards the limit cycle. Then, actuation excites the stable oscillator which mitigates the first one via the growth rate.

The cost function to be minimized contains the average fluctuation level of the unstable oscillator

$$J_a = \overline{a_1^2 + a_2^2} \tag{5.4}$$

penalized by the actuation cost

$$J_b = \overline{b^2} \tag{5.5}$$

with penalization parameter γ,

$$J := J_a + \gamma\, J_b = \overline{a_1^2 + a_2^2} + \gamma\, \overline{b^2} \stackrel{!}{=} \min. \tag{5.6}$$

The overbar denotes numerically an average over the time window $[20\pi, 220\pi]$, i.e. the average of a time-dependent function $f(t)$ reads

$$\overline{f(t)} := \frac{1}{200\pi} \int_{20\pi}^{220\pi} dt\, f(t).$$

The range of integration starts at $t_0 = 20\pi$, corresponding to 10 periods of the unstable oscillator, so that transients have time to die out. The upper integration bound is $t_1 = 220\pi$ to include 100 periods, which is sufficient for a statistical average. It should be noted that MLC requires only an approximately accurate ordering of the

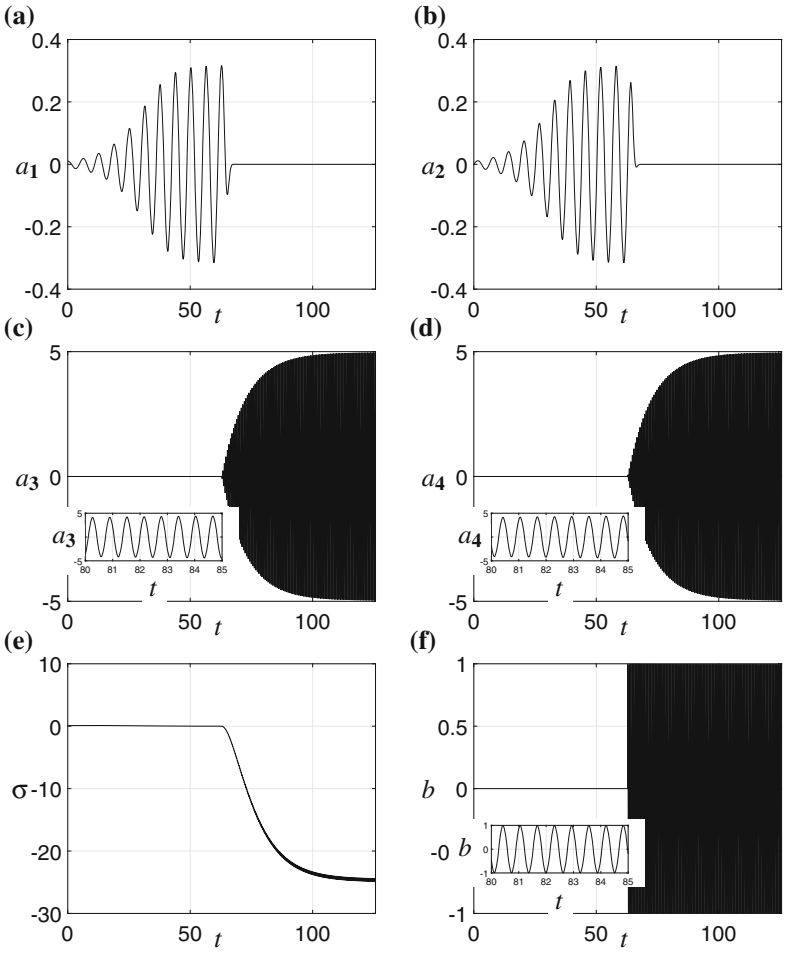


Fig. 5.1 Dynamics of the generalized mean-field model (5.2) with the initial condition (5.3). Periodic forcing $b = \cos(10t)$ is applied at $t \geq 20\pi$

costs associated with the considered control laws. Hence, we refrain from using, say, 1000 periods to obtain slightly more accurate values.

A canonical strategy for a stabilizing control employs a linearization of the evolution equation around the fixed point. The generalized mean-field model (5.2) has the fixed point $a_1 = a_2 = a_3 = a_4 = 0$. Linearizing around that point yields two uncoupled oscillators

$$\frac{da_1}{dt} = 0.1\, a_1 - a_2 \tag{5.7a}$$

$$\frac{da_2}{dt} = 0.1\, a_2 + a_1 \tag{5.7b}$$

$$\frac{da_3}{dt} = -0.1\, a_3 - 10\, a_4 \tag{5.7c}$$

$$\frac{da_4}{dt} = -0.1\, a_4 + 10\, a_3 + b. \tag{5.7d}$$

The amplitude of the first oscillator grows without bound while the second oscillator converges to its fixed point $a_3 = a_4 = 0$ without forcing. Evidently, the unstable oscillator cannot be stabilized by arbitrary actuation commands b, because the linearization has removed the pivotal nonlinear frequency crosstalk encoded in Eq. (5.1e). In terms of control theory from Chap. 3, the linearized system is not controllable.

5.2 Machine Learning Control

In this section, the control problem described in Sect. 5.1 is solved with MLC. Sect. 5.2.1 specifies the mathematical problem to be solved. In Sect. 5.2.2, the choice of parameters of MLC is outlined and motivated. The results are provided in Sect. 5.2.3.

5.2.1 Formulation of the Control Problem

The control problem for MLC consists of minimizing the cost function J (5.6) for the simplified generalized mean-field model (5.2) under initial condition (5.3). The penalization parameter is chosen to be $\gamma = 0.01$. The system is integrated numerically for a time interval of $[20\pi, 220\pi]$, which allows for an unrecorded 10 period transient and evaluates 100 periods of the unstable oscillator deemed sufficient for representative statistics. At this point, we know that the linearized dynamics will not reveal the enabling frequency crosstalk mechanism and that an open-loop periodic forcing can completely stabilize the first oscillator. We search for an autonomous full-state feedback law minimizing the cost function,

$$b = K(\mathbf{a}) = K(a_1, a_2, a_3, a_4). \tag{5.8}$$

Thus, we explore all potential non-linear feedback mechanisms stabilizing the first oscillator. The optimization problem formally reads

$$K_{\text{opt}}(\mathbf{a}) = \underset{K(\mathbf{a})}{\operatorname{argmin}}\, J\left[K(\mathbf{a})\right] \Big|_{\substack{\text{subject to Eq. (5.2),} \\ \text{and initial condition (5.3)}}} \tag{5.9}$$

Table 5.2 MLC parameters used for the control of the generalized mean-field model (5.2)

Parameter	N_i	P_r	P_m	P_c	N_p	N_e
Value	1000	0.1	0.3	0.6	7	1
Operations	$+, -, \times, /$, sin, exp, log, tanh					

where $K_{\text{opt}}(\mathbf{a})$ denotes the optimal control law which minimizes the cost. The dependency of the solution on the initial condition might be considered problematic and should normally be avoided. However, we recall that actuation will be evaluated after a long transient time. Secondly, the results have been found to hardly change after incorporating an ensemble of initial conditions in the regression problem (5.9).

5.2.2 MLC Parameters

The function space of MLC is explored by using a set of elementary operations $(+, -, \times, /)$ and transcendental (exp, sin, ln and tanh) functions. The functions are 'protected' to allow them to take arbitrary arguments in $\mathbb{R}$ (e.g. a thresholding is achieved on denominators in divisions to avoid division by zero). Additionally, the actuation command is limited to the range $[-1\,,\ 1]$ to emulate an experimental amplitude-bounded actuator. Up to $N_g = 50$ generations comprising $N_i = 1000$ individuals are processed. The tournament size is $N_p = 7$, elitism is set to $n_e = 1$, the probabilities of replication, crossover and mutation are $P_r = 0.1$, $P_c = 0.6$ and $P_m = 0.3$ respectively (see Table 5.2).

5.2.3 MLC Results

Figure 5.2 displays the MLC learning process associated with the optimization problem (5.9). The enforced ordering of the individuals with respect to their cost

$$J_1^j \le J_2^j \le \ldots \le J_{N_i}^j \qquad j = 1, \ldots, N_g$$

is evidenced in the jth column by increasing J-value with increasing i. The learning of increasingly better control laws with increasing generation number j can be seen from decreasing J values towards the right. In particular, elitism enforces that the cost of the best individual cannot increase,

$$J_1^1 \ge J_1^2 \ge \ldots \ge J_1^{N_g}.$$

The 'spectrogram' of all computed J_i^j is visualized in Fig. 5.3. Each generation j is seen to consist of a large range of cost values.

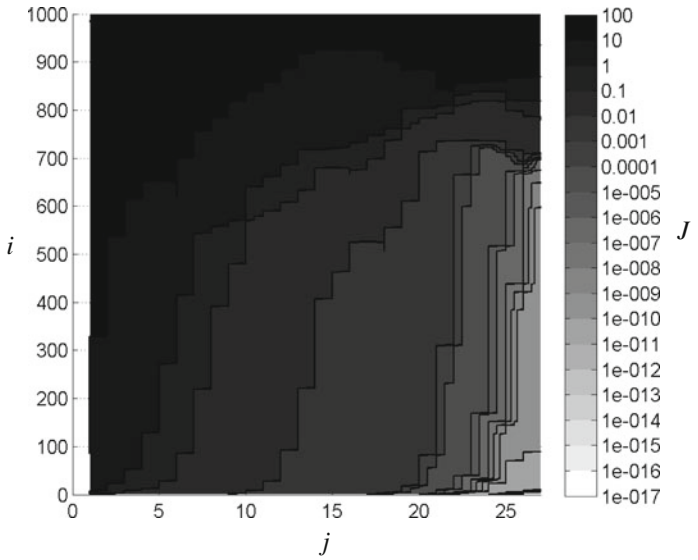


Fig. 5.2 MLC learning process for the control problem (5.9) with the generalized mean-field model. The abscissa displays the generation number j. The ordinate refers to the individual i. The background shows the value of the cost function J_i^j for each tested individual

The best individual $i = 1$ in the last generation $j = N_g$ defines solution of regression problem for the MLC feedback law (5.9). The corresponding actuated dynamics is depicted in Fig. 5.4. Intriguingly, MLC does not emulate periodic forcing with regular 'soft' excitation of the second oscillator. Instead, it chooses to stabilize the first oscillator by occasional hard 'kicks', i.e. by strongly exciting the second oscillator and decreasing the growth rate to low negative values.

The instance of these kicks is best appreciated in a logarithmic plot of the fluctuation levels of both oscillators (Fig. 5.5). The kicks occur at fluctuation levels of roughly 10^{-5} and last until this level has been decreased to around 10^{-40} or lower.

The MLC law solving the regression problem (5.9) is visualized in Fig. 5.6 as binary tree. The formula can be expressed as follows:

$$b = K_1(a_4) \times K_2(a_1, a_2, a_3, a_4) \tag{5.10}$$

with

$$K_1(a_4) = 5.475 \times a_4$$

and

$$K_2(a_1, a_2, a_3, a_4) = \frac{\left(\frac{(\frac{a_4}{4.245}) \times (\sin(\tanh(\frac{a_4}{-5.987})))}{\tanh(a_2)} + a_2\right)}{\cos(3.053)}$$

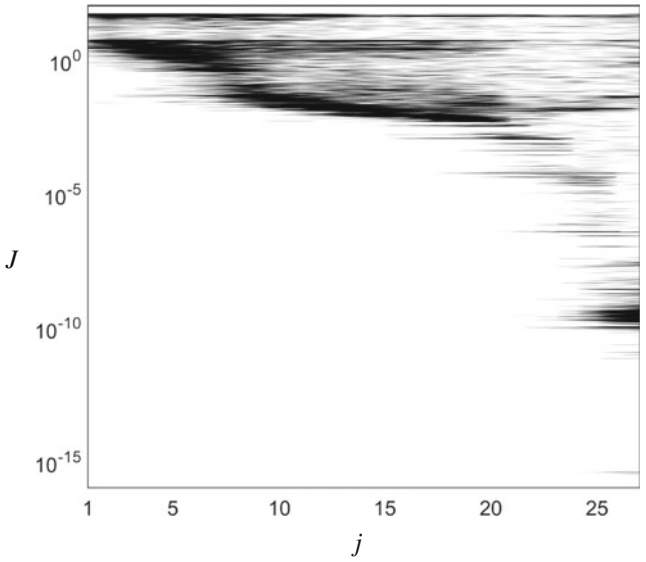


Fig. 5.3 Same MLC run as Fig. 5.2. Now, the J_i^j values, $i = 1, \ldots, N_i$, for each generation j are indicated on a logarithmic ordinate scale

$$\times \left(\frac{\frac{8.965}{a_1}}{\tanh(a_2)} + \frac{\frac{a_2}{\cos(3.053)}}{\frac{a_4+\cos(-8.208)}{\log(a_3)}} \right) \times \left(\frac{\left(\frac{\frac{a_1}{a_1}}{\frac{a_1}{a_4}} \right)}{\frac{\left(\frac{(\sin(a_4))\times(\tanh((4.640)\times(a_2)))}{-6.912-(a_4)} \right) \times \left(\frac{a_1}{(a_2)\times(a_4)} \right)}{\frac{a_2}{a_1}+\tanh(a_2)}} \right) \times \left(\frac{a_1}{\frac{a_2}{a_4}} \right) \times \frac{}{-7.092}.$$

The function $K_1(a_4)$ describes a phasor control that destabilizes the stable oscillator. The function $K_2(a_1, a_2, a_3, a_4)$ acts as a gain dominated by the energy of the unstable oscillator. This control law cannot be derived from a linearized model of the system. Moreover, (slightly) less energy is used as compared to the best periodic excitation.

A revealing illustration of the MLC control law in a four-dimensional space is a challenge. In the following, we propose a generic strategy. Let $p(\mathbf{a})$ be the probability density associated with the MLC-actuated dynamics (5.2). Thus, the expectation value of the actuation command $b = K(a_1, a_2, a_3, a_4)$ at given values a_1 and a_2 can be formally defined:

$$\langle b \rangle_\bullet = \langle K \rangle_\bullet = \int \int p(a_1, a_2, a_3, a_4)\, K(a_1, a_2, a_3, a_4)\, da_3\, da_4. \tag{5.11}$$

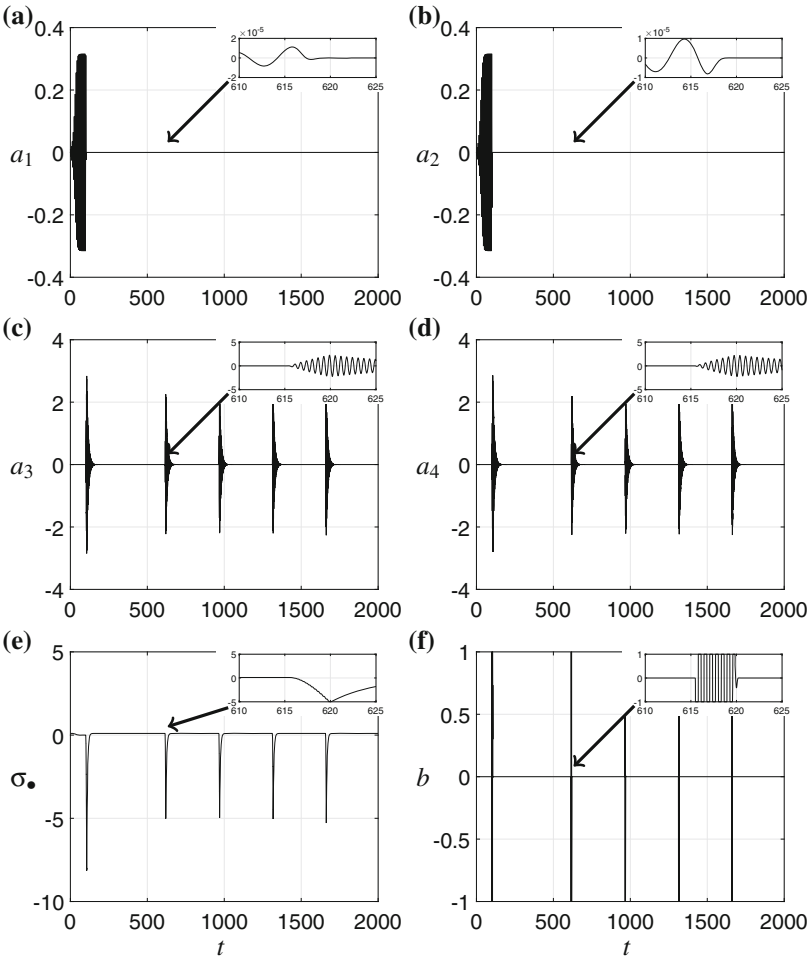


Fig. 5.4 Dynamics of the MLC-controlled generalized mean-field model (5.2)

The analogous quantity for the second oscillator reads

$$\langle b \rangle_\circ = \langle K \rangle_\circ = \int\int p(a_1, a_2, a_3, a_4)\, K(a_1, a_2, a_3, a_4)\, da_1\, da_2. \tag{5.12}$$

Figure 5.7 depicts this expectation value in the a_1-a_2 plane. More precisely, we employ polar coordinates $a_1 = r_\bullet \cos\phi_\bullet$, $a_2 = r_\bullet \sin\phi_\bullet$, and plot the radius on a logarithmic scale. Thus, the phase associated with the expectation value of the control command b can be resolved even at small fluctuation values. Expectedly, no strong phase preference $\phi_\bullet$ for control action is apparent. Each period of the unstable oscillator is associated with 20 sign changes of $\langle b \rangle_\bullet$, because the actuation drives the

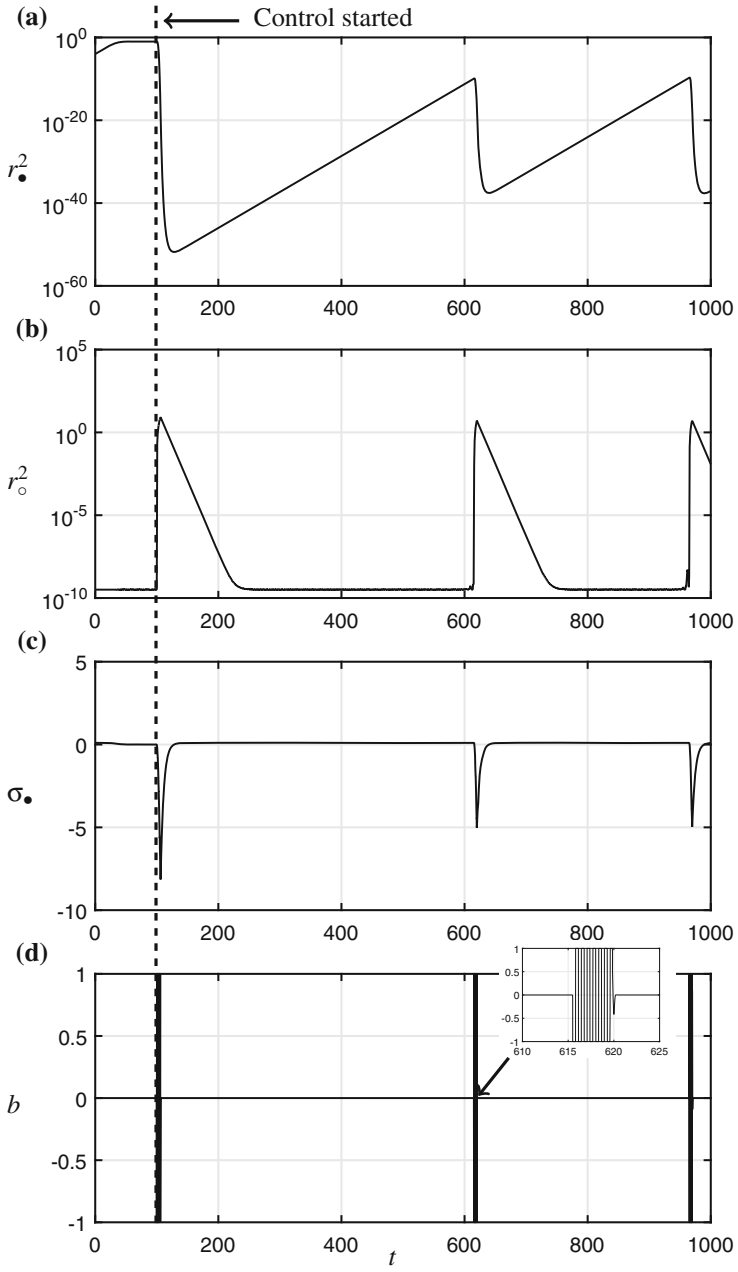


Fig. 5.5 Energy levels of the oscillators displayed in Fig. 5.4. When the energy contained in the first oscillator (*top*) is larger than 10^{-10}, the control (*bottom*) excites the second oscillator, and its energy grows to roughly 3 so that σ reaches approximately -6 ± 1. This results in a fast decay of the energy in the first oscillator after which the control goes back to a "stand-by" mode

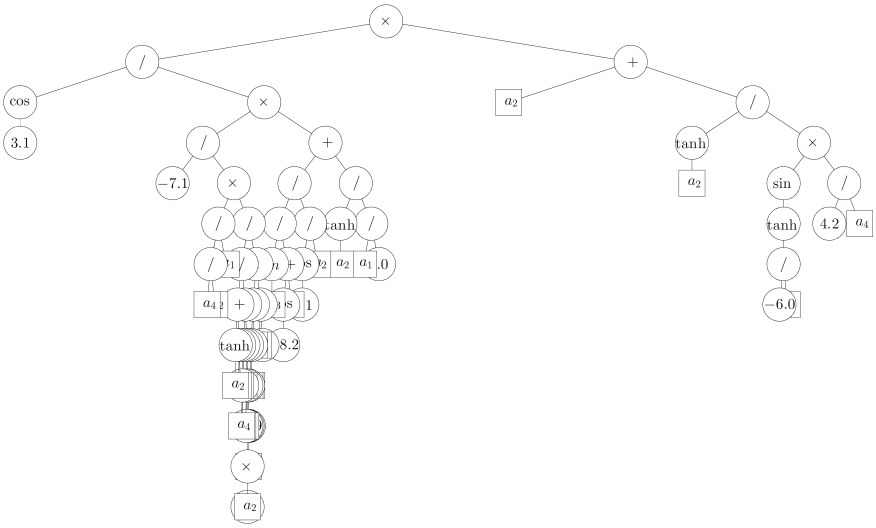


Fig. 5.6 Tree representation of the MLC law solving the regression problem (5.9) and used in Figs. 5.4 and 5.5

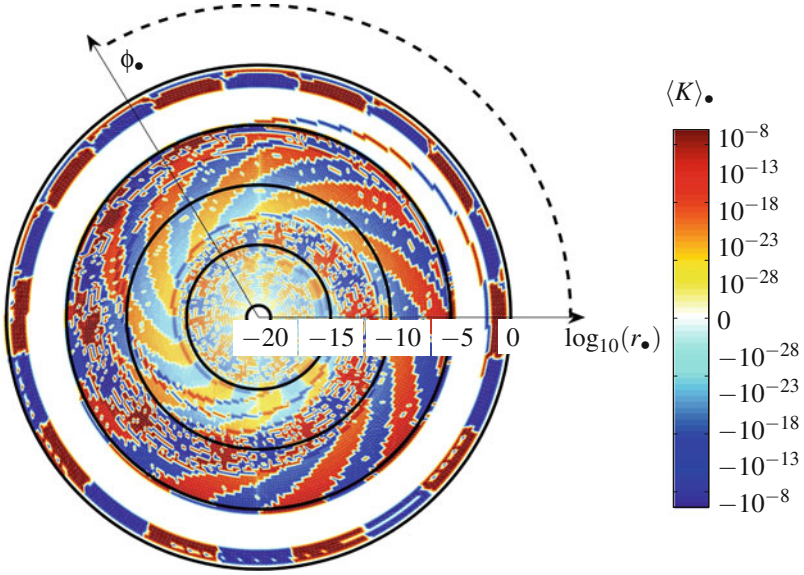


Fig. 5.7 Visualization of the MLC feedback law (5.9) in the a_1-a_2 plane. The figure displays the expectation value (5.11)

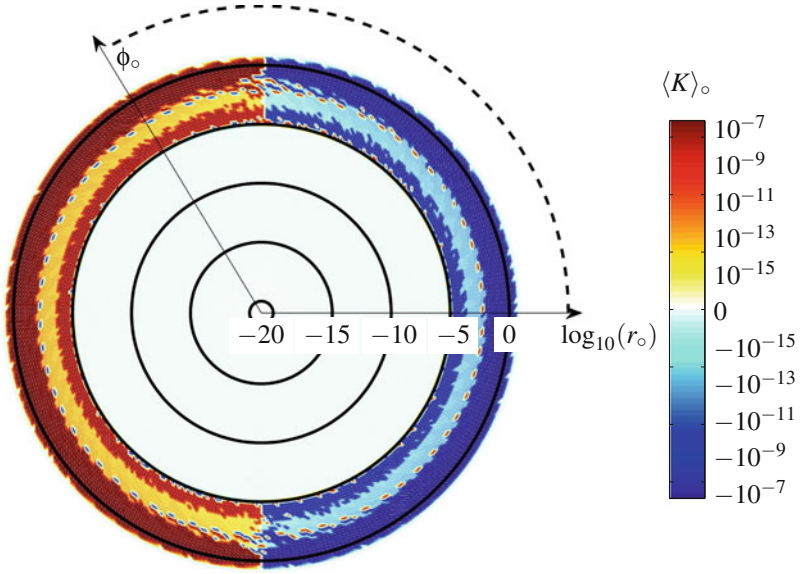


Fig. 5.8 Same as Fig. 5.7 but for (5.12) the a_3-a_4 plane

stable high-frequency oscillator. The mostly white inner ring corresponds to regions without available data from the simulation. Figure 5.8 is an analogous visualization of the control law (5.12) for the a_3-a_4 plane. This figure reveals the destabilizing factor $K_1(a_4)$ of the MLC control law (5.10).

5.3 Derivation Outline for the Generalized Mean-Field Model

In this section, we outline the derivation of the generalized mean-field model of Sect. 5.1. The underlying approximations will be used in the alternative control design (Sect. 5.4). We consider an incompressible uniform flow around an obstacle in a steady domain Ω. The location is denoted by $\mathbf{x} = (x, y, z) \in \Omega$ and the time by t. Here, x, y, z are Cartesian coordinates. For a nominally two-dimensional shear flow, x points in the direction of the flow, y in the direction of the main gradient and z is the spanwise coordinate. The unit vectors in x, y and z directors are $\mathbf{e}_x$, $\mathbf{e}_y$, $\mathbf{e}_z$, respectively. Let $\mathbf{u} = (u, v, w)$ be the velocity and and p be the pressure in this domain, respectively. Here, u, v, w are the Cartesian coordinates of the velocity. Let D and U represent the characteristic size and free-stream velocity, respectively. The incompressible Newtonian fluid is characterized by its density ρ and kinematic viscosity ν. The properties of the flow are determined by the Reynolds number

$Re = UD/\nu$. In the following, all quantities are assumed to be non-dimensionalized with respect to the length scale D, the velocity scale U and density ρ.

The mass conservation or equation of continuity reads

$$\nabla \cdot \mathbf{u}(\mathbf{x}, t) = 0. \tag{5.13}$$

Here, '∇' represents the Nabla operator with respect to $\mathbf{x}$ and '$\cdot$' an inner product. The momentum balance for an incompressible Newtonian fluid is given by the Navier-Stokes equations:

$$\partial_t \mathbf{u}(\mathbf{x}, t) + \mathbf{u}(\mathbf{x}, t) \cdot \nabla \mathbf{u}(\mathbf{x}, t) = -\nabla p(\mathbf{x}, t) + \frac{1}{Re} \triangle \mathbf{u}(\mathbf{x}, t). \tag{5.14}$$

The left-hand side corresponds to the acceleration of the fluid, the right-hand side contains the pressure and viscous forces. Here, $\nabla \mathbf{u}$ represents the velocity Jacobian, i.e. the outer product of ∇ with $\mathbf{u}$. $\triangle$ denotes the Laplace operator.

At the domain boundary $\partial\Omega$, the velocity satisfies Dirichlet conditions: it vanishes at the stationary body (no-slip condition) and assumes free-stream velocity at infinity,

$$\mathbf{u}(\mathbf{x}, t)|_{\partial\Omega} = \begin{cases} \mathbf{0} & \text{at the body;} \\ \mathbf{e}_x & \text{at infinity.} \end{cases} \tag{5.15}$$

Let $\mathbf{v}(\mathbf{x})$ represent the initial condition at time $t = 0$,

$$\mathbf{u}(\mathbf{x}, 0) = \mathbf{v}(\mathbf{x}), \quad \forall \mathbf{x} \in \Omega. \tag{5.16}$$

Equations (5.13)–(5.16) define an initial boundary value problem which is assumed to have a unique solution under sufficiently smooth initial and boundary conditions. The uniqueness is mathematically proven for some two-dimensional flows [169] but is still an open problem for three-dimensional flows. It may be noted that examples of non-uniqueness are found for unsteady boundary conditions [225].

In the following, the derivation of a least-order model for flows dominated by two frequencies will be sketched. Details can be found in the original literature [176]. Generally, the Navier–Stokes equations are assumed to have one (and only one) steady solution $\mathbf{u}_s(\mathbf{x})$ with corresponding pressure field $p_s(\mathbf{x})$,

$$\mathbf{u}_s(\mathbf{x}) \cdot \nabla \mathbf{u}_s(\mathbf{x}) = -\nabla p_s(\mathbf{x}) + \frac{1}{Re} \triangle \mathbf{u}_s(\mathbf{x}). \tag{5.17}$$

There exist only few known exceptions of flows with no steady solution or multiple solutions which concern closed flows, like diffuser flow. Let $\mathbf{u}_\bullet(\mathbf{x}, t)$ denote the frequency contribution at angular frequency $\omega_\bullet$ of the unstable unforced flow. Similarly, let $\mathbf{u}_\circ(\mathbf{x}, t)$ represent the actuated contribution at angular frequency $\omega_\circ$. The later component is assumed to vanish without forcing. $\omega_\bullet$ and $\omega_\circ$ are assumed be incommensurable so that no lock-in occurs. Amplitudes and frequencies of both components may slowly vary with time. Correspondingly slow base flow changes due

to the Reynolds stress are included in $\mathbf{u}_\Delta(\mathbf{x}, t)$. The resulting velocity decomposition reads

$$\mathbf{u}(\mathbf{x}, t) = \mathbf{u}_s(\mathbf{x}) + \mathbf{u}_\Delta(\mathbf{x}, t) + \mathbf{u}_\bullet(\mathbf{x}, t) + \mathbf{u}_\circ(\mathbf{x}, t). \tag{5.18}$$

In a simple, yet not unrealistic case, the unforced oscillation may be well approximated by a linear combination of two spatial modes $\mathbf{u}_1(\mathbf{x})$ and $\mathbf{u}_2(\mathbf{x})$ with time-dependent coefficients $a_1(t)$ and $a_2(t)$. These modes may be the first POD modes, the real and imaginary part of the dominant DMD mode [229, 238] or cosine and sine component of a Fourier mode at $\omega_\bullet$. Similarly, the actuated oscillatory structures can be expected to be well resolved by a linear combination of two modes $\mathbf{u}_3(\mathbf{x})$ and $\mathbf{u}_4(\mathbf{x})$ with amplitudes $a_3(t)$ and $a_4(t)$. For simplicity, the modes are assumed to build an orthonormal basis without loss of generality, as they can easily be orthonormalized. Summarizing,

$$\mathbf{u}_\bullet(\mathbf{x}, t) = a_1(t)\, \mathbf{u}_1(\mathbf{x}) + a_2(t)\, \mathbf{u}_2(\mathbf{x}) \tag{5.19a}$$
$$\mathbf{u}_\circ(\mathbf{x}, t) = a_3(t)\, \mathbf{u}_3(\mathbf{x}) + a_4(t)\, \mathbf{u}_4(\mathbf{x}). \tag{5.19b}$$

Following Kryloff and Bogoliubov [167], the modal amplitudes a_i, $i = 1, \ldots, 4$ are considered to be nearly pure harmonics, i.e.

$$a_1(t) = r_\bullet \cos\phi_\bullet \tag{5.20a}$$
$$a_2(t) = r_\bullet \sin\phi_\bullet \tag{5.20b}$$
$$a_3(t) = r_\circ \cos\phi_\circ \tag{5.20c}$$
$$a_4(t) = r_\circ \sin\phi_\circ \tag{5.20d}$$
$$\frac{d\phi_\bullet}{dt} = \omega_\bullet \tag{5.20e}$$
$$\frac{d\phi_\circ}{dt} = \omega_\circ, \tag{5.20f}$$

where the amplitudes $r_\bullet$ and $r_\circ$ and frequencies $\omega_\bullet$ and $\omega_\circ$ are slowly varying functions of time. It may be noted that (5.20) allows for arbitrary phase offsets. Thus, the Reynolds decomposition of the flow in Eq. (5.18) into a mean $\overline{\mathbf{u}}$ and a fluctuation $\mathbf{u}'$ reads

$$\overline{\mathbf{u}}(\mathbf{x}, t) = \mathbf{u}_s(\mathbf{x}) + \mathbf{u}_\Delta(\mathbf{x}, t) \tag{5.21a}$$
$$\mathbf{u}'(\mathbf{x}, t) = \sum_{i=1}^{4} a_i(t)\, \mathbf{u}_i(\mathbf{x}). \tag{5.21b}$$

The mean velocity is understood as an ensemble or short-term average to allow for slow unforced or actuated transients.

The base-flow deformation $\mathbf{u}_\Delta$ is inferred from the Reynolds equation, i.e. time-averaged Navier–Stokes equations (5.14). We substitute equation (5.21a) in (5.14) and subtract the steady Navier–Stokes equations (5.17). Averaging and neglecting second-order terms in the base-flow deformation $\mathbf{u}_\Delta$ yields

$$\mathbf{u}_s \cdot \nabla \mathbf{u}_\Delta + \mathbf{u}_\Delta \cdot \nabla \mathbf{u}_s = -\nabla p + \frac{1}{Re} \triangle \mathbf{u}_\Delta - \nabla \cdot \overline{\mathbf{u}' \otimes \mathbf{u}'}. \tag{5.22}$$

In this equation, the spatio-temporal dependencies have been dropped for brevity. The right-most term is the Reynolds-stress force driving the base-flow deformation. The symbol '$\otimes$' emphasizes the outer product between the fluctuation vectors, leading to a matrix after the Nabla operator. Note that this system of partial differential equations is linear in $\mathbf{u}_\Delta$ and has a single forcing term. The pressure gradient can be considered as a projection on an incompressible velocity subspace, i.e. it neither interferes with the linearity in $\mathbf{u}_\Delta$ nor with the forcing term.

The Reynolds stress is given by

$$\overline{\mathbf{u}' \otimes \mathbf{u}'} = \frac{1}{2} r_\bullet^2 \left(\mathbf{u}_1 \otimes \mathbf{u}_1 + \mathbf{u}_2 \otimes \mathbf{u}_2\right) + \frac{1}{2} r_\circ^2 \left(\mathbf{u}_3 \otimes \mathbf{u}_3 + \mathbf{u}_4 \otimes \mathbf{u}_4\right) \tag{5.23}$$

exploiting (5.20). Evidently, the Reynolds-stress term has one contribution at the natural frequency $\omega_\bullet$ and another one at the actuated one $\omega_\circ$. By the linear nature of (5.22), the base-flow deformation of both frequency components of the Reynolds stress are additive and can be associated with two shift-modes [196, 199]. Let $a_5\,\mathbf{u}_5$ be the base-flow change corresponding to the natural frequency $\omega_\bullet$ and $a_6\,\mathbf{u}_6$ the analog of the actuated frequency $\omega_\circ$. From (5.22) and (5.23), we observe

$$a_5 = \alpha_\bullet r_\bullet^2 = \alpha_\bullet \left(a_1^2 + a_2^2\right) \tag{5.24a}$$

$$a_6 = \alpha_\circ r_\circ^2 = \alpha_\circ \left(a_3^2 + a_4^2\right). \tag{5.24b}$$

These equations define the mean-field manifolds hosting slow transients in the 6-dimensional state space $\mathbf{a} = [a_1, a_2, \ldots, a_6]^T$.

The dynamic equations for a_i, $i = 1, 2, 3, 4$ can be obtained from the Navier–Stokes equations (5.14) exploiting the Kryloff–Bogoliubov approximation. Filtering (5.14) for $\omega_\bullet$ terms ignores all constant and quadratic terms since none of them can give rise to the frequency $\omega_\bullet$. The projection onto $\mathbf{u}_i$ for $i = 1, 2$ yields an oscillator which is base-flow dependent. A similar reasoning holds for the $\omega_\circ$ frequency. A volume force gives rise to an additive forcing term gb where b is the actuation command and g the gain. Without loss of generality, this forcing acts on the dynamic equation for a_4, as the modes $\mathbf{u}_i$, $i = 3, 4$ can be rotated. Summarizing, we obtain (5.1).

5.4 Alternative Control Approaches

In the following, we will assess the efficiency of machine learning control and benchmark it against periodic forcing (Sect. 5.4.1), and against an energy-based closed-loop control design (Sect. 5.4.2). In Sect. 5.4.3, the efficiency of MLC on-off control is assessed in an analytical framework. We rewrite the generalized mean-field model (5.2), giving all growth-rates symbols:

$$\frac{da_1}{dt} = \sigma_\bullet\, a_1 - a_2 \tag{5.25a}$$

$$\frac{da_2}{dt} = \sigma_\bullet\, a_2 + a_1 \tag{5.25b}$$

$$\frac{da_3}{dt} = \sigma_\circ\, a_3 - 10\, a_4 \tag{5.25c}$$

$$\frac{da_4}{dt} = \sigma_\circ\, a_4 + 10\, a_3 + b \tag{5.25d}$$

$$\sigma_\bullet = \sigma_\star - a_1^2 - a_2^2 - a_3^2 - a_4^2 \tag{5.25e}$$

$$\sigma_\star = 0.1 \tag{5.25f}$$

$$\sigma_\circ = -0.1. \tag{5.25g}$$

In the analytical computations, we will keep the growth-rate symbols of (5.25e)–(5.25g) to track the physical meaning of each formula.

5.4.1 Open-Loop Forcing

In this section, we minimize the cost functional (5.6) with (open-loop) periodic forcing. The stable oscillator is efficiently excited at its eigenfrequency

$$b = B\cos(10t)\,. \tag{5.26}$$

The resulting fluctuation amplitude is proportional to the forcing amplitude, or, equivalently,

$$r_\circ^2 = \kappa\, B^2. \tag{5.27}$$

Here, κ is proportional to the reciprocal of the damping rate $\sigma_\circ$.

In the sequel, we apply the Kryloff–Bogoliubov approximation (5.20) for slowly varying amplitudes. Then, the first oscillator assumes a non-vanishing amplitude $r_\bullet > 0$ if and only if

$$\frac{dr_\bullet}{dt} = \sigma_\bullet\, r_\bullet = 0.$$

Equation (5.25e) implies the fluctuation level

$$J_a = r_\bullet^2 = \sigma_\star - r_\circ^2 = \sigma_\star - \kappa\, B^2. \tag{5.28}$$

The oscillation completely vanishes if $\sigma_\bullet \leq 0$ or, equivalently $B^2 \geq \sigma_\star/\kappa$. The associated actuation cost reads

$$J_b = \overline{b^2} = B^2/2. \tag{5.29}$$

Finally, the cost functional (5.6) reads

$$J = J_a + \gamma J_b = \sigma_\star - \kappa\, B^2 + \frac{\gamma}{2} B^2 = \sigma_\star + \left[\frac{\gamma}{2} - \kappa\right] B^2. \tag{5.30}$$

At vanishing actuation $B = 0$, we have the unactuated limit cycle and

$$J_\bullet = \sigma_\star. \tag{5.31}$$

For later reference, we give this cost value the subscript '$\bullet$'. Complete stabilization $r_\bullet = 0$ can be achieved with the minimum actuation level $B^2 = \sigma_\star/\kappa$, corresponding to the cost

$$J_\circ = \frac{\gamma\, \sigma_\star}{2\, \kappa}. \tag{5.32}$$

For later reference, this cost value has the subscript '$\circ$'. Intermediate actuation amplitudes $0 < B^2 < \sigma_\star/\kappa$ yield intermediate costs. The trade-off between achieved stabilization and actuation cost is easily appreciated in the Pareto diagram in Fig. 5.9.

Intriguingly, the solution of the optimization problem depends discontinuously on γ. If $\gamma < \gamma_{\text{crit}} := 2\,\kappa$, $J_\circ < J_\bullet$ and minimization of the cost functional leads to complete stabilization of the unstable oscillator. If $\gamma > \gamma_{\text{crit}}$, minimization leads to the unactuated limit cycle with vanishing forcing. If $\gamma = \gamma_{\text{crit}}$, any forcing $0 \leq B^2 \leq \sigma_\star/\kappa$ leads to the same J and the minimization problem has no unique solution. The γ chosen in Sect. 5.2.1 was subcritical. Hence, MLC has targeted complete stabilization.

The periodic forcing constitutes a benchmark against which closed-loop control can be measured. Periodic forcing is easily realizable in any system and any experiment. The optimal forcing parameters can be determined, e.g. by gradient search or extremum/slope seeking. The corresponding Pareto diagram illustrates which level of stabilization can be achieved at which actuation cost. In principle, the best closed-loop control may be inside or outside the triangle $J_a + J_b \leq J_\bullet$. There is no a priori guarantee that closing the loop will beat periodic forcing. We shall explore this aspect in later sections.

We shall not pause to inquire if Eq. (5.26) defines the best open-loop actuation command $b = K(t)$ with respect to the cost-functional.

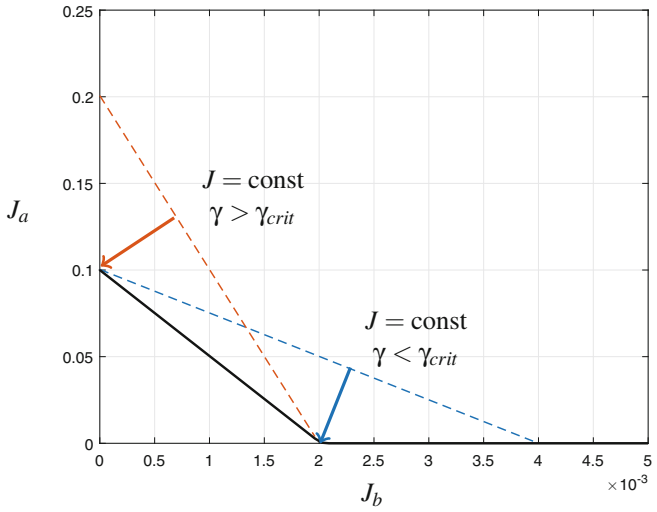


Fig. 5.9 Pareto diagram for the stabilization of the generalized mean-field model (5.2). The *black line* denotes equilibrium points for open-loop forcing from the unforced limit cycle ($B = 0$) to complete stabilization $r_\bullet = 0$ according to Eq. (5.30). For more details see text

5.4.2 Closed-Loop Forcing

In this section, we design a closed-loop forcing (5.8) which stabilizes the first oscillator. As seen in Sect. 5.1, linear control theory is not applicable. Instead, we employ an energy-based control design under the Kryloff–Bogoliubov approximation (5.20).

The starting point is the fluctuation level

$$J_a = \overline{r_\bullet^2} = r_\bullet^2 = a_1^2 + a_2^2.$$

We assume an average over an infinite time window, i.e. we neglect transient behavior. Thus, the averaging sign over $r_\bullet^2$ is redundant under the Kryloff–Bogoliubov approximation. Differentiating with respect to time and employing the evolution equations (5.25a), (5.25b) and dividing by 2 yields

$$r_\bullet \frac{r_\bullet}{dt} = a_1 \frac{a_1}{dt} + a_2 \frac{a_2}{dt} = \sigma_\bullet \, r_\bullet^2.$$

Stabilization of the first oscillator implies $r_\bullet = 0$ by definition and $\sigma_\bullet < 0$ for stability under noise. From Eq. (5.25e), this requires an excitation of the second oscillator to the level $r_\circ^2 \geq \sigma_\star$. It should be noted that the Kryloff–Bogoliubov assumption implies slowly varying amplitudes and accounting for time-averaging effects is not necessary. Hence, stabilization of the first oscillator implies a destabilizing control

for the second one. We proceed as with the first oscillator, and differentiate

$$r_\circ^2 = a_3^2 + a_4^2$$

with respect to time, employ Eq. (5.25c), (5.25d), and divide by 2 to obtain

$$r_\circ \frac{r_\circ}{dt} = a_3 \frac{a_3}{dt} + a_4 \frac{a_4}{dt} = \sigma_\circ \, r_\circ^2 + a_4 b. \tag{5.33}$$

For limit-cycle behavior, the average energy $\overline{a_4 b}$ needs to overcome the dissipation $\sigma_\circ r_\circ^2$.

$$0 = \sigma_\circ r_\circ^2 + \overline{a_4 b}. \tag{5.34}$$

We see that b contributes to the fluctuation energy only if it has the same sign as a_4. This is satisfied by the linear feedback ansatz

$$b = K a_4, \tag{5.35}$$

with $K > 0$. The sinusoidal behavior (5.20d) allows one to estimate the actuation power with $\overline{a_4 b} = K \overline{a_4^2} = K r_\circ^2/2$. Thus, the steady-state gain can be derived from Eq. (5.34) to be $K = -2\sigma_\circ$, leading to

$$b = -2\sigma_\circ \, a_4. \tag{5.36}$$

This control law implies a vanishing growth rate, or $r_\circ = \text{const}$, where the constant is determined by the initial conditions. In other words, Eq. (5.36) does not drive the actuated dynamics towards specific limit-cycle radii.

In contrast, the nonlinear gain

$$b = K \, a_4, \quad \text{where} \quad K = -2\sigma_\circ + \sigma_\star - r_\circ^2 \tag{5.37}$$

ensures that the minimal fluctuation level $r_\circ^2 = \sigma_\star$ is stabilized. Substituting Eq. (5.37) in (5.33) and averaging over one period yields the following amplitude equation for the actuated dynamics:

$$\frac{dr_\circ}{dt} = \frac{1}{2} \, r_\circ \left(\sigma_\star - r_\circ^2 \right).$$

A fluctuation level that is too small (large) is compensated for by a larger (smaller) gain K as compared to the equilibrium value $-2\sigma_\circ$.

The above nonlinear feedback law (5.37) stabilizes the desired fluctuation level of the second oscillator but does not compensate for any error of the dynamics. The alternative law

$$b = K \, a_4 \quad \text{where} \quad K = -2\sigma_\circ + 10 \, r_\bullet^2 \tag{5.38}$$

increases the gain sharply if the first oscillator is not stabilized.

In summary, the discussed nonlinear feedback laws lead to the optimal periodic forcing of Sect. 5.4.1 with the same cost functional. There is no steady-state performance benefit from using the discussed closed-loop control. This equivalence is not overly surprising as the very Kryloff–Bogoliubov assumption implies nearly periodic behavior of both oscillators. Yet, feedback can buy an improved stabilization in case of model uncertainty, e.g. accounting for unknown errors of the growth rates. Accounting for model errors is one of the very purposes of feedback.

5.4.3 Short-Term Forcing

In this section, we analytically assess the benefits from strong short-term periodic forcing (5.26) to reduce $r_\bullet$ from $r_{\max} \ll \sqrt{\sigma_\star}$ to $r_{\min}$. This forcing is an idealization of the MLC law in Sect. 5.2. Here, $r_{\max}$ is at least 4 orders of magnitude below its unforced limit-cycle value.

Let $\sigma_a < 0$ be the commanded decay rate during periodic forcing. According to Eq. (5.25e),

$$\sigma_a = \sigma_\star - r_\bullet^2 - r_\circ^2 \approx 0.1 - r_\circ^2.$$

In this approximation, we ignore $r_\bullet^2 \leq r_{\max}^2 \ll \sigma_\star$ by the smallness assumption of $r_{\max}$. In addition, we neglect transient effects, as the second oscillator is forced over many cycles. Then, we can use the quasi-equilibrium assumption of Sect. 5.4.1 and arrive at

$$\sigma_a = \sigma_\star - \kappa B^2. \tag{5.39}$$

By similar reasoning, the unactuated growth rate reads

$$\sigma_u = \sigma_\star, \tag{5.40}$$

since $r_\circ$ vanishes without actuation and $r_\bullet$ is assumed to be negligible as compared to $\sqrt{\sigma_\star}$. Summarizing,

$$\frac{dr_\bullet}{dt} = \sigma_\bullet\, r_\bullet \quad \text{where} \quad \sigma_\bullet = \begin{cases} \sigma_a & \text{during actuation} \\ \sigma_\star & \text{otherwise} \end{cases}. \tag{5.41}$$

The time interval τ_a for the actuation is given by

$$-\sigma_a\, \tau_a = \ln\left[\frac{r_{\max}}{r_{\min}}\right]. \tag{5.42}$$

Similarly, the time for the unforced period τ_u reads

$$\sigma_\star \, \tau_u = \ln \left[\frac{r_{\max}}{r_{\min}} \right]. \tag{5.43}$$

The period for one on-off cycle is the sum:

$$\tau = \tau_a + \tau_u = \left[-\frac{1}{\sigma_a} + \frac{1}{\sigma_\star} \right] \ln \left[\frac{r_{\max}}{r_{\min}} \right]. \tag{5.44}$$

The ratio of the actuation time with respect to this period is

$$\frac{\tau_a}{\tau} = \frac{\frac{-1}{\sigma_a}}{\left[\frac{1}{\sigma_\star} - \frac{1}{\sigma_a} \right]} = \frac{1}{\left[1 - \frac{\sigma_a}{\sigma_\star} \right]} = \frac{1}{\left[1 - \frac{\sigma_\star - \kappa B^2}{\sigma_\star} \right]} = \frac{\sigma_\star}{\kappa B^2}. \tag{5.45}$$

The stronger the actuation, the smaller the relative actuation time. Note that the ratio does not depend on the limits imposed on $r_\bullet$.

Following earlier reasoning, the stabilization can be considered complete, since $J_a = r_\bullet^2 \ll J_\bullet$. The only contribution to the cost functional comes from the actuation. The average actuation level is the product between the relative actuation time τ_a/τ and the maximum actuation level $B^2/2$:

$$J = \gamma \overline{b^2} = \gamma \, \frac{\tau_\circ}{\tau} \, \frac{B^2}{2} = \gamma \, \frac{\sigma_\star}{\kappa B^2} \, \frac{B^2}{2} = \gamma \, \frac{\sigma_\star}{2\kappa} = J_\bullet. \tag{5.46}$$

Intriguingly, the cost of on-off actuation is identical to the best periodic forcing $J_\bullet$ of Eq. (5.32). Numerically the MLC control law is found to be slightly better due to a finite-window effect. The difference decreases with increasing integration time. MLC exploits even this finite-window effect for closed-loop control design.

The decision to turn actuation on or off in the framework of full-state feedback (5.8) is far from obvious. One 'relay switch' using the Heaviside function H reads

$$\chi = H\left(r_\bullet - r_{\max}\right) - H\left(r_{\min} - r_\bullet\right) + H\left(-\sigma_\star + r_\circ^2\right).$$

If $r_\bullet > r_{\max}$, $\chi > 0$ and actuation is turned on. If $r_\bullet < r_{\min}$, $\chi \leq 0$ and actuation is turned off. At intermediate values $r_{\min} < r_\bullet < r_{\max}$, χ is kept on if actuation has a damping effect (actuated transient) and χ is kept off if the second oscillator is not excited enough. MLC has constructed such a switch for an incremental finite-window performance benefit. This is an impressive performance of an automated control design.

5.5 Exercises

Exercise 5–1: Consider the following three coupled oscillators

$$\frac{da_1}{dt} = \sigma_1\, a_1 - a_2 \tag{5.47a}$$
$$\frac{da_2}{dt} = \sigma_1\, a_2 + a_1 \tag{5.47b}$$
$$\frac{da_3}{dt} = \sigma_2\, a_3 - \pi\, a_4 \tag{5.47c}$$
$$\frac{da_4}{dt} = \sigma_2\, a_4 + \pi\, a_3 + b \tag{5.47d}$$
$$\frac{da_5}{dt} = \sigma_3\, a_5 - \pi^2\, a_6 \tag{5.47e}$$
$$\frac{da_6}{dt} = \sigma_3\, a_6 + \pi^2 a_5 + b \tag{5.47f}$$
$$\sigma_1 = -r_1^2 + r_2^2 - r_3^2 \tag{5.47g}$$
$$\sigma_2 = 0.1 - r_2^2 \tag{5.47h}$$
$$\sigma_3 = -0.1, \tag{5.47i}$$

where $r_1^2 := a_1^2 + a_2^2$, $r_2^2 := a_3^2 + a_4^2$, and $r_3^2 := a_5^2 + a_6^2$. Explore the unforced behavior ($b \equiv 0$) by numerical simulations. Explain the coupling between the oscillators in words. Derive an analytical solution of the unforced system (5.47).

Exercise 5–2: Stabilize the first oscillator of Eq. (5.47) with a full-state feedback law $b = b(\mathbf{a})$ by minimizing

$$J = \overline{r_1^2} + \overline{b^2}. \tag{5.48}$$

Linearize (5.47) and design a corresponding LQR controller (see Chap. 4). Explain the results.

Exercise 5–3: Stabilize the first oscillator of (5.47) with a full-state nonlinear feedback law $b = b(\mathbf{a})$ by minimizing J of Eq. (5.48). Use the Kryloff–Bogoliubov approximation of Sect. 5.4. Explain the results.

Exercise 5–4: Find the best periodic actuation

$$b = B\, \cos(\omega t)\,. \tag{5.49}$$

(a) Set $B = 1$ and perform a frequency scan of ω, which effects all oscillator amplitudes r_1, r_2, r_3. Can you explain the extrema of the amplitudes?
(b) Determine analytically the best control law with smallest J of Eq. (5.48), i.e. determine the best B and Ω. Justify physically why these parameters are

optimal (no proof needed). Is this open-loop control better or worse than the closed-loop of the previous exercise? Why?

Exercise 5–5: Apply MLC with the same parameters as in Sect. 5.2. Take

$$\mathbf{a}(0) = [0.1, 0, 0.1, 0, 0.1, 0]^T$$

as initial condition, integrate 20 periods of the first oscillator ($t \in [0, 20\pi]$) and evaluate the cost functional in the next 100 periods ($t \in [20\pi, 220\pi]$). Bound the actuation by the interval $[-1, 1]$. Can you explain the control law and solution? How does it compare with the closed-loop and open-loop solution of Exercises 3 and 4?

5.6 Suggested Reading

Texts

(1) **Turbulence, Coherent Structures, Dynamical Systems and Symmetry**, by P. Holmes, J.L. Lumley, G. Berkooz and C.W. Rowley, 2012 [138].
The book represents a classic of POD Galerkin models of turbulent flows from the pioneers of the field.

(2) **Nonlinear Ordinary Differential Equations**, by D.W. Jordan and P. Smith, 1988 [148].
This textbook provides an easily comprehensible and thorough introduction into nonlinear dynamics and the Kryloff–Bogoliubov approximation used in this chapter.

Seminal papers

(1) **Rods and plates: series occurring in various questions regarding the elastic equilibrium of rods and plates (translated)**, by B.G. Galerkin 1915 [109].
This seminal paper proposed an elegant method for deriving ordinary differential equations (ODE) from partial differential equations (PDE) using modal expansions. This Galerkin method has become a very foundation for over 100 years of research in computational methods for PDEs and in reduced-order modeling.

(2) **Nonlinear stability theory**, by J.T. Stuart, 1971 [257].
This review article summarizes the development of mean-field models which were pioneered by the author and which are the foundation of this chapter. J.T. Stuart was the first to derive a low-order Galerkin model explaining the coupling between fluctuation and base flow.

5.7 Interview with Professor Mark N. Glauser

Mark Glauser is Professor of Mechanical and Aerospace Engineering and Associate Dean for Research and Doctoral Programs at the College of Engineering and Computer Science of Syracuse University, NY, USA. He is also Professor of Physics at the College of Arts and Sciences of the same university.

As Associate Dean for Research and Doctoral Programs, Prof. Glauser is responsible for overseeing current research activities and coordinating the development of the college's future research portfolio. In his own research portfolio, Prof. Glauser, along with his co-workers, post-docs, graduate and undergraduate students, conducts major experimental, computational and theoretical efforts to apply low-dimensional models to turbulent and transitional flows for understanding and control. Flows studied range from high speed aerospace type applications to those around thermal breathing manikins within the micro-environment. Recent work involves developing closed-loop flow control methods based on the use of Proper Orthogonal Decomposition (POD) and Stochastic Measurement (SM) for various turbulent flows including that over a NACA 4412 airfoil, high speed (high subsonic and supersonic) turbulent jets for noise reduction/enhanced mixing, 3D separated flow control over turrets for improving aero-optics and for improving efficiency and reducing unsteady loading on large wind turbines. Prof. Glauser has or is currently serving as: a member of the US Army Science Board where he just finished co-chairing a 2014–2015 study on The Future of Army Aviation; as a member of the NASA Langley Fundamental Aerodynamics Peer Review Panel (2014, 2009); Associate Editor, AIAA Journal (2007–2016); Program Manager for the Turbulence and Internal Flows Program at the US Air Force Office of Scientific Research (AFOSR) from 1996–1999; meeting Chair for the 56th APS Annual Meeting of the Division of Fluid Dynamics, November 2003; Technical Chair for the AIAA Summer Fluid Dynamics Meeting, June 2006; an ABET evaluator for Aerospace Engineering programs since 2004; and an ABET EAC member (2013–2015). Prof. Glauser has obtained more than 12 million dollars in research funding as PI or Co-PI from AFOSR, NSF, NASA, EPA, DoE, Dantec, GE, United Technologies, Spectral Energies, ClearScience Corporation and others.

Prof. Glauser has published more than 110 peer-reviewed publications and conference proceedings and has presented more than 100 invited presentations and keynote talks worldwide. Over the past 25+ years he has mentored several postdocs and more than 30 Ph.D. and MS students. Prof. Glauser is a Fellow of the American Institute of Aeronautics and Astronautics, the American Society of Mechanical Engineers, the American Physical Society, and the Institute of Physics (UK). In 1995, he was a Fulbright Scholar in Poitiers, France. Prof. Glauser received his BS (1982) and his Ph.D. (1987) from the Department of Mechanical and Aerospace Engineering of the University at Buffalo SUNY, NY, USA.

Authors: Dear Mark, you have been one of few pioneers in reduced-order modeling of turbulence, particularly for application-related experiments. What did you learn about flow control? Where do you still see rewarding research opportunities for young talented researchers?

Prof. Glauser: Beginning with the ActiveWing dynamic separation flow work (joint with Lumley, Leibovich and Berkooz, see Taylor and Glauser 2004 [261]) in the mid 1990 s to the NACA 4412 closed-loop separation control work (see Pinier et al. 2007 [215]), to the aero-optics related turret separation control work (Wallace et al. 2012 [265, 274]) and to our more recent high speed jet control work [26, 175], I have learned that this is a hard problem and that the most progress is made when there is strong interaction between controls and fluids experts with a nice mix of experimentalists, theorists and omputationalists from both fields.
It is my view that there are many rich and interesting closed loop flow control problems in the Energy and Aerospace sector and beyond. Reducing unsteady loading on wind turbines with large wind farms for example is an important potential application for closed loop flow control. With the world-wide explosion of Unmanned Ariel Systems this would seem to be an especially important area due to the need for advanced intelligent platforms that can operate safely in complex and uncertain environments (gusts and other extreme weather events, degraded visual environments and etc.).

Authors: You have also pioneered closed-loop turbulence control in real-world experiments. You have decided to perform a model-free control and did not use your reduced-order models for control design. Why?

Prof. Glauser: This has not been entirely the case. The closed-loop flow control work on turrets for Aero-optics applications incorporated models (joint work with Hal Carlson, see for example Wallace et al. 2012 [274]) and conceptually the early ActiveWing work with Lumley and Leibovich had modeling at its core. Frankly, it is just not always possible to have a full team to handle the challenges associated with bringing in the models due to funding constraints. In addition, the complex experiments we run in our lab are very challenging and it is generally not feasible to have a Ph.D. student do both the modeling and experiments and have them graduate in a reasonable time frame. With the Aero-optics work where

modeling was incorporated we were fortunate enough to have a strong team across the board. In addition, the experiments we have been doing in jets are very high Reynolds number and hence the flows themselves are very high dimensional so model development is more difficult than for the lower dimensional separation flow control problems we have performed. The bottom line, the flows examined and availability (or lack of) of a complete team have played the key roles in the level of modeling that we have been able to successfully incorporate.

Authors: Yet, the literature contains myriad of studies on model-based flow stabilization in numerical simulations. What is the difference between experimental and numerical control?

Prof. Glauser: This is partially answered in my response to the question above. Typically many of the numerical simulation-based flow control studies have been at lower Reynolds number with relatively simple boundary conditions. Thankfully the simulation tools are improving and we are starting to reach more realistic Reynolds numbers with LES. The experimental tools are improving as well, including powerful Time Resolved PIV tools. The best approach, if possible is to work the problem from both sides, using the high spatial resolution of simulations to provide key guidance to experiments. This can include, for example, simulation-guided placement of sensors and actuators along with key time and spatial scales at which to drive the flow to achieve the desired control objectives. Simulation derived low-dimensional models, even if somewhat limited, can be used, at least as a starting point, or perhaps fused with experimentally derived models, to provide the model-based control.

Authors: Where do you see the range of applicability of model-based control which has motivated this chapter?

Prof. Glauser: In principle, model-based control can and should, if possible, be used across the range of applications experienced in the energy and aerospace sector and beyond.

Authors: You have been enthusiastically supporting computer science methods for years. Can you give us an idea about evolving machine learning applications in turbulence control in the coming decade?

Prof. Glauser: It is my view that machine learning methods must be brought to bear on the difficult nonlinear stochastic problem we are trying to control if we are going to make real progress. However, I view machine learning as a complement to our Navier–Stokes based tools and not an either-or scenario. All of it should be thought of and used as "information" to help solve the nonlinear control problems we are faced with.

Authors: Which fluid dynamics expertise is not likely to be replaced by machine learning in the coming decade?

Prof. Glauser: We will continue to need theorists, experimentalists and computationalists, all who, however, in my view, will need to have a working knowledge of the latest math and computer science tools for both understanding and controlling high dimensional non-linear time dependent stochastic systems such as turbulence.

Authors: We look forward to your next breakthroughs in experimental turbulence control and thank you for this interview!

Chapter 6
Taming Real World Flow Control Experiments with MLC

An approximate answer to the right problem is worth a good deal more than an exact answer to an approximate problem.
John Tukey

In Chap. 2, MLC was introduced as a generic method to identify optimal control laws for arbitrary dynamics. In Chaps. 4 and 5, MLC has been applied to the control of low-dimensional dynamical systems. In these examples, we have shown (1) that it is comparable to optimal linear control design for linear dynamics, (2) that it outperforms linear control methods in the case of a weak nonlinearity, and (3) that it can even identify the enabling strongly nonlinear actuation mechanism in the case that the linear dynamics are uncontrollable.

In this chapter, we describe and exemplify the application of MLC to real-world turbulence control experiments. These come with the additional challenges of high-dimensional dynamics, long time delays, high-frequency noise, low-frequency drifts and, last but not least, with the non-trivial implementation of the algorithm in the experimental hardware. In experiments, MLC is executed in the same way as for the dynamical system plants in Chaps. 4 and 5:

1. MLC provides a generation of control laws to be evaluated by the plant.
2. The plant is used to evaluate and grade these individuals with respect to the given cost function.
3. MLC evolves the population.
4. The process stops when a pre-determined criterion is met.
5. After this learning phase, the best control law can be used.

The only difference between a simulation and an experiment is the need to interrogate an experimental plant. This is a technical challenge but not a conceptual point of departure from MLC. Running MLC on an experiment using an existing code is a matter of a few days to a week of work, if the experimental hardware and software is ready for closed-loop control.

We choose three configurations: a laminar flow over a backward-facing step in a water tunnel (Sect. 6.1), separating turbulent boundary layers in wind tunnels

T. Duriez et al., *Machine Learning Control – Taming Nonlinear Dynamics and Turbulence*, Fluid Mechanics and Its Applications 116,
DOI 10.1007/978-3-319-40624-4_6

(Sect. 6.2), and a turbulent mixing layer (Sect. 6.3). These examples encompass key phenomena encountered in most flow control problems: boundary layers, separation, mixing layers and a recirculation zone. Different kind of sensors, actuators and time scales are used and illustrate the versatility of MLC. Section 6.4 highlights the limitations of model-based linear control for the turbulent mixing layer. In Sect. 6.5, we focus on implementation issues with respect to software and hardware. Section 6.6 suggests reading on a spectrum of flow control aspects. Our interview (Sect. 6.7) addresses past and future developments in experimental closed-loop turbulence control with Professor Williams, a pioneer and leading scholar of this field.

6.1 Separation Control Over a Backward-Facing Step

The first example is the control of the recirculation zone behind a backward-facing step in a water tunnel. This experiment has been performed by Nicolas Gautier and Jean-Luc Aider in the PMMH laboratory at ESPCI, Paris. This first application of MLC in separation control is described in detail in [111].

6.1.1 Flow Over a Backward-Facing Step

The flow over a backward-facing step is an important benchmark configuration in fluid mechanics. From a practical perspective, this flow represents cold mixing in a simplified combustor configuration. Increases in cold mixing, i.e., smaller recirculation zones, indicate better combustion in corresponding reacting flow. From a more academic perspective, the flow features an incoming boundary layer, which undergoes a geometrically forced separation at the edge of the step [9, 22, 141]. Excluding creeping flow, the fluid particles are unable to follow the 90° turn at the step and separate. Thus, a mixing layer is formed between the recirculation zone and the outer flow. As is typical for a mixing layer, a Kelvin–Helmholtz instability triggers vortex shedding. This advective mixing increases the width of the vorticity region downstream and determines the position of the reattachment point. This point defines the length of the recirculation zone, L_{rec}. A literature survey indicates that acting on the shear layer at the most amplified frequency contributes to the build-up of eddies, enhances mixing and, at the end, reduces the length of the recirculation zone [64, 68, 112, 130].

The described separation phenomena can be witnessed in many applications with sharp corners: flows over mountains, cars, buildings, mixing elements in microfluidics, just to name a few. This separation affects the lift and drag of the obstacle, the resulting noise level or the mixing.

The actuation of the experiment is set up so that there is no direct access to the shear-layer oscillation at the high-receptivity point, the corner. For a sensor signal, MLC is given the extent of the recirculation zone defined as the area with backward flow. This sensor is blind to the phase of the Kelvin–Helmholtz vortex shedding.

The interest in this choice of sensor is twofold. First, MLC must now discover the shedding frequency indirectly or else MLC has to find another mechanism to control the flow. Second, from an engineering point of view, the recirculation area is much easier to infer than the state of the shear layer. For instance, optical monitoring of seeding or combustion processes can be used instead of particle image or hot-wire velocimetry. This is particularly true for hot reacting flow which rules out most velocity and pressure measurements but which may be optically assessed.

6.1.2 Experimental Setup at PMMH

The PMMH water tunnel (Fig. 6.1a) operates a gravity driven flow with velocities up to 22 m/s. Its test section is $L = 0.8$ m long, $l = 15$ cm wide and $H_{\text{section}} = 20$ cm high (before the step). The step height is $h_{\text{step}} = 1.5$ cm, as depicted in

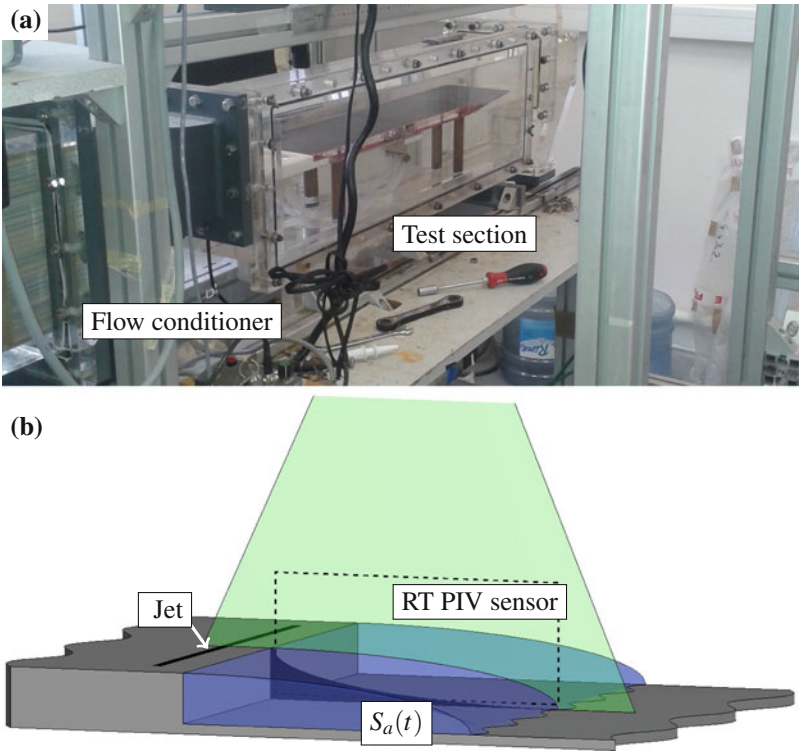


Fig. 6.1 (**a**) Photograph of the PMMH experiment. (**b**) Experimental configuration. A slotted jet is situated just upstream of the separation and performs blowing or suction in the boundary layer. A laser sheet is placed in the symmetry plane for real-time PIV and to determine the backward-flow region

Fig. 6.1b. The operating Reynolds number for the experiment results presented here is $Re = U_\infty \times h_{\text{step}}/\nu = 1350$, where ν is the kinematic viscosity of water. At this Reynolds number, the flow is well separated, creating a shear layer between the high-speed stream and the low-speed recirculation zone extending from the step to the reattachment point. Furthermore, at this Reynolds number, large vortices are visible in the mixing layer.

The control goal is to reduce the size of the recirculation zone. The actuation is achieved by blowing or sucking in a nominally two-dimensional slot upstream of the step, which is oriented at 45° with respect to the streamwise and wallnormal directions. By regulating the pressure in the reservoir of the slot, the exit velocity of the slot can be changed (positive for a jet, negative in case of suction). The exit velocity is taken as actuation command b.

The recirculation size is monitored by a Real-Time (RT) Particle Image Velocimetry (PIV) system which determines the flow fields at a 42 Hz frequency. This sampling frequency is over one order of magnitude larger than the characteristic Kelvin–Helmholtz frequency of around 1 Hz. However, the PIV system would be too slow for the following wind-tunnel experiments with frequencies around 10–100 Hz. There are many possibilities to estimate the size of a recirculation zone. We choose a simple quantity, namely the area in which instantaneous streamwise velocity is negative. The sensor signal $s(t)$ is defined as the normalized instantaneous recirculation zone,

$$s(t) = \frac{S_a(t)}{S_{a,u}}, \tag{6.1}$$

where

$$S_a(t) = \int H(-u(x, y, t))dxdy,$$
$$S_{a,u} = \langle S_a(t) \rangle_T, \text{ without actuation.}$$

Here, u is the streamwise velocity component, H the Heaviside function and $\langle \cdot \rangle_T$ a time-averaged value of its argument over period T. Note that $S_{a,u}$ is the time-averaged recirculation area for the uncontrolled flow. The chosen sensor is not sensitive to the shear-layer vortices for the reasons mentioned above.

The goal function J reads:

$$J = \langle s \rangle_T + \gamma \langle |b| \rangle_T^2, \tag{6.2}$$

where γ is a penalization coefficient for the actuation b. This parameter sets the admissible trade-off between realizing the control objective (reducing the recirculation) and the cost of the actuation. A low value of γ will favor performance over economy and a high value gives preference to economy over performance.

Table 6.1 MLC parameters used for the control of the backward-facing step flow

Parameter	Value
N_i	500
P_r	0.1
P_m	0.20
P_c	0.70
N_p	7
N_e	1
Node functions	$+, -, \times, /$, exp, log, tanh

Ideally, the actuation power investment should be measured against the achieved power savings. One example is aerodynamic drag reduction of a car in which the cost function comprises the saving in propulsion power minus the invested actuation power, i.e., the net energy savings. In the backward-facing step, mixing enhancement has no direct energetic benefit and we resort to an order-of-magnitude argument for choosing γ. Departure point is an optimal periodic forcing which minimizes the recirculation zone. In particular, the penalization term of this periodic forcing is set equal to the normalized size of the recirculation zone, i.e., unity. This arguably avoids the case where the actuation cost is under- or over-emphasized and leads to a value of $\gamma = 3/2$. The parameters used to apply MLC are listed in the Table 6.1.

Advanced material 6.1 Activated options for experimental applications.
Contrary to previous implementations of `OpenMLC` in this book, there are the following changes for experiments:

(a) All individuals of a new generation are evaluated, even if they have already been evaluated in previous generations. The cost function of an individual is the average value over all generations.
(b) The 5 best individuals of the final generation are evaluated 5 times. The cost function is, now, the average value from the final generation.
(c) Control laws that are dismissed as unpromising in the first generation (e.g., valves closed or open more than 90 % of the time) are replaced until all individuals of the first generation are considered as legitimate candidates.

The options (a) and (b) are extremely important, for two reasons: (1) The evaluation time $T = 10$ s is made small in order to have the fastest evaluation time for a generation. Having more evaluation time means also better statistical values for the individuals that keep appearing. (2) This rule prevents inefficient individuals that have obtained an accidentally good evaluation to stay in the top of the ranking and contaminate the next generation. Option (c) ensures that the first generation contains a large spectrum of promising control laws.

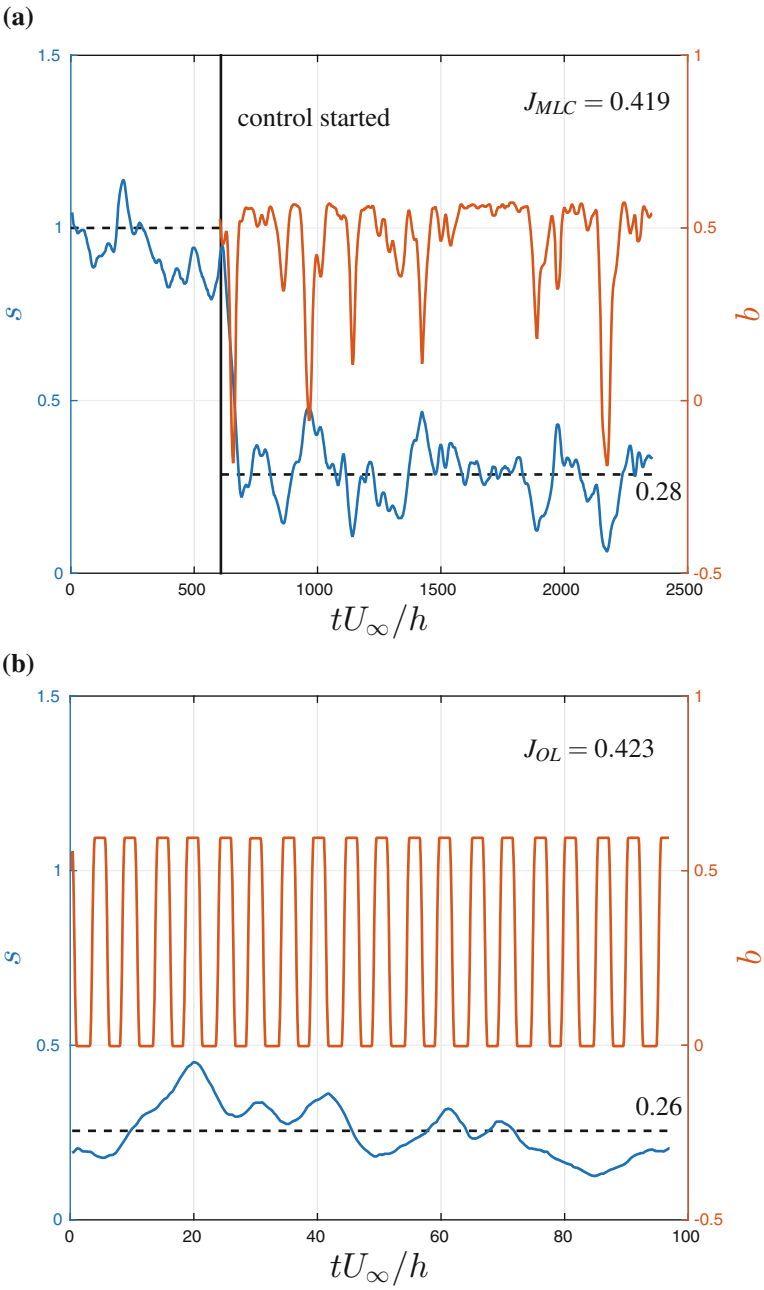


Fig. 6.2 Sensor and control command for the control of the PMMH backward-facing step flow by MLC (**a**) and best periodic forcing (**b**)

Table 6.2 Cost function of MLC, J_{MLC} and open loop J_{OL} at different Reynolds numbers

Re_h	J_{OL}	J_{MLC}
900	0.75	0.33
1350	0.42	0.42
1800	0.76	0.59

6.1.3 Results

After 12 generations, MLC returned the best control law. Its behavior is illustrated in Fig. 6.2a. When the control starts, the sensor value s goes from an average of 1 (by definition) to an average of 0.28, which corresponds to more than 70 % reduction. The other curve presented in the figure illustrates the control command providing a visual estimate of the actuation cost of the control law. Though it appears that most of the time the control command is set at $b = 0.5$, this individual has a cost of $J_{MLC} = 0.419$ while the best open-loop command, a periodic command at 1 Hz (corresponding to a Strouhal number of 0.2, displayed in Fig. 6.2b) has a cost of $J_{OL} = 0.423$ as defined by Eq. (6.2).

These comparable cost function values show that MLC has been able to find a control law which is as good as the best periodic forcing. A spectral analysis of the control command and the sensor signal under optimal MLC-based forcing [113] shows that the frequency which is most amplified by the shear layer is neither noticeable in the sensor signal nor exploited in the actuation command. The study suggests that MLC has found a novel way to act directly on the recirculation bubble with frequencies on the order of one tenth of the Kelvin–Helmoltz instability. This is compatible with the so-called flapping frequency of the bubble [254].

Furthermore, the resulting control law is closed-loop by construction. Thus, it should add intrinsic robustness to changing flow conditions since the control command is decided by flow events and not by a predetermined signal. Both MLC and optimal periodic forcing have been tested under off-design conditions, using the same cost function defined by Eq. (6.2). The results are displayed in Table 6.2. Unsurprisingly, the periodic forcing performs poorly on off-design conditions. In comparison, the MLC law performs better (in terms of minimizing the cost J) at lower and higher Reynolds numbers that were not included in the learning process.

This constitutes a key highlight of the experiment: MLC has found a new unexpected actuation mechanism which is more robust against changing operating conditions than the optimal periodic forcing. Intriguingly, closed-loop control has also outperformed periodic forcing in experiments with simple phasor control. Examples are mixing enhancement after a backward-facing step [209] and drag reduction of a d-shaped cylinder [210].

6.2 Separation Control of Turbulent Boundary Layers

In the second example, we also reduce a recirculation zone by closed-loop actuation. Yet, the geometry is a smooth ramp in which the separation point is not geometrically prescribed but physically controlled. In fact, MLC has been applied on two geometrically similar sister experiments in the Laboratoire de Mécanique de Lille, France, with Cécric Raibaudo, Christophe Cuvier and Michel Stanislas and in the Laboratoire PRISME from Orleans, France, with Antoine Debien, Nicolas Mazellier and Azeddine Kourta. Both experiments feature a turbulent boundary layer which separates under the influence of an identical descending ramp. Both experiments use vortex generators (though slightly different in geometry) as actuators and hot-films as sensors. The main difference between both experiments are (1) the Reynolds number, with a factor of 10 difference between experiments, and (2) the use of different cost functions.

6.2.1 Separating Boundary Layers

Every obstacle in ambient high-speed flow will generate a boundary layer on its surface. By definition, the boundary layer is a large gradient zone of the velocity between the no-slip condition at the wall and the outer potential flow. The force on the body is strongly dependent on the separation line which in turn depends on the state of the boundary layer. Hence, boundary layer separation is pivotal for the control of aerodynamic forces, like drag reduction behind a bluff body or lift increase of an airfoil. Flow separation gives rise to a shear layer, also called a mixing layer, between the slow recirculation zone and the fast outer stream.

This shear layer is prone to a Kelvin–Helmholtz instability, which generates large-scale spanwise vortices [191, 262]. Near the point of separation, the corresponding shedding frequency is characterized by a Strouhal number $St_\Theta = 0.012$ based on the boundary-layer momentum thickness Θ before separation [128, 281]. The vortices of the shear layer shed with a Strouhal number $St_{L_{sep}} = 0.6 - 0.8$ where L_{sep} is the separation length [62, 75, 182].

The shear layer separates the oncoming flow from the recirculation bubble extending from the separation line to the reattachment line. The reattachment point is determined by the efficiency of mixing between the high-speed oncoming flow and the low-speed recirculation bubble. The recirculation bubble is typically associated with low pressure region which increases the drag of the body. Large recirculation regions also reduce the lift force on wings.

Flow control can mainly target two mechanisms to manipulate this flow: (1) change the kinetic energy level of the boundary layer to prevent/promote separation, and (2) change the mixing properties of the separated shear layer to prevent/promote reattachment. The first mechanism is achieved by blowing/sucking the boundary layer or by using vortex generators. The second mechanism is achieved by manipulat-

ing the mixing layer. For instance, introducing perturbations at a sensitive frequency may excite vortex shedding which promotes an earlier reattachment. The use of unsteady vortex generators (UVG) enables the exploitation of both mechanisms: the streamwise vortices re-distribute the kinetic energy in the separating boundary layer, while the unsteadiness promotes vortex formations in separated shear layer. These types of actuators have been used extensively for the control of boundary layers. The optimal parameters for control such as the geometry, position and frequencies are still widely discussed [8, 120, 121, 123, 173, 244].

6.2.2 Experimental Setups at LML and PRISME

The experiments have been carried out in the Malavard closed-loop wind tunnel at the PRISME laboratory [79] and the closed-loop LML wall-turbulence wind tunnel [74]. The AVERT profile used in both experiments is the descending ramp as detailed in [74] for the LML wind tunnel. It features a sharp edge to force the position of the separation line (see Fig. 6.3). The initial slant angle of the ramp is 25°, and can be characterized by its height h_{ramp} and length ℓ. The Malavard wind tunnel has a 2 m wide and 5 m long cross section and the height and length of the ramp are $h = 100$ mm and $\ell = 470$ mm, respectively. With a free-stream velocity of $U_\infty = 20$ m/s, the Reynolds number $Re_h = U_\infty h_{\text{ramp}}/\nu$ is around 1.3×10^5, where ν is the kinematic viscosity of air. In the LML wind tunnel, the Reynolds number is ten times smaller due to lower velocities and smaller geometry.

For control purposes, unsteady vortex generators (UVG) have been implemented one boundary layer thickness upstream of the sharp edge ramp (see Fig. 6.3). Their design, location and orientation have been chosen based on the results from [73, 121, 248]. The UVG are set up so that the vortices are co-rotating in the LML wind tunnel, and counter-rotating in the PRISME wind tunnel. The jet velocity ratio is $V_{jet}/U_\infty = 3$. The jets are made unsteady by the use of identical electro-valves which can operate in an on/off fashion up to 300 Hz.

In both experiments, the friction downstream of the separation line is monitored by hot-films (Fig. 6.3). Additionally, unsteady pressure taps are available in the PRISME experiments for the computation of the cost-matching function. In both cases, PIV is used as a post-processing tool to assess the effect of the control on the flow. Unlike the water-tunnel experiment, PIV can neither be used in real-time nor in the learning loop. The feedback sensors s_i used for the MLC control law are based on the hot-film signals:

$$s_i = \frac{h_i - h_{i,u}}{h_{i,\max} - h_{i,u}} \quad \text{with} \quad i = A, B, \tag{6.3}$$

where h_i is the raw voltage output of sensor i, $h_{i,u}$ is the average voltage for the uncontrolled case (corresponding to a separated flow and low friction) and $h_{i,\max}$

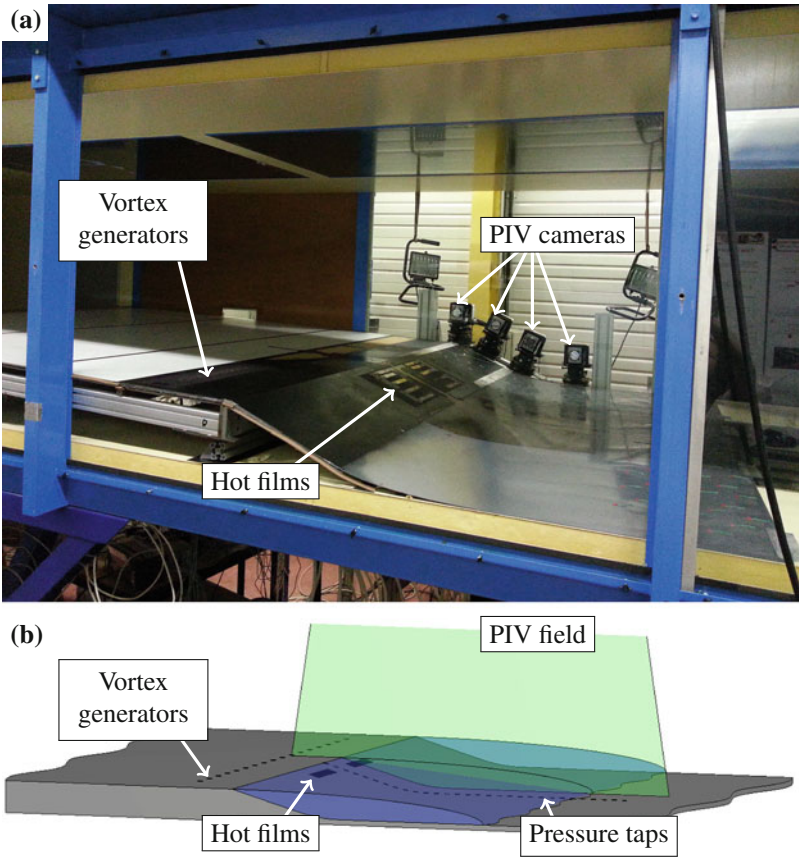


Fig. 6.3 (**a**) Photograph of the test section of the LML wind tunnel. (**b**) Experimental configuration of the separating boundary layer in the PRISME wind tunnel. The jets are placed to generate counter-rotating stream-wise vortices. Hot-film sensors are placed after the separation line and static pressure sensors are located in the symmetry plane

is the average voltage for the most effective constant blowing leading to maximal friction.

The control law is subject to a cost function promoting a reduced recirculation zone and penalizing blowing:

$$J = J_{\mathrm{HF}} + \gamma_{\mathrm{p_{stat}}} J_{\mathrm{p_{stat}}} + \gamma_{\mathrm{act}} J_{\mathrm{act}}, \tag{6.4}$$

with J_{HF} being an evaluation of the friction recorded from the hot-films, $J_{\mathrm{p_{stat}}}$ an evaluation based on the static pressure distribution and J_{act} an evaluation of the actuation cost. $\gamma_{\mathrm{p_{stat}}} = 1/200$ and $\gamma_{\mathrm{act}} = 0.6$ are chosen as penalization coefficients. The evaluation based on the friction is defined as:

$$J_{\mathrm{HF}} = \frac{1}{N_{\mathrm{HF}}} \sum_{i=1}^{N_{\mathrm{HF}}} \left[1 - \tanh\left(\langle s_i \rangle_T\right)\right], \tag{6.5}$$

where $N_{\mathrm{HF}} = 2$ is the number of hot-film sensors. The value of J_{HF} is 1 when no effect is recorded and approaches 0 as the friction increases. The evaluation based on the static pressure is defined as:

$$J_{\mathrm{p_{stat}}} = \frac{1}{0.1 + \sum_{i=1}^{14} \langle (p(x_i,t)\rangle_T - \left\langle p_u(x_i,t))^2 \right\rangle_T \frac{x_{\max} - x_i}{x_{\max} - x_{\min}}}, \tag{6.6}$$

with x_i being the position of the ith pressure tap after the edge, $x_{\min}$ the position of the pressure tap closest to the edge, $x_{\max}$ the furthest downstream pressure tap, $p(x_i, t)$ the static pressure recorded at position x_i and $p_u(x_i, t)$ the static pressure recorded at position x_i in the uncontrolled case. $\langle\cdot\rangle_T$ is the average over the evaluation time T. $J_{\mathrm{p_{stat}}}$ is equal to 10 when controlled and uncontrolled pressure distributions coincide and approaches zero when they significantly deviate from each other, with a weight which increases linearly towards the separation point. The LML experiment has no pressure taps and this term is not taken into account.

The evaluation of the actuation cost is defined by:

$$J_{\mathrm{act}} = \begin{cases} \left\langle \frac{Q}{Q_u} \right\rangle_T & \text{at PRISME,} \\ \left\langle b^2 \right\rangle_T & \text{at LML,} \end{cases} \tag{6.7}$$

where Q is the flow-rate and Q_u the flow-rate under constant blowing. In the LML experiment the flow rate could not be integrated in the learning loop and the control command was used instead. In both cases, J_{act} is equal to 1 for constant blowing and vanishes when no actuation is recorded. The parameters of the MLC algorithm are summarized in Table 6.3.

Table 6.3 MLC parameters used for the control of the PRISME and LML separating boundary layer

Parameter	Value (PRISME)	Value (LML)
$\gamma_{\mathrm{p_{stat}}}$	1/200	0
γ_{act}	0.6	2
N_i	100	500
P_r	0.1	0.1
P_m	0.2	0.25
P_c	0.7	0.65
N_p	7	7
N_e	1	1
Node functions	$+, -, \times, /$, exp, log, tanh	

6.2.3 Results

Both experiments led to successful separation mitigation of the boundary layer such as displayed in Fig. 6.4. In the case of the LML experiment, the best open-loop control reference is a constant blowing of the UVG at $V_{jet}/U_{\infty} = 3$ leading to $J_{OL} = 3$. MLC achieves a control with similar performance but at a reduced actuation cost leading to $J_{MLC} = 2.1$ (Fig. 6.5). In this case, γ_{act} directly selects the operating point of the system as the dominating mechanism seems to be strongly linked to the kinetic energy injection in the boundary layer. The mechanism can be summarized as follows: the more one blows, the more the boundary layer attaches. Vanishing penalization leads to constant blowing while high penalization leads to the unforced state.

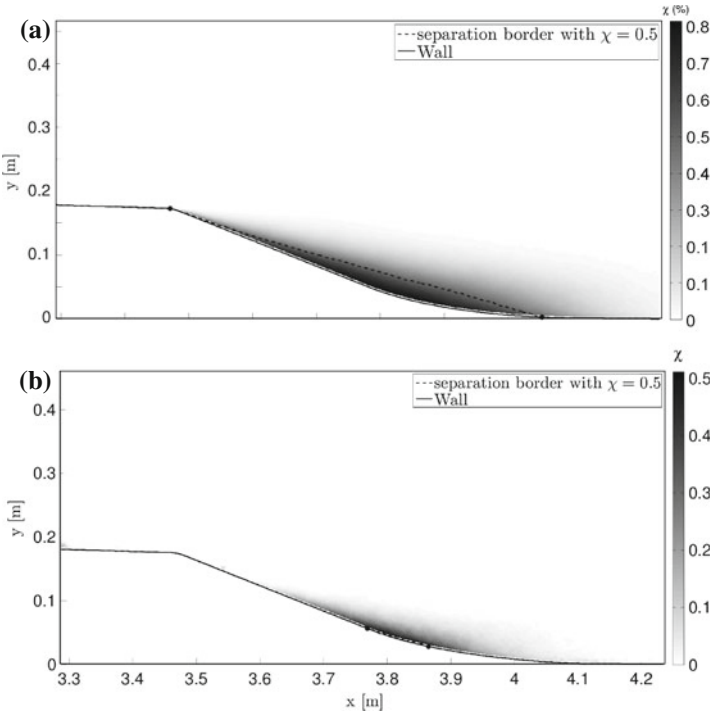


Fig. 6.4 Cartographies of back-flow coefficient χ in the LML experiment. This coefficient is defined in analogy of the intermittency factor as the average percentage of the occurrence of a negative streamwise velocity at the considered point [250]. **(a)** Uncontrolled flow. **(b)** MLC modified flow. The recirculation zone has been drastically reduced. For both cases the iso-line at $\chi = 50\,\%$ has been traced

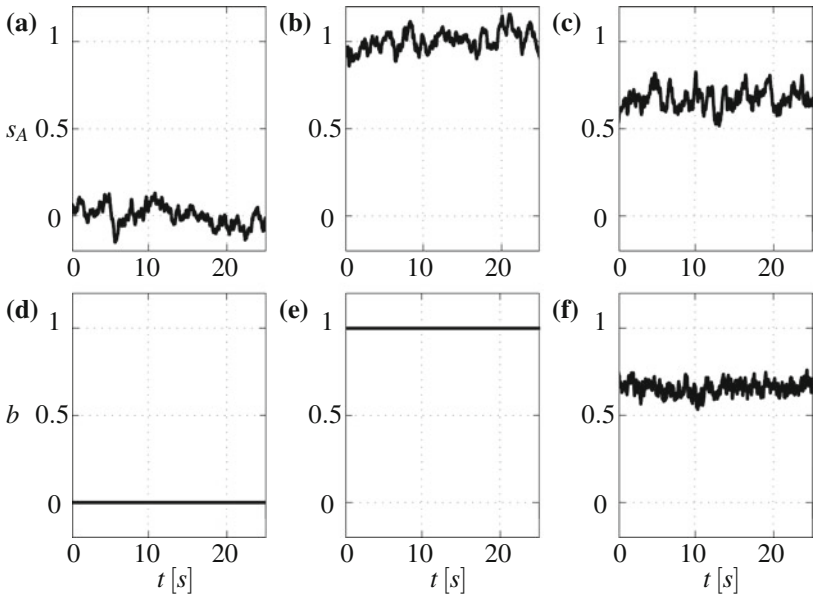


Fig. 6.5 Time series of sensor A (**a**–**c**) and low-pass filtered control signal (**d**–**f**) for uncontrolled (**a**, **d**), open-loop constant blowing control (**b**, **e**) and MLC control (**c**, **f**) in the LML experiment. The level of friction obtained appears to be proportional to the amount of kinetic energy injection

This dominance of the kinetic energy injection has also been demonstrated in the LML experiment [80]. In that case, the pressure distribution obtained in controlled cases is compared to one of the baseline flow in Fig. 6.6a. Both constant blowing and MLC schemes lead to a reduction of the mean recirculation region since the recovery region associated with the pressure plateau is shifted upstream. Noticeable is the acceleration of the flow induced by the UVGs upstream of the sharp edge location ($x/h_{\text{ramp}} = 0$) as emphasized by the strong decrease in pressure. However, pressure distributions computed for the best open-loop actuation and MLC almost coincide. This implies that the efficiency of both control approaches are approximately equivalent. This is confirmed by the measurement of the separation length L_{sep} from the PIV dataset which is reduced by about 40 % when control is applied (see Table 6.4). Nevertheless, the actuation cost to achieve the same recirculation zone reduction is significantly lower ($\approx 20\%$) for MLC as evidenced by the momentum coefficient

$$c_\mu = \frac{S_j d_c V_j^2}{1/2 S_{\text{ref}} U_\infty^2},$$

where S_j represents the cross section of the jets, d_c the duty cycle and S_{ref} the surface of the ramp (defined by the flow exposed surface of the descending part of the ramp) reported in Table 6.4.

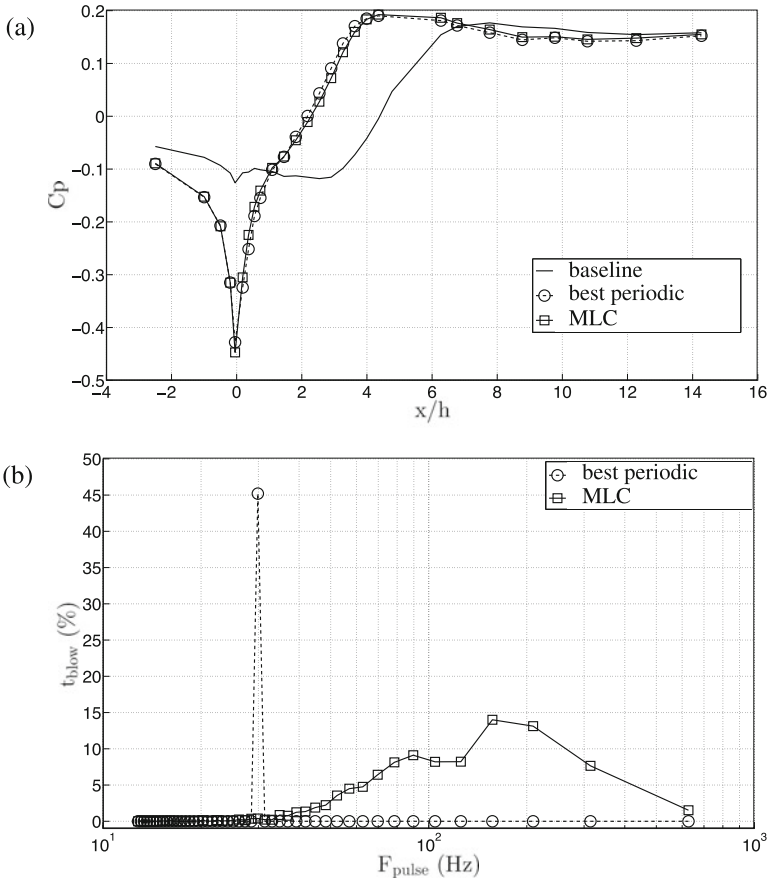


Fig. 6.6 PRISME experiment: (**a**) Pressure distribution along the ramp. (**b**) Frequency distribution of the blowing

Table 6.4 Cost function values, separation and actuation properties for the natural, best open-loop and MLC cases in the PRISME separating boundary layer experiment

Case	Natural	Open-loop	MLC
J	50.4	0.291	0.32
L_{sep}/h_{ramp}	5.4	3.14	3.16
c_μ ($\times 10^{-4}$)	–	16.51	13.66

Intriguingly, the reference open-loop control and MLC are distinctly different in terms of the frequency distribution of the actuation command. The frequency distribution is computed from a zero-crossing algorithm applied to the mass-flow controller signal (see Fig. 6.6b). Unlike open-loop periodic forcing for which the blowing frequency is fixed at 30 Hz, the frequency distribution of the MLC is broadband. More surprisingly, the frequencies of MLC are significantly larger than for the best periodic forcing. Thus, MLC and periodic forcing yield similar separation mitigation by significantly different underlying actuation mechanisms. A more detailed study of the flow physics is reported in [80].

6.3 Control of Mixing Layer Growth

We finally present a feedback control of a turbulent mixing layer built and operated in the ANR Chair of Excellence 'TUrbulence COntrol using Reduced-Order Models (TUCOROM)' (ANR-10-CHEX-0015) at Institute PPRIME, France. This experiment comprises all feedback control challenges described in Chap. 1: The complex vortex dynamics between actuation and sensing leads to high-dimensional dynamics, strong nonlinearity and large time delays. The goal of this experiment is to optimize the mixing between both sides of the shear layer. To date, this is arguably the most challenging implementation of MLC in an experiment.

6.3.1 Mixing Layer Flows

Mixing layers or, equivalently, shear layers, arise when two streams of flow at different velocities interact. Such mixing layers can be observed in almost every flow. The recirculation zone of bluff-body wakes is bounded by shear layers. The near-field dynamics of a jet is determined by surrounding mixing layers. Any separation phenomenon, e.g., in a diffuser or an airfoil, leads to mixing layers. The convectively unstable dynamics of these flows is particularly rich, as the streamwise evolution is a noise amplifier of upstream perturbations with large frequency bandwidth. The Kelvin–Helmhotz instability leads to large-scale vortex formation in laminar and turbulent mixing layers [134]. These vortices merge in the streamwise direction to form larger and larger structures which decay in a turbulence cascade to increasingly small vortices. Thus, the mixing layer leads inevitably to high-dimensional, multi-scale vortex dynamics. Already by phenomenology it is clear that the possibility of low-dimensional or linear models are very limited as will be corroborated in Sect. 6.4.

Nonetheless, simple feedback flow control strategies have been successfully applied to mixing layers [210]. From early periodic forcing studies [134], it is observed that actuating at the frequency of the main structures reinforces said structures and invigorates the mixing, while actuation at much higher frequencies interferes with the production of large structures and mitigates the width of the shear layer.

6.3.2 Experimental Setup of the TUCOROM Wind Tunnel

The TUCOROM experiment has a double turbine wind tunnel aimed at creating a turbulent mixing layer. The wind tunnel generates two flows that can be set at different velocities and which are separated by a plate. The optically accessible test section has a cross section of 1×1 m^2 and a length of 3 m (Fig. 6.7). The separation plate ends with a triangular profile with a tip 3 mm thick which includes 96 holes for actuation jets blowing in the streamwise direction. The jets can be operated independently. A vertical rake of 24 hot wires, operated at 20 kHz, serves as a sensor array and can be placed at any position downstream in the test-section. The hot wires, vertically separated by a 4 mm offset, map the mixing layer profile. The closed-loop control is implemented on a Concurrent® RT system which combines the use of Simulink® loop design and Fortran home-made adaptor functions to encode sensor acquisition, compute control decisions and command actuation signals up to a maximum rate of 10 kHz.

The unforced and forced mixing layers are described in detail in [206, 207]. For the results presented here, the wind tunnel is operated with a velocity ratio of $U_1/U_2 = 3.6$, and the Reynolds numbers based on the initial mixing layer thickness is 500 (laminar) for the learning process and can be set to 2000 (turbulent) to test off-design conditions. The hot-wire rake is placed at $x = 500$ mm downstream from the separation plate. The sensors are based on the fluctuations of the raw hot-wires sensors:

$$s_i(t) = h_i(t) - \langle h_i(t) \rangle_{T_{rms}}, \tag{6.8}$$

where $h_i(t)$ is the raw velocity measured at hot-wire number i and $T_{rms} = 1$ s is the time interval used in order to compute the moving average of the velocity signal.

The goal of the control is to increase mixing. In this chapter, we chose the width of the shear layer profile as a quantity to maximize. This width is approximated by the ratio of the integral of the average fluctuation energy profile over the maximum of this profile. Thus, the cost function reads:

$$J = 1/W \quad \text{with } W = \frac{\sum_{i=1}^{24} \left\langle s^2(t) \right\rangle_T}{\max_{i \in [1,24]} \left(\left\langle s_i^2(t) \right\rangle_T \right)}, \tag{6.9}$$

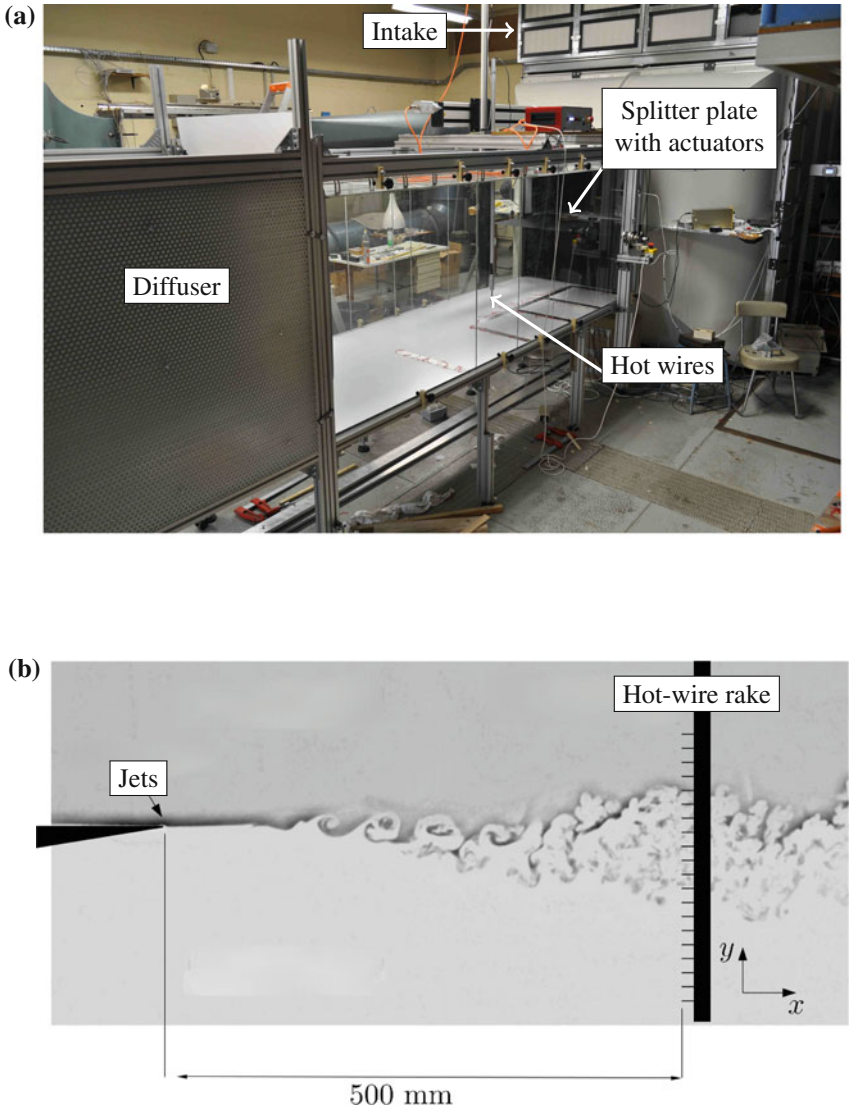


Fig. 6.7 TUCOROM mixing layer demonstrator: (**a**) Photo of the test section. (**b**) Experimental setup of the mixing layer. The hot-wire rake is placed at 500 mm downstream of separating plate to capture the structures in the shear layer. The spacing of the hot-wire probe is $\Delta y = 8\,\text{mm}$

Table 6.5 MLC parameters used for the control of the TUCOROM mixing layer

Parameter	Value
N_i	1000 (first generation)
	100 (other generations)
P_r	0.1
P_m	0.25
P_c	0.65
N_p	7
N_e	1
Node functions	$+, -, \times, /$, sin, cos, exp, log, tanh

where $T = 10$ s is the evaluation time for a control law. This time corresponds to approximately 950 Kelvin–Helmoltz periods of the initial shear layer.

The 96 actuators are operated simultaneously as a single input, although they could have been actuated independently. Studies with periodic forcing of different actuator configurations have indicated that this synchronous operation is best for enhancing the mixing layer width. The control law should only return a binary value (0 or 1) as the valves commanding the jets can only be operated in an on/off mode. The maximum jet velocity is kept constant by the pressure tank. Details on the implementation of the control law by MLC are given in the next section. The parameters used for the operation of MLC in this experiment are detailed in Table 6.5.

The output of the expression trees can be an arbitrary function of the sensors $s_i(t)$, which could take any value in $\mathbb{R}$. In our case, the valves commanding the actuation jets can only be opened or closed. Hence, the output of the trees is compared to zero, and thus $b(t) = 0$ if the output of the tree is negative, and $b(t) = 1$ otherwise. This operation is achieved inside the Fortran part of the RT loop, after interpretation of the expression trees, and before sending back the actuation command to the Labview® framework (see Sect. 6.5.4).

Evaluating an individual costs 3 s transient time and $T = 10$ s evaluation time. Thus the first generation with 1000 individuals requires around 3.5 h and subsequent generations require around 20 min each. This represents around 8–10 hours for a typical run with 10–15 generations. The reader is strongly advised to implement health monitoring of the experiment and simple automatic reset procedures.

6.3.3 Results

Though the experiment was run until the 15th generation, the best control law was already obtained after the 8th generation. The effect of the MLC-controlled flow is shown in Fig. 6.8—together with the unactuated flow and the best periodically

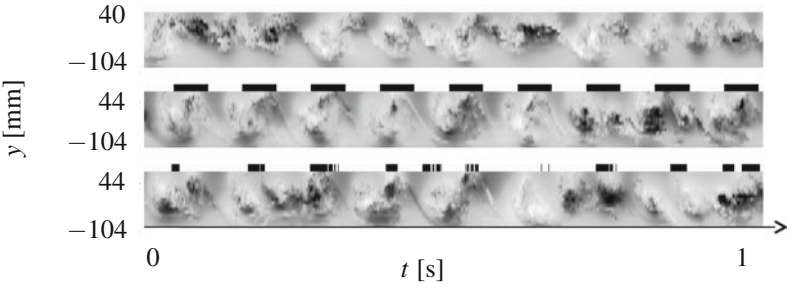


Fig. 6.8 Pseudo-visualizations of the TUCOROM experimental mixing layer demonstrator [206] for three cases: natural baseline (*top*, width $W = 100\,\%$), the best periodic-forcing benchmark (*middle*, width $W = 154\,\%$), and MLC closed-loop control (*bottom*, width $W = 170\,\%$). The velocity fluctuations recorded by 24 hot-wires probes (see Fig. 6.7) are shown as contour-plot over the time t (abscissa) and the sensor position y (ordinate). The *black stripes* above the controlled cases indicate when the actuator is active (taking into account the convective time). The average actuation frequency achieved by the MLC control is comparable to the periodic-forcing benchmark

forced flow. The periodic-forcing benchmark has been obtained through an extensive parametric study of harmonic forcing with many frequencies ranging from 1 to 500 Hz and various duty cycles. A periodic forcing at frequency of 9 Hz and 50 % duty cycle was found to yield the largest increase in mixing-layer width W of a 54 %. In terms of performance, the MLC controlled flows exhibits a 70 % increase of the shear-layer width, which outperforms the periodic-forcing benchmark by 16 % with respect to this width or 30 % with respect to increase of the width by periodic forcing. Looking back at Fig. 6.8, the similarity between the periodic forcing and the effect of the MLC law is evident. A spectral analysis of the MLC control command shows that MLC also operates around 9 Hz. Both actuations amplify the large-scale eddies at the downstream measurement position. The difference between both optimal periodic forcing and MLC is fourfold:

1. MLC is a sensor-based closed-loop control and does not explicitly depend on time. In other words, MLC schedules actuation based on flow events and not on a predefined 'clockwork'.
2. The shear-layer width associated with MLC actuation is significantly larger.

Advanced material 6.2 Varying the population size to minimize the time of the runs.
The MLC of this experiment follows the experimental changes of advanced material 6.1. In addition, the first generation contains 1000 individuals while the following ones are restricted to 100 individuals. The large first generation (with removal of uninteresting individuals) ensures that the search space contains effective control laws. The significantly reduced size of further generations reduces the time investment in an inefficient exploration.

3. The effective duty cycle exhibited by MLC is much smaller, leading to a 48 % decrease in the actuation cost. Intriguingly, this benefit arises although actuation was not penalized in the cost function (6.9).
4. The actuation effect of MLC dramatically outperforms periodic forcing for changing operating conditions, such as changing the maximum stream velocity. The situation is comparable to the PMMH experiment in Sect. 6.1 or drag reduction of the D-shaped body with phasor control [210]-

Feeding back sensor signals is particular important to optimize the control law. The periodic forcing acts like a clockwork on the noise amplifier and performs well as long as no perturbations are modifying the underlying base flow. On the other hand, MLC triggers the actuation based on what is sensed. MLC adapts with a time delay of the order of $2 \times x/(U_1 + U_2)$ which is comparable to the period of maximum excitation. Thus, MLC realizes a phasor-type control as in [210] which is more 'in phase' with the flow physics and more robust to changing operating conditions.

6.4 Alternative Model-Based Control Approaches

To motivate the use of machine learning control using genetic programming, we consider the inability of linear system identification to capture the strongly nonlinear dynamics in the TUCOROM mixing layer experiment from Sect. 6.3. In particular, we attempt to identify a linear input–output model from actuation to hot-wire sensors using the ERA/OKID algorithms described in Sect. 3.5.1.

As mentioned earlier, it is often difficult to obtain clean impulse response experiments, especially in turbulent fluid systems with stochastic fluctuations. Instead, a pseudo-random actuation sequence is constructed by repeatedly turning the jets on or off for a random interval of time. The hold time is sampled from a discrete Poisson distribution with parameter $\lambda = 4$. This distribution is then scaled so that the mean hold time is 0.05 s. Using pseudo-random excitation sequences for system identification of fluid systems is inspired by [158]. The actuation sequence is shown in Fig. 6.9 (top).

A particular pseudo-random blowing sequence is used for 300 identical experimental runs, and the sensor measurements from a rake of nineteen hot wires at a downstream location of $x = 500$ mm are collected and phase averaged. The resulting phase-averaged velocity measurements are shown in Fig. 6.9 (bottom). Immediately, coherent structures may be observed as correlations in the hot-wire signals; however,

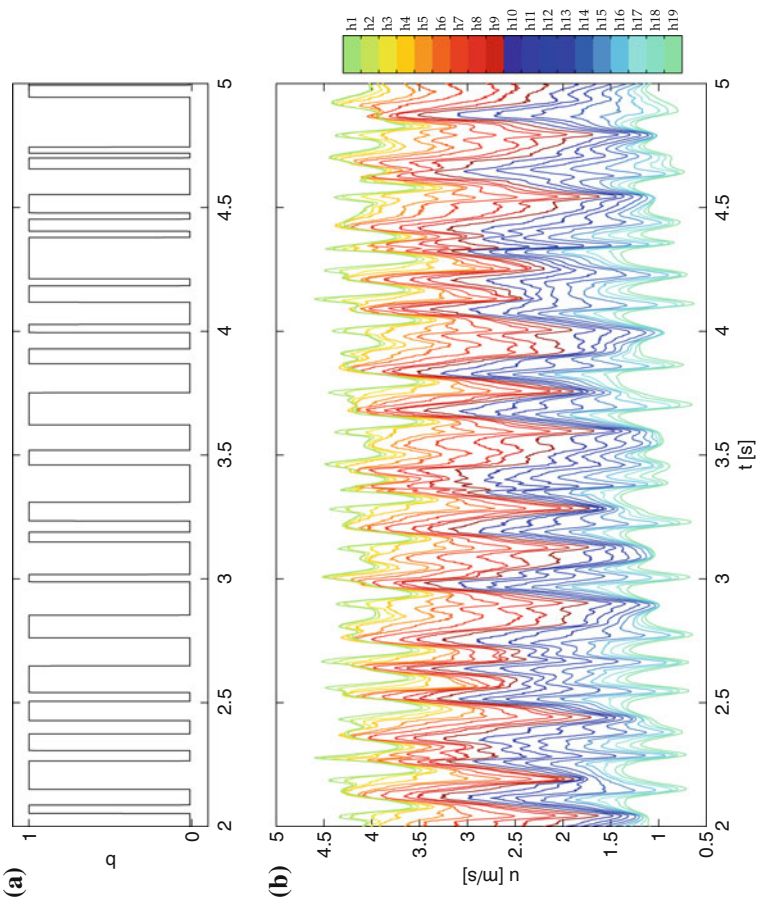


Fig. 6.9 (**a**) Pseudo-random actuation sequence for system identification and (**b**) velocity measurements from hot-wire rake. The actuation sequence consists of Poisson blowing at the leading edge of the splitter plate with average hold duration of 0.05 s. Nineteen hot-wire probes are used at a downstream location of $x = 500$ mm

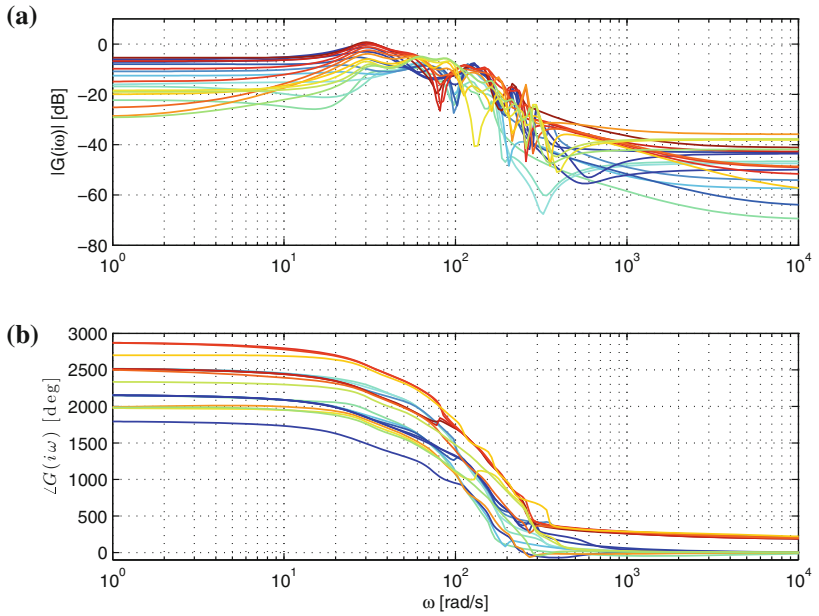


Fig. 6.10 The ERA/OKID models for the input–output response of each hot wire may be combined to produce a frequency response plot. The color code indicates the corresponding hot wire and is adopted from Fig. 6.9. The phase information (**b**) strongly suggests a time delay, while the magnitude plot (**a**) indicates weak flow resonances

phase-averaging removes important stochastic fluctuations that account for a significant amount of the kinetic energy. Therefore, even perfect model reconstruction of the phase-averaged velocity measurements would only account for a limited aspect of the full nonlinear flow.

The phase-averaged measurements are then used in the ERA/OKID algorithm to obtain input–output models for the various hot-wire signals. A frequency response of the various models for the nineteen hot-wire signals is shown in Fig. 6.10. The magnitude plot shows moderate flow resonance at certain frequencies, and the phase plot captures the time delay between actuation and measurements. However, when analyzing the model reconstruction of a particular hot wire in Fig. 6.11, it is clear that the model error is on the same order of the signal strength. This indicates that the phase-averaged measurements still contain strong, reproducible nonlinear flow effects that cannot be captured by the linear models.

To summarize, linear model identification fails to capture two features of the mixing layer flow field: stochastic fluctuations and coherent nonlinear phenomena. Taken together, these effects account for a significant portion of the kinetic energy, and are likely important for flow control efforts. However, it may still be interesting to test the performance of robust model-based control, considering that many nonlinearities may be considered as model uncertainty or disturbances.

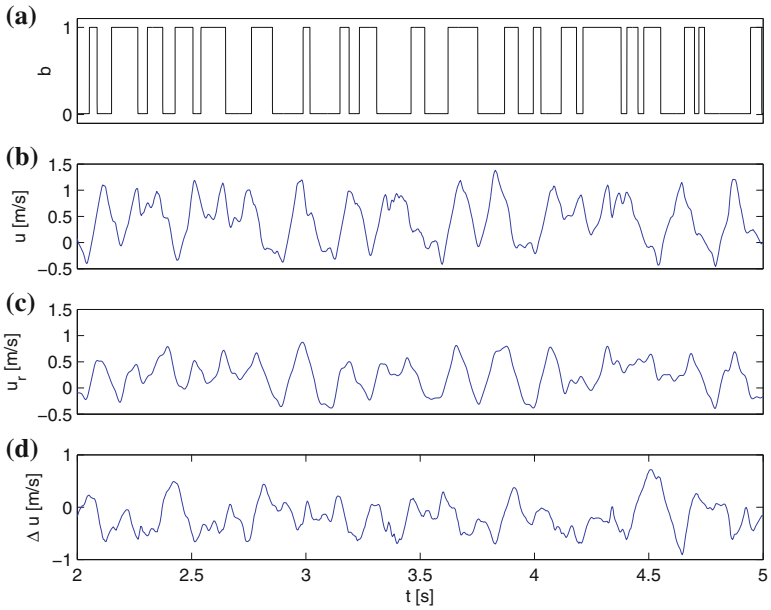


Fig. 6.11 The measured hot-wire signal u is compared with the model-reconstructed signal u_r for the 9th hot wire near the middle of the rake. Note that the error Δu is nearly as large as the original signal

6.5 Implementation of MLC in Experiments

This section details the software and hardware implementation of the experiments described in the previous sections. Following Chap. 2, two closed loops need to be set up: a fast inner real-time (RT) control loop (Sect. 6.5.1) and a 'slow' outer MLC learning loop. Section 6.5.1 describes the real-time control loop which extracts information from the experiment through sensors, computes the actuation command and passes it to the actuator. Details about the hardware and software implementation of MLC for the PMMH, PRISME and TUCOROM experiments are provided in Sects. 6.5.2–6.5.4, respectively.

6.5.1 Real-Time Control Loop—from Sensors to Actuators

Any experimental RT control loop consists of four elements: the plant, sensors, actuators and a computational unit. The sensors get information about the plant. The computational unit processes these signals and sends a control command to the

actuators which, in turn, will act on the plant. All of these elements must be able to operate in a frequency range of the actuation mechanism at work. Stabilizing an unstable oscillator, for instance, typically requires a sampling rate which is significantly larger than the eigenfrequency. In fact, the Nyquist theorem would impose the need for a sampling frequency which is at least twice that frequency and a rule of thumb is to use a sampling frequency of at least ten times the eigenfrequency.

The RT loop constitutes a classical feedback experiment for a given controller. The slow outer MLC loop provides new controllers for this experiment after each cost function evaluation. This implies that the computational unit needs to be programmable, either by allowing it to read and interpret the individuals provided by the learning loop (at every time step), or by allowing the learning loop to change its content. Two approaches have been used to achieve this goal:

- The outer loop is able to change (and compile) the controller code on the RT-loop computation unit at least once per generation (see Sect. 6.5.3 as an example).
- The use of an on-board control law parser: the controller implemented in the computational unit is re-loaded every sampling cycle and could thus change even during one evaluation period (see Sect. 6.5.4).

As the control law provided by the MLC learning loop is arbitrary, the RT loop has to be protected against control commands that provide values outside the safe operation range of the experiment. This means that the controller—implemented as a parser or as an externally modified part—should be encapsulated in a framework that can protect the actuators from operating out of safety margins.

The last aspect to be taken into account is that the experiment may evaluate hundreds or thousands of control laws and may hence take more time than a standard control experiment where only a single control law is tested or optimized. The time scales for MLC experiments may range from hours in wind tunnels to days in water tunnels. Hence, all components of the experiment should be able to operate in an unsupervised manner at least to evaluate a few generations. This might imply the addition of other regulation loops for temperature, calibration updates, tank refill, to name only few examples. It is also advisable to design some telltales in order to detect any unusual behavior of the experiment. Feedback loops have a reputation of pointing at weak spots of experiments and this is amplified by the exploratory nature of the MLC algorithm.

6.5.2 MLC Implementation in the PMMH Flow Over a Backward-Facing Step

In the water tunnel, the characteristic frequency is about 1 Hz. With such a low characteristic frequency, the controller can be implemented on any modern

computer. The real-time loop, including the RT PIV part, is implemented in Labview®. Thus, the Labview® project contains the real-time PIV acquisition, the command signal to the actuator and the controller. The outer MLC loop employs `OpenMLC` introduced in Sect. 2.3.

The control law is first translated from LISP to a classical symbolic expression by `OpenMLC` then transmitted through file exchange on the computer hard-drive to the Labview® project. Each time `OpenMLC` evaluates an individual, it writes a file (e.g., `ind.dat`) containing the symbolic expression describing the control law. The appearance of the file is detected in one of the RT-loop iterations. The file is read, the string variable containing the expression is then updated and the file is deleted. A parser function is then used to transform the sensor value to a control command inside the controller block.

That control law is used during the evaluation period and at the same time the cost function is integrated. The Labview® program then writes the cost function value of the individual in a file (e.g., `J.dat`) in the exchange directory of the computer. `OpenMLC` detects the presence of the file, reads and deletes it, which tells Labview® that the transmission is complete. The next individual can then be evaluated in the same way.

The exchange by files may sound like an unnecessary and time-consuming way to proceed. Yet it is relatively simple, as only the read/write protocol needs to be implemented. The file exchange requires that a parser can be implemented on the RT loop.

6.5.3 MLC Implementation in the LML and PRISME Experiments

In the LML and PRISME experiments, no dedicated, high-performance and highly expensive computer with real-time capacities was available. The RT loop has instead been implemented in an Arduino micro-controller (Fig. 6.12). In contrast to the water-tunnel experiment in Sect. 6.5.2, a parser on the Arduino would have led to a significant performance loss. The water-tunnel experiment at PMMH had a large characteristic time scale of 1 s and the communication between RT and MLC loop happened over file exchange. The time scales of wind-tunnel experiments are one to two orders of magnitude faster and we chose a more efficient communication strategy. `OpenMLC` has been set up so that (1) the evaluation function generates code for the Arduino with all control laws for one generation, (2) compiles it, and (3) burns it on the micro-controller. By abandoning the possibility of changing the individual on the fly, the computational load achieved on the board has been reduced to a bare minimum. This speed-up enables a control update frequency of 1 kHz without even having to implement low-level programming on the Arduino.

The burned code contains all individuals of one generation and sends back the cost function values. The outer loop is controlled by Matlab® code which generates

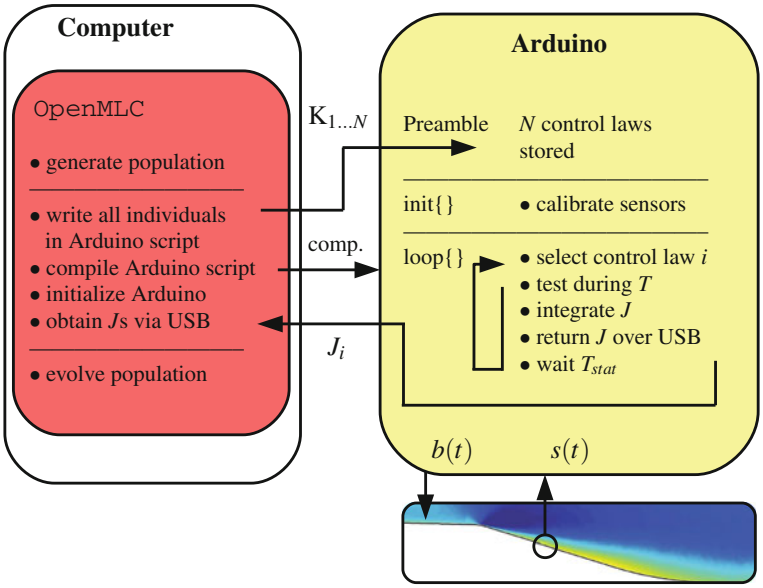


Fig. 6.12 MLC implementation architecture in LML and PRISME boundary-layer experiments

the codes for the individuals and retrieves the cost function values in addition to reading any other information which may be fed back by the Arduino. This code can interrupt the experiment, realize a calibration, and re-burn and re-start the evaluation if needed.

6.5.4 MLC Implementation in the TUCOROM Experiment

The controller is implemented on a high performance PC with 1 TeraFlop computing power (Fig. 6.13). The real-time loop is designed in Labview® and compiled to run at the desired frequency enabled by a real-time scheduler. The reading of the sensors, the signal processing, and the calculation of the control command is achieved by a Fortran function whose output is then transferred inside the Labview® designed project. The Labview® project encompasses all other commands needed to operate the wind tunnel, from the turbine operation, pressure regulation for the jets, calibration, temperature monitoring, and so on.

The interaction between `OpenMLC` and the RT loop is achieved through file exchange as in the PMMH water-tunnel experiment (Sect. 6.5.2). This has the advantage of changing the control law on the fly. Here, the loop described by the Labview® project is in charge of monitoring the experiment and act accordingly in case of an unexpected event, such as failing actuators or sensors.

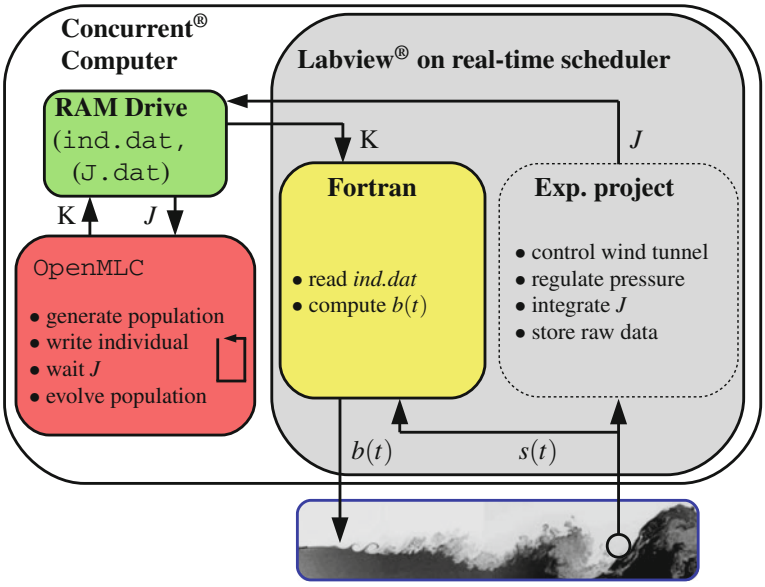


Fig. 6.13 MLC implementation architecture in TUCOROM mixing layer

Advanced material 6.3 LISP interpreter pseudo-code.
All LISP parsers operate in the same way and are rather easy to build, through a recursive function: function def:
translated_string=read_my_LISP(LISP_string):
(1) Detect outer parenthesis
if no parenthesis: translated_string = LISP_string
(2) Detect spaces
operator=LISP_string(par1:space1)
argument#n=LISP_string(space#n:space#n+1)
(3) Translate arguments
tr_args#n=read_my_LISP(argument#n)
(4) Replace strings
if op is a function
translated_string=operator(argument#1,…)
if op is an operation
translated_string=argument#1 operator argument#2

6.6 Suggested Reading

(1) **Modern developments in flow control**, by M. Gad-el-Hak, *Applied Mechanics Reviews*, 1996 [108].
This review provides an early perspective on flow control with a number of future directions that have since been developed.

(2) **Control of turbulence**, by J. L. Lumley and P. N. Blossey, *Annual Review of Fluid Mechanics*, 1998 [179].
This review is among the earliest overviews summarizing efforts on modeling for turbulence control.

(3) **Feedback control of combustion oscillations**, by A. P. Dowling and A. S. Morgans, *Annual Review of Fluid Mechanics*, 2005 [85].
This review considers combustion control using linear control theory.

(4) **Dynamics and control of high-Reynolds number flows over open cavities**, by C. W. Rowley and D. R. Williams, *Annual Review of Fluid Mechanics*, 2006 [232].
This review describes the control of flow over open cavities, which provides an illuminating success story of flow control design.

(5) **A linear systems approach to flow control**, by J. Kim and T. R. Bewley, *Annual Review of Fluid Mechanics*, 2007 [161].
This review provides a clear summary of modern techniques applying linear control theory to fluid flow control.

(6) **Control of flow over a bluff body**, by H. Choi, W.-P. Jeon, and J. Kim, *Annual Review of Fluid Mechanics*, 2008 [65].
This review considers the problem of controlling bluff body flows from physical and mathematical perspectives.

(7) **Optimal and robust control of fluid flows: Some theoretical and computational aspects**, by T. T. Medjo, R. Temam, and M. Ziane, *Applied Mechanics Reviews*, 2008 [186].
This review provides an overview of mathematical and numerical considerations in modern flow control.

(8) **Input-output analysis and control design applied to a linear model of spatially developing flows**, by S. Bagheri, J. Hoepffner, P. J. Schmid, and D. S. Henningson, *Applied Mechanics Reviews*, 2009 [13].
This review explores linear model-based control of fluid systems with the Ginzburg–Landau equation as an illuminating example.

(9) **Actuators for active flow control**, by L. Cattafesta, *Annual Review of Fluid Mechanics*, 2011 [54].
This review considers one of the most important factors in experimental flow control: the actuators.

(10) **Adaptive and model-based control theory applied to convectively unstable flows**, by N. Fabbiane, O. Semeraro, S. Bagheri, and D. S. Henningson, *Applied Mechanics Reviews*, 2014 [101].
This review provides an overview of research on active control for transition delay with example code for the linearized Kuramoto–Sivashinsky equation.

(11) **Analysis of fluid systems: stability, receptivity, sensitivity**, by P. J. Schmid and L. Brandt, *Applied Mechanics Reviews*, 2014 [239].
This review investigates flow analysis in terms of cost optimization.

(12) **Closed-loop turbulence control: Progress and challenges**, by S. L. Brunton and B. R. Noack, *Applied Mechanics Reviews*, 2015 [43].
This review explores the state-of-the-art of closed-loop turbulence control and provides and emphasis on modern methods from machine learning.

6.7 Interview with Professor David Williams

David Williams is Professor of Mechanical and Aerospace Engineering at the Illinois Institute of Technology, IL, USA. Since 1999 he has been the Director of the Fluid Dynamics Research Center at IIT. In addition, Professor Williams is an active member of the American Institute of Aeronautics and Astronautics, and is currently serving his third term on the fluid dynamics technical committee.

Professor Williams is a leading expert on experimental closed-loop flow control, with numerous experimental success stories on challenging flow configurations, such as the reduction of acoustic tones in aircraft cavities or separation control over an airfoil in flight maneuvers. He is a highly collaborative researcher and draws researchers from around the globe to work with him in his advanced wind tunnel facility.

Professor Williams has received numerous national and international honors for his research accomplishments and excellence in teaching. He is Fellow of the American Physical Society and was awarded with the prestigious Alexander von Humboldt Fellowship and the Honeywell Advanced Technology Achievement Award. He is also a licensed commercial pilot with more than 1000 h of flying time.

Authors: You are a leader in experimental flow control, with many successful demonstrations of closed-loop control for aerodynamic applications. It would be an understatement to say that there are issues that arise in experimental flow control that are not usually present in simulations and theory. Can you discuss key enablers in experimental flow control that have been developed in the last two decades? Which challenges do you see as guiding foci of relevant future research?

Prof. Williams: The "key enablers" for experimental active flow control (AFC) have come primarily from the adaptation of modern methods of closed-loop feedback control algorithms into experiments, and the rapid development of sensors and digital signal processing (DSP) hardware, e.g., dSPACE and Arduino DSP's. This combination enables AFC to become much more than an exercise in actuator development and open-loop control. It is now possible to adapt AFC to changing flight/flow conditions, and to interact directly with internal flow instabilities, which leads to far more robust and effective controllers than can be achieved with

open loop control or by changing a base flow state. With closed-loop active controllers the actuator power requirements can be minimized, which is not possible with open-loop methods.

The focus in experimental fluid dynamics is shifting from open-loop demonstrations of actuator performance at steady flow conditions to closed-loop control during unsteady flow conditions. Open-loop actuator demonstrations compete with passive flow control devices. For example, separation control on wings will likely continue to be done with passive vortex generators, because the active devices are more expensive and less reliable.

Future research in experimental AFC will continue to emphasize the development of low-dimensional models that accurately represent the flow's response to external disturbances and the flow's response to actuator input, i.e., disturbance models and plant models. These can be used to design practical and effective feed-forward and feedback controllers that run in real-time on the digital-signal processing hardware. Examples of active flow control systems that have successfully employed these techniques include the reduction of unsteady loads in gusting flows on wings and road vehicles, thermo-acoustic combustion instability suppression, control over the reattachment length in separated flows over steps, reduced drag on bluff bodies, reduce acoustic tone amplitudes in cavity tones, control the rotational motion of slender bodies at high angles of attack, and reduce lift hysteresis on pitching wings.

Another important enabler in experimental flow control has been in sensor development. MEMs-based sensor technology is reducing the cost and size, and increasing the performance of pressure sensors, motion sensors with IMU's, gyro's, and accelerometers. This enables distributed sensing capability, such as multiple pressure sensors in a diffuser or over the surface of a wing, for better flow state estimation in practical applications. Shear stress sensors continue to be developed and applied with varying degrees of success in the laboratory, but have not developed to practical application levels. Full-field sensing techniques, such as real-time PIV, enable us to explore new methods of integrating experiments with numerical simulations. Data assimilation methods combine experiments and simulations as a way of correcting simulations.

Actuator technology is sufficiently mature for solving a large number of flow control problems, but one finds that our understanding of how the flow will respond to an arbitrary actuator input is somewhat limited. System Identification methods have proven to be useful for developing black-box models that are effective in modeling flows, but they are not as reliable as models developed from first principles.

Another challenge for all flow control investigators is to recognize that fluid dynamic time scales introduce time delays into the system that we are trying to control. These delays limit the achievable bandwidth of the overall system control, so actuator bandwidth is not a major issue for most flow control applications.

Authors: In your experience, what are some of the biggest gains in experimental flow control that came about from simulations or models? In contrast, what do you see as the key limitations of models in capturing real-world fluid phenomena?

Prof. Williams: Most of the significant advances in flow control were the result of collaborations between experiments, theory and simulations, so it is difficult to attribute advances in experimental flow control as being the direct result of simulations or models. For example, in what I consider to be the first modern flow control experiment, Prandtl used suction to suppress separation from one side of a cylinder, which provided evidence for the role of the boundary layer in flow separation and became the principal motivator for the development of boundary layer flow control. In this case theory preceded the experiment, but experimental observations led to the theory.

In my experience the key limitations of simulations in capturing real-world are matching Reynolds number, computing an accurate actuator input to the flow, and matching the dimensionality of the experiment. Two-dimensional simulations of the flow's response to actuation often produce inaccurate results, because the real-world actuators typically introduce three-dimensional disturbances into a flow. It is difficult for actuators used in experiments to produce two-dimensional disturbances.

In the case of closed-loop active flow control, models of the flow field response to external disturbances and models for the flow field response to actuator input are very useful for achieving effective control.

Authors: Can you comment on the role and benefits of experimental flow control in the future as computer simulations become more powerful?

Prof. Williams: The collaboration between experiments and simulations will become even stronger as simulations become more powerful. Fortunately the engineering community has outgrown the notion from the 1990s that simulations will make experiments obsolete. The emerging area of research known as "data assimilation" seeks to capitalize on integration of large-scale, real-time, experimental data sets with numerical simulations. Although the techniques being developed are not specifically focused on flow control applications, it seems likely that the approach will be useful for the entire flow control community. For example, imagine real-time experimental data being used to "correct" a full-scale DNS simulation of a particular flow. The DNS can provide the full-state feedback information to a controller that would not be available from the experimental data.

Simulations done correctly provide more detailed information about a flow than can be obtained by experiment. Experiments, on the other hand, can explore a wider parameter space than simulations. Simulations and models often identify instabilities that are susceptible to control. In principle, simulations (adjoint methods) can identify spatial locations where actuator placement can be most effective.

Authors: In the coming decades, where do you see the big future gains in our everyday lives and in industry due to flow control?

Prof. Williams: The Holy Grail of flow control would be the ability to control all scales of turbulence in shear flows, such as, in turbulent boundary layers, jets, wakes and mixing layers. Imagine being able to 'flip a switch' and laminarize a

turbulent boundary layer. Fuel savings for the aerospace industry would be enormous. However, I don't envision that capability in the near-term future, primarily because we don't have sufficient understanding of the turbulence production mechanisms, or ways in which actuators can be used to interact with those mechanisms. However, pursuing the goal of turbulence control drives the development of new technology in numerous areas. New approaches to flow modeling (e.g., resolvent modes), advances in sensor and actuator technology, and novel control algorithms will lead to improved understanding of the fundamental phenomena governing turbulence.

Linear thinking about the current trends in flow control suggests that AFC will result in more efficient energy recovery via improved aerodynamics on aircraft and wind turbines. Reduced drag and improved gust tolerance on commercial aircraft can be expected. I believe that it is possible for conventional flight control surfaces to be replaced by AFC actuators, and these aircraft will fly without vertical stabilizers and will fly with substantially larger aerodynamic efficiency, i.e., L/D ratio. AFC will enable tailored combustion processes that will reduce pollutants from combustion processes.

The rapid development of low cost and ever more powerful DSP's will enable more intelligent control of flows and systems. The application of distributed sensing will improve state estimation. Will there be a breakthrough in our understanding of turbulent flow mechanisms that enables laminarization of turbulent shear flows? Linear extrapolation says 'no', but fortunately, nature does not follow linear extrapolations.

Authors: Do you envision these being facilitated by better hardware, increasing model fidelity, access to more complete data, or some combination thereof?

Prof. Williams: Improvements in a combination of all of the above may lead to the better understanding that is required to control turbulent shear flows.

There is a difference between what we can do with flow control and what is of interest to industry. For example, we know how to delay airfoil stall with various types of leading edge actuation, such as, pulsed-pulsed blowing jets, but industry continues to use mechanical vortex generators for stall control, even though there is a continuous drag penalty. From industry's perspective the complexity and reliability issues of an AFC stall control system outweigh the benefits of reduced drag. It is obvious that AFC techniques must buy their way onto a system by providing greater benefits than the cost of added complexity. Sometimes those benefits are unexpected, such as, the use of pneumatic AFC for flight control of an aircraft. The primary function of the pneumatic actuator is to provide roll and yaw control, which takes bleed air from the engine. At the same time a secondary effect occurs, where the AFC increases the efficiency of the flight vehicle, so that the range and endurance are better with AFC than without. In this case, the overall system benefits from the application of AFC.

Authors: We thank you for sharing your insights in this interview!

Chapter 7
MLC Tactics and Strategy

Heavy is the brick of reality on the strawberry cake of our illusions.
Gilles "Boulet" Roussel, cartoonist, Translated from his Twitter account Bouletcorp, 10th December 2013

If the ideal MLC experiment existed, this chapter would not be needed. The literature contains many examples of control methods that fail the translation from a numerical study to the corresponding experimental demonstration. MLC removes one of the major obstacles: the discrepancy between the employed model and the physical reality of the experiment. Nonetheless there still exist many pitfalls that lead to disappointing results. Often, these results relate to the difference between the idealized experiment and reality. Through our first applications of MLC in experiments we have explored many pitfalls and have developed a mitigation strategy. This chapter guides experimental control demonstrations with MLC. The advice may also apply to numerical studies.

7.1 The Ideal Flow Control Experiment

The ideal flow control experiment—or any other control experiment—does not exist. If it existed, it would have the following properties:

Full knowledge about the dynamics: The evolution equation $d\mathbf{a}/dt = \mathbf{F}(\mathbf{a}, \mathbf{b})$ and the parameters of the dynamics $\mathbf{F}$, like the Reynolds or Mach number, are exactly known. A corollary is that reproducibility is guaranteed.

Accurate measurements: The measurement equation $\mathbf{s} = \mathbf{G}(\mathbf{a}, \mathbf{b})$ is known with perfect precision and the signals $\mathbf{s}(t)$ are accurately recorded.

Ideal cost function: The cost function $J(\mathbf{a}, \mathbf{b})$ describes the quantity which shall be optimized and which can be measured in the experiment.

T. Duriez et al., *Machine Learning Control – Taming Nonlinear Dynamics and Turbulence*, Fluid Mechanics and Its Applications 116,
DOI 10.1007/978-3-319-40624-4_7

Known noise: Any noise terms affecting the dynamics, the measurement equation or the cost function are accurately modeled and accounted for. A corollary is the reproducibility of arbitrary ensemble averages.

Controllability: Any desired final state $\mathbf{a}$ of the system can be reached with finite actuation in finite time. At a minimum, the achieved change of cost function due to MLC is so impressive, that the results merit a career promotion and convince the funding agencies to invest more money.

Infinitely powerful computer: The control law $\mathbf{b} = \mathbf{K}(\mathbf{s})$ is computed instantaneously. At minimum, the actuation command is computed with predictable small time delay.

No aging: Of course, the ideal experiment never breaks, nor suffers slowly drifting external parameters such as atmospheric pressure and temperature, nor is effected by opening and closing doors of the room, nor experiences any changes which are not reflected in the dynamics $\mathbf{F}$, in the measurement function $\mathbf{G}$ and in the cost function J.

Infinite resources: The perfect experiment can be operated until the cost function value is converged and until MLC is converged to the globally optimal control law.

Arguably, the ideal flow control experiment is an instance of a direct Navier–Stokes simulation on an infinitely powerful computer. When building a MLC control experiment the best degree of perfection shall be reached with given resources. In the following sections, we address different aspects that one needs to keep in mind when preparing an MLC experiment. Some decisions can be made to prepare a perfect experiment. And other decisions deal with imperfections of an existing experiment. This chapter can be considered as a checklist of important aspects that need to be taken into account in order to maximize the chances of success and the quality of the results.

7.2 Desiderata of the Control Problem—From the Definition to Hardware Choices

In this section, we outline desiderata of the multi-input multi-output (MIMO) control problem. This comprises the definition of the cost function (Sect. 7.2.1), the choice of the actuators (Sect. 7.2.2), and the choice of the sensors (Sect. 7.2.3), including a preconditioning of the actuation command and sensor signals for MLC. In (Sect. 7.2.4) we remind the reader that the search space for control laws is a design choice as well. Many aspects are trivial. This does, however, not exclude that one or the other aspect might be overlooked. Even the most sophisticated control problem can be decomposed in myriad of trivialities.

7.2.1 Cost Function

The purpose of any actuation is to change the behavior of the plant to be more desirable. The corresponding performance measure is generally called the *cost function* in control theory and is generally a positive quantity to be minimized. A control problem may have a 'natural' cost function comprising the departure of the system from the ideal state J_a and the associated cost J_b. We take the aerodynamic drag reduction of a car by unsteady actuators as an example. The propulsion power to overcome the drag F_D at velocity U reads $J_a = U F_D$ and is ideally zero. The actuation power by blowing can be estimated by the energy flux $J_b = \rho S_b \overline{U_b^3}/2$, where ρ is the density of the fluid, S_b is the area of the actuator slot, and $\overline{U_b^3}$ the third power of the actuator velocity averaged over the slot area and time. The resulting cost function reads

$$J = \underbrace{F_D\, U}_{J_a} + \underbrace{\rho S_b \overline{U_b^3}}_{J_b}. \tag{7.1}$$

Let J_u be the parasitic drag power without forcing. Then, the net energy saving reads $\Delta J = J_a + J_b - J_u$ and shall be maximized, or, equivalently the total energy expenditure $J_a + J_b$ minimized. The ratio between energy saving and actuation energy $(J_u - J_a)/J_b$ defines the efficiency of the actuators and should be much larger than unity. We shall not pause to consider potential improvements on this particular cost function.

Two common cases may obstruct the definition of a 'natural' cost function. First, the cost function cannot be measured in experiments. For instance, the test section of the wind tunnel may not have a scale and the drag force cannot be measured. In this case, a surrogate cost function needs to be defined. A good surrogate function is expected to follow approximately—at least qualitatively—the changes of the intended quantity for all considered control laws. In case of the car example, the base pressure coefficient is a possible choice.

A second challenge may be the incomparableness of control goal J_a and actuation cost J_b. This may, for instance, be manifested in different physical units of J_a and J_b. The control goal may be to improve mixing as quantified by an non-dimensional entropy and the actuation cost may be measured in Watts. In this case, cost function decomposition in a state and actuation contribution $J = J_a + J_b$ is even dimensionally wrong and there is no a priori performance benefit $J_{a,u} - J_a$ which is worth J_b in actuation energy. In this case, the actuation cost is typically rescaled with a penalization parameter γ,

$$J = J_a + \gamma\, J_b. \tag{7.2}$$

In the following, we assume that all quantities are non-dimensionalized and $\gamma > 0$ is a positive real number. Let us further assume that the experimentalist decides that the performance benefit $J_1 = J_u - J_a$ is worth the actuation cost J_2. In this case,

the corresponding penalization parameter reads $\gamma = J_1/J_2$. The reference values J_1 and J_2 might be obtained from another control experiment. For instance J_1 may be the performance benefit from the actuation cost J_2 from periodic forcing. In this case, the same choice of γ leads to a lower J-value (7.2) for MLC if the actuation benefit $J_u - J_a$ is larger at the same actuation cost or if the same actuation benefit is achieved at lower actuation cost.

7.2.2 Actuators

The goal of actuation is to modify the system behavior to reduce the cost function. This self-evident goal may not be achieved by the actuators in real-world experiments, as the authors have witnessed a number of times.

The goal may be to reduce jet noise with plasma actuators at the jet nozzle exit. But these actuators may be orders of magnitude more noisy than the jet. Then, the best control strategy is to keep actuation off. The goal may be to reduce the drag of a bluff body, but the blowing actuators increase drag for all tested periodic forcing. The goal may be to mitigate flutter in a turbine, but the zero-net-mass-flux actuator in front the propeller has no perceivable effect. The goal may be to reduce skin friction by mitigating incoming Tollmien–Schlichting waves, but the actuator only triggers three-dimensional transition, which is far worse for drag. An ill-chosen actuation strategy can undermine the entire control campaign. Hence, it is very comforting if there exists a control study in the considered plant for which the actuation has been shown to have a beneficial affect on the cost function.

A second, much easier aspect to consider is the formulation of the regression problem for MLC. As the constants for genetic programming are chosen from a parameter range, say $[-5, 5]$, the actuation command $\mathbf{b}$ and sensor signal $\mathbf{s}$ should also be of order one. Evidently, the learning of any regression technique is accelerated, if it has to find mappings between order-one quantities as opposed to translating a $O(10^6)$ sensor signal into $1 + O(10^{-6})$ actuation command. In the simple case, that actuation may only be 'on' or 'off', as with some valves, and a good choice of the actuation command is 0 for 'off' and 1 for 'on'.

7.2.3 Sensors

The purpose of the sensors is to infer control-relevant features of the flow state. In linear control theory, the sensors ideally allow one to estimate the controllable subspace of the system. In a turbulent flow, the choice of the type, the locations, and the dynamic bandwidths of the sensors should reflect the assumed enabling actuation mechanism. We refer to [43] and references therein for corresponding guidelines.

Secondly, the sensor signals are ideally hardly correlated so that each sensor provides new information about the state.

Thirdly, we reiterate that the sensor signals should be suitably centered and scaled to become of order unity. Let s_i^* be the raw ith sensor signal. Let $s_{i,\min}^*$ and $s_{i,\max}^*$ be the minimal and maximal values, respectively. Then, learning of MLC is accelerated by using the normalized signal,

$$s_i = \frac{s_i^* - s_{i,min}^*}{s_{i,max}^* - s_{i,min}^*}, \tag{7.3}$$

as input in the control law.

As discussed in the examples of Chap. 6, fluctuations may be a better input for the control law than the raw signal which is affected by drifts. Frequency filtering of the raw signals may also improve performance of MLC. Yet, we strongly disadvise against any of such filtering—at least in the first applications of MLC. Any filtering is an expression of an a priori assumption about the enabling actuation mechanism. This bias may prevent MLC to find a more efficient unexpected control, as we have witnessed in half of our studies.

7.2.4 Search Space for Control Laws

The search space for control laws is a potentially important design parameter of MLC. In Chap. 3, the linear-quadratic regulator $\mathbf{b} = \mathbf{K}\mathbf{a}$ was shown to be optimal for full-state feedback stabilizing a linear plant. This result has inspired the MLC search for full-state, autonomous, nonlinear feedback law $\mathbf{b} = \mathbf{K}(\mathbf{a})$ stabilizing the nonlinear dynamical system of Chap. 5. Similarly, MLC has improved the performance of the experiments in Chap. 6 with a sensor-based, autonomous, nonlinear feedback law $\mathbf{b} = \mathbf{K}(\mathbf{s})$. However, the optimal control of a linear plant which perturbed by noise involves filters of the sensor signals, as discussed in Chap. 4.

For the simple nonlinear plant in Chap. 5, we have made the case that closed-loop control is not necessarily more efficient than open-loop control. The generalized mean-field system of this chapter can easily be modified to make periodic forcing more optimal than full-state autonomous feedback. In some experiments, a periodic high-frequency forcing $b^*(t) = B^* \cos \omega^* t$ may stabilize vortex shedding much better than any sensor-based autonomous feedback. In this case, any deviation of the actuation command from a pure periodic clockwork might excite undesirable vortex pairing. A simple solution for the MLC strategy is to consider the optimal periodic forcing as an additional 'artificial' sensor $s_{N_s} = b^*$ and, correspondingly, as input in the feedback law $\mathbf{b} = \mathbf{K}(\mathbf{s})$. Now, this law depends explicitly on time and is not autonomous anymore. The plant sensors may help to adjust the forcing amplitude, like in extremum seeking control, or introduce other frequencies. The authors have supervised MLC experiments in which this approach has lead to pure harmonic actuation, to multi-frequency forcing and to nonperiodic forcing as best actuation. The search space for optimal control laws may also include multi-frequency inputs

and filters. Our preferred strategy is to start with direct sensor feedback $\mathbf{b} = \mathbf{K}(\mathbf{s})$ and integrate the best periodic periodic forcing as control input before exploring more complex laws.

7.3 Time Scales of MLC

The MLC experiment is affected by three time scales. First, the inner loop has a time delay from sensor signals to actuation commands. This time scale is dictated by the employed hardware, for instance, the computating time for the control law (Sect. 7.3.1). Second, the inner loop has a plant-specific response time, i.e. a time delay from the actuation to the sensed actuation response. This time scale is related to the plant behavior, e.g. the choice of the actuators and sensors and the flow properties (Sect. 7.3.2). Third, the outer MLC loop has a learning time governed by the number and duration of the cost function evaluations before convergence (Sect. 7.3.3). In this section, we provide recommendations how to deal with all three time scales.

7.3.1 Controller

In the computational control studies of Chaps. 2, 4 and 5, we write the control law as

$$\mathbf{b}(t) = \mathbf{K}(\mathbf{s}(t)), \tag{7.4}$$

i.e. the actuation reacts immediately on the sensor signals. In a real-world experiment, there is a time delay τ_b from sensing to actuation leading to another optimal control law,

$$\mathbf{b}(t) = \mathbf{K}_{\tau_b}(\mathbf{s}(t - \tau_b)). \tag{7.5}$$

This time scale comprises three delays: (1) the propagation time from the sensor location to the computational unit, e.g. the propagation of the pressure from the body surface to the measurement device in tubes; (2) the computational time needed to compute $\mathbf{K}_{\tau_b}(\mathbf{s})$; and (3) the time from the computed actuation command $\mathbf{b}$ to the actuator action. The times (1) and (3) would not occur in a computational study. One could argue that (1) and (3) belong to the physical response time of the plant or to the controller. In this section, we do something much simpler: We focus on the computational time and ignore the plant-related delays. In addition, we assume that τ_b is fixed and small with respect to the time scale of the plant dynamics, $\tau_b \ll \tau$. The plant time scale might be quantified by or related to the first zero of the sensor-based correlation function,

$$C_{\mathbf{s}}(\tau) = \overline{\mathbf{s}(t) \cdot \mathbf{s}(t - \tau)} = 0. \tag{7.6}$$

In fact, we can be less restrictive with respect to τ_b if the sensor signal at time $t - \tau_b$ allows a good prediction of the values at time t,

$$\mathbf{s}(t) = \phi_{\tau_b}\left[\mathbf{s}(t - \tau_b)\right]. \tag{7.7}$$

Here, ϕ_{τ_b} is the propagation map which may also be identified with the discussed regression techniques. This can, for instance, be expected for oscillatory dynamics. In this case, the instantaneous control law (7.4) can be recovered from Eq. (7.5) using the predictor (7.7).

In the sequel, τ_b will be considered as the computational time. This can easily be estimated for a binary tree of genetic programming with single input ($N_b = 1$). Let us assume that the tree is maximally dense and has the depth 10. In this case, the evaluation requires $2^9 + 2^8 + \ldots + 1 = 1023$ operations. A cheap modern laptop has roughly 1 GFlop computing power, i.e. can compute 10^9 operations per second. Hence, the computation time for one actuation command is 1 μs, or one millionth of a second. This is one order of magnitude faster than an ultra-fast sampling frequency 100 kHz of modern measurement equipment. Typical time scales in wind tunnels range from 10 to 100 Hz. Thus, $\tau_b/\tau \leq 10^{-4}$ and the computation time τ_b can be considered small if not tiny by a very comfortable margin. The computer in the TUCOROM experiment (Sect. 6.3) has 1 TFlop computing power, which would add another factor of 1000 as safety margin. The computational time for the control law is apparently not a major concern.

The real challenge is transferring the LISP expression of the control law to the computational unit. The LISP expression may be pre-compiled and transferred to the computational unit before evaluation of the cost function. In this case, the computation time is minimal. The LISP expression may also be read from the hard disk every computation cycle and interpreted by the computational unit. The corresponding time is orders of magnitude slower.

In other words, the potential bottleneck is the communication between the inner RT loop and the outer learning loop. Many communication protocols and architectures exist.[1] We consider here two philosophies, the mentioned interpreter implementation (LISP parser) and the pre-compiled solution.

RT LISP parser: This parser function takes the LISP expression and sensor readings as the inputs and outputs the control commands. The principle for this function is simple but auto-regressive which means that the CPU and memory cost is bounded by $2^l - 1$, where l represents the depth of the expression tree, i.e. function complexity. Due to CPU and memory usage it can lead to low RT scheduler frequency and overruns. This strategy is advisable only with Concurrent, DSpace or equivalent high-end RT systems. Functions of this type have already been written and tested in Simulink®, Labview®, Fortran, Matlab®and C++. This approach has been employed in the TUCOROM mixing layer experiment (see Sect. 6.3).

[1] In principle, the file exchange can happen over different locations in a cloud, with a fast computational unit close to the experiment and the MLC learning performed remotely.

Pre-compiled RT controller: In this case, MLC generates the code for the controller and compiles it. The code will contain the whole generation to be evaluated. The generated code will also make the update of the control law after each evaluation and account for the transient time between evaluations. The cost function values may be returned to the outer MLC loop either after each evaluation or after grading the whole generation. The second communication protocol is evidently much faster than the RT LISP parser and advised for slower systems, like the Arduino processors of Sect. 6.2.

The choice of the communication depends on the kind of RT system available: a parser allows one to update the control law on the fly but is more demanding in memory and CPU resources. This strongly indicates the use of a proper RT dedicated computer with an efficient scheduler. On the other hand, a compiled micro-controller is less flexible: the change of control law is either a scheduled event or enforces a new compilation. The pre-compiled controller is not demanding in terms of programming skills and can be operated with low-cost hardware.

The code of the RT system and of MLC may be written different languages. The authors have successfully implemented the Matlab®code `OpenMLC` on:

- Matlab®and Simulink®(for ODE integration, as in Chaps. 4 and 5);
- C++, (Arduino) as in Sect. 6.2;
- Fortran (CFD code) and
- Labview®as in Sect. 6.3.

The use of any other language should not be more than a 3 hour exercise for a skilled programmer.

7.3.2 Response Time of the Plant

Fluid flows are convective in nature. This leads to a time delay between actuation and sensing. In feedforward control, actuators respond to oncoming perturbations detected by upstream sensors. The mitigation of Tollmien–Schlichting waves in laminar flows is one example for which this approach is shown to be successful. In a feedback arrangement, the sensors record the actuation effect with a time delay. This time delay can often be estimated by the convection time. Most turbulent flows fall in this category, in particular drag reduction or mixing enhancement in bluff-body wakes.

In model-based control design, time delays are known to cause undesirable control instabilities and lack of robustness. For model-free control, the time delay between actuation and sensing is not a problem per se. Several MLC-based actuation strategies have successfully worked with a time delay of 1 to 2 characteristic periods. Yet, a degrading control performance may be caused by the increasing uncertainty of the flow state between actuation and sensing. Ideally the time delay between actuation and sensing should be long enough so that the actuation effect is clearly identifiable

beyond the noise threshold but short enough so that the state uncertainty is still acceptable. This aspect is beautifully demonstrated in wake stabilization experiment by Roussopoulos [227].

7.3.3 Learning Time for MLC

The learning time T_{MLC} of MLC using genetic programming with N_g generations and N_i individuals reads:

$$T_{\mathrm{MLC}} = N_g \times N_i \times (T_{\mathrm{transient}} + T_{\mathrm{evaluation}}) \tag{7.8}$$

where $T_{\mathrm{transient}}$ is the transient time for a new control law and $T_{\mathrm{evaluation}}$ the evaluation time of the cost function.

A proper planning of a MLC experiment includes good choices for the parameters that determine T_{MLC} in Eq. (7.8). A minimal transient time $T_{\mathrm{transient}}$ is determined by the flow physics. The time for a transient from an actuated to an unforced state (or the inverse process) is a good reference value. The evaluation time $T_{\mathrm{evaluation}}$ should be large enough so that the *approximate* ordering of the J-values is right. This requirement is distinctly different from requiring a 1 % or 5 % accuracy of the J-value. The evaluation time for MLC can be expected to be much shorter if high accuracy is not required. One might want to invest in the evaluation time during the final generations when the explorative phase turns into an exploiting phase of MLC.

The next decision concerns the population size N_i and the number of generations N_g. In larger-scale experiments, the measurement time T_{MLC} is expensive and given as a side constraint. Thus, the decision is how to distribute a given number of runs $N_i \times N_g$ among population size and convergence. Given, for instance, $N_i \times N_G = 1000$ evaluations in a measurement campaign, one is left with several possibilities:

1 **generation of** 1000 **individuals.** The search space is explored as widely as possible but no convergence is achieved: this is a Monte-Carlo process.

10 **generations of** 100 **individuals.** The search space is initially less explored, but the evolution will provide increasingly better individuals.

100 **generations of** 10 **individuals.** The search space is hardly explored initially. If the initial generations include winning individuals, the evolution may provide progressively better values, like in gradient search. Otherwise, the evolution may terminate in a suboptimal minimum for the cost function.

In other words, the choice depends on the assumed complexity of the cost function variation or number of local minima. The more complex the landscape of the cost function is, the more MLC will profit from exploration with a large number of individuals in a given generation. The more simple the landscape, the more MLC will benefit from exploitation with large number of generations. A good strategy is to start with a large population at the beginning for good exploration and switch to a much smaller population for good exploitation. The degree of exploration and

exploitation is also determined by the MLC parameters as will be elaborated in the following section.

7.4 MLC Parameters and Convergence

In this section, we discuss the convergence process of MLC based on genetic programming. The discussion is representative for other evolutionary algorithms as well. The purpose is to minimize the MLC testing time (7.8), more precisely $N_i \times N_g$, for an acceptable control law.

Most regression techniques are based on iterations. Gradient search, like Newton's method, can be conceptualized as an iteration of a population with a single individual $N_i = 1$ sliding down a single smooth local minimum. This process is called *exploitation*. Evolutionary algorithms, like genetic programming, assume multiple minima and hence evolve a large number of individuals $N_i \gg 1$. The search for new minima is called *exploration*. Evidently, evolutionary algorithms need to make a compromise between exploring new minima and exploiting found ones.

In the following, we discuss diagnostics of the convergence process (Sect. 7.4.1), MLC parameters setting the balance between exploration and exploitation (Sect. 7.4.2), and pre-evaluation of individuals to avoid testing unpromising candidates (Sect. 7.4.3).

7.4.1 Convergence Process and Its Diagnostics

We analyze a complete *MLC run* from the first to the last generation N_g with N_i individuals each. Focus is placed on the cost function values as performance metrics. Let J_i^j be the cost function value of the ith individual in the jth generation. These values are sorted as outlined in Chap. 2. First, within one generation the cost function values are numbered from best to worst value:

$$J_1^j \le J_2^j \le \ldots \le J_{N_i}^j \quad \text{where} \quad j = 1, \ldots, N_g. \tag{7.9}$$

Second, elitism guarantees that the best individual is monotonically improving with each generation, neglecting noise, uncertainty and other imperfections of the plant:

$$J_1^1 \ge J_1^2 \ge \ldots \ge J_1^{N_g}. \tag{7.10}$$

Figure 7.1 illustrates the J-values of the first regression problem in Sect. 2.3.1. The performance of MLC may be assessed from the following quantities. The best individual J_1^j, $j = 1, \ldots, N_g$ is depicted in the bottom blue curve in Fig. 7.1. MLC appears to be converged after about 30 generations.

The whole population may be characterized by the median value $J^j_{\lfloor N_i/2 \rfloor}$, where the brackets $\lfloor \cdot \rfloor$ represent the nearest integer. The J-values vary over orders of magnitude in each generation so that the mean value or variances are strongly affected by the worst outliers. Arguably, the median value is a much better parameter of the evolution of the whole population. The difference between best and median individual is an indicator of the diversity of the population. The diversity is needed to explore new potentially better minima.

The diversity of the evolutionary process is also characterized by another J-value. The larger the J-value of an individual the less probable is the selection for any genetic operation into the next generation. The J-value above which only 1 % of the population is selected for an operation is called the *1 % limit* and displayed in Fig. 7.1. This limit follows roughly the median in the presented example.

A peculiar feature of Fig. 7.1 is the yellow curve corresponding to the identically vanishing control function. This *zero-individual* can easily be produced from any individual by multiplying it with 0.

A more detailed insight in the evolution dynamics may be gained from population densities. Let $J_{\min}$ and $J_{\max}$ be the best and worst cost values found in the whole MLC run. We equipartition $\log_{10} J$ in 1000 equidistant intervals $I^{\text{pdf}}_k = \left[J^{\text{pdf}}_{k-1}, J^{\text{pdf}}_k\right]$ where

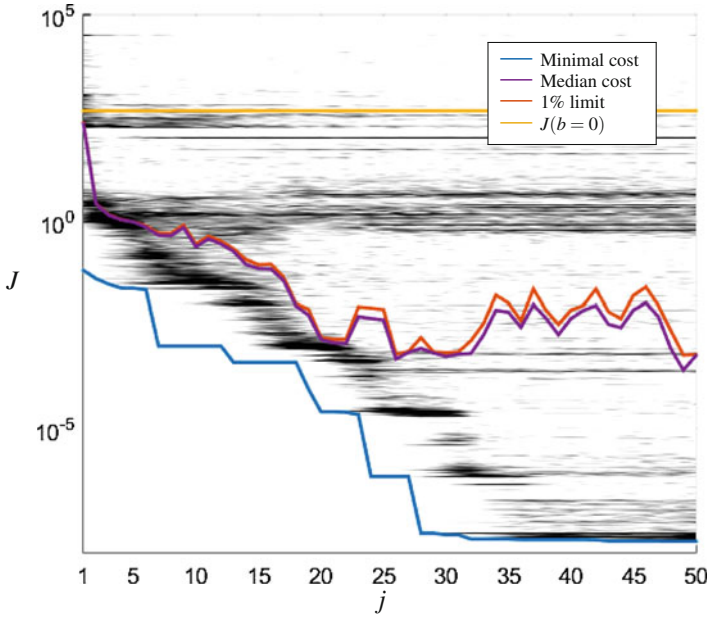


Fig. 7.1 MLC run for example in Sect. 2.3.1 featuring the best and median J-values In addition, the 1 % limit indicates that all individuals above this line cumulate a 1 % contribution to the genetic content of the next generation. The *yellow line* indicates the cost of all identically null individuals

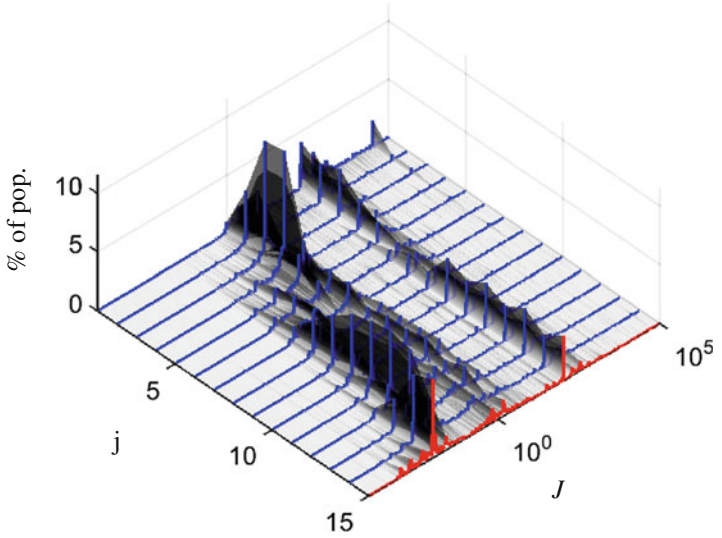


Fig. 7.2 Same MLC run as in Fig. 7.1 but with a 3-dimensional view of the population density (7.11)

$$J_{\min} = J_0^{\text{pdf}} < J_1^{\text{pdf}} < \ldots < J_{1000}^{\text{pdf}} = J_{\max}.$$

The logarithmic equipartition separates values which are orders of magnitude different. Let

$$n_k^j := \text{card}\left\{ J_i^j : i = 1, \ldots, N_i \wedge J_i^j \in I_k^{\text{pdf}} \right\} \tag{7.11}$$

be the number of J values of the jth generation in the kth interval. Figure 7.2 displays the population density n_k^j / N_i. We see islands of highly populated similar J-values over several generations. These islands tend to get initially increasingly populated until better islands are discovered.

The evolution of the generations can be beautifully visualized with multidimensional scaling. This leads to a mathematically well-defined illustration of the principle sketch of Fig. 2.12, i.e. indicates the landscape of J spanned by the population of control laws. Let $\mathbf{K}^i(\mathbf{s})$ denote the ith control law of an MLC run, i.e. $i = 1, \ldots, N_i$ N_g. Let $\gamma^i = \left(\gamma_1^i, \gamma_2^i\right)$ be a two-dimensional vector corresponding to the control law. Let $D^{ij} = \langle \|\mathbf{K}^i(\mathbf{s}) - \mathbf{K}^j(\mathbf{s})\| \rangle$ be a suitable metric between the ith and jth control law. Kaiser et al. suggest such a metric in [155]. Then multidimensional scaling provides a configuration of γ^i which optimally preserves the metric $\|\gamma^i - \gamma^j\| \approx D^{ij}$. Figure 7.3 shows an example of a closed-loop control experiment targeting drag reduction of a car model. The illustration indicates a clustering and scattering of control laws with an apparent minima in the lower left corner. Similar visualizations of other MLC experiments showed far more complex topologies. The understanding of these topologies is still a subject of active research.

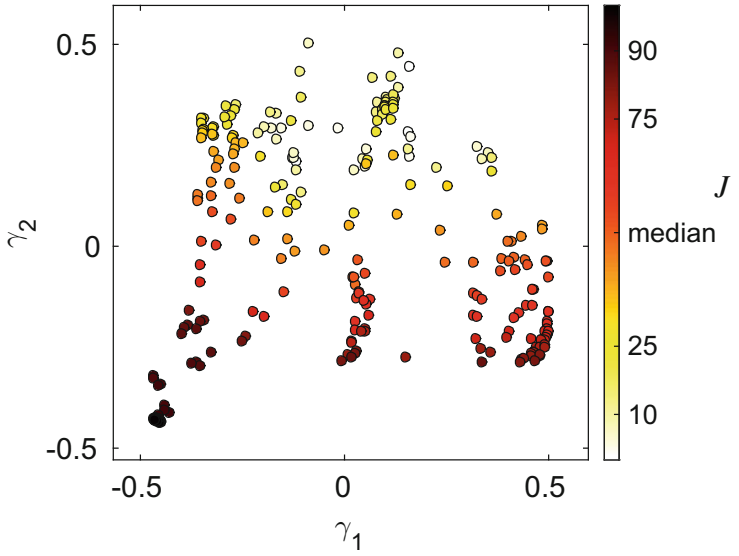


Fig. 7.3 Visualization of an experimental MLC run for drag reduction of a car model with multi-dimensional scaling (Courtesy: Ruiying Li for the data and Eurika Kaiser for the visualization). The circles correspond to all individuals of a converged evolution with $N_i = 50$ and $N_g = 5$. The J-performance of the individuals are color-coded with respect to the ordering. A control law at the 90th percentile for instance illustrates that 90 % of the J-values are worse and only 10 % of the J-values are better. The plane (γ_1, γ_2) preserves a metric between control laws in an optimal manner (multi-dimensional scaling, see [155] for details)

7.4.2 Parameters

As outlined in Sect. 7.3.3, a large population size N_i promotes exploration while a long evolution N_g supports convergence. In addition, the learning rate from generation to generation with given population size is affected by several MLC parameters. The visualizations of Sect. 7.4 give a good indication of this learning process.

The learning speed of MLC is influenced by the genetic operation probability and the selection of individuals.

Elitism N_e: A non-vanishing N_e ensures that the best performing individuals reappear in the next generation and are not forgotten. Generally, $N_e = 1$ is a good minimal choice. A larger number tends to copy very similar individuals in the new generation, i.e. neither contributes to diversity nor to performance.

Genetic operation probability P_c, P_m **and** P_r: The sum of the three probabilities must be unity, implying that only two probabilities can be independently chosen. As discussed in Chap. 2, replication ensures the memory of good individuals, crossover promotes exploitation and thus better individuals, and mutation promotes exploration and diversity of the population. Exploration, i.e. finding the right minima, is initially more important that exploiting the minima that have

already been discovered. Good starting values for $N_i \sim 100$ were found to be $(P_r, P_c, P_m) = (0.1, 0.6, 0.3)$. $P_r = 0.1$ appears to be a uniformly good choice independent of the population size N_i. For much larger population size $N_i \gg 100$, a smaller P_m and larger P_c are advisable. Conversely, smaller population sizes profit from choosing a larger mutation rate, e.g. $P_m = 0.4$.

Selection: The arguments of the genetic operations are determined by a stochastic selection process of the individuals. Like genetic operations, the selection may promote exploitation or exploration. The harsher the bias towards best performing individuals, the larger the probability that the local minima will be exploited. The individuals are chosen with equal probability in a tournament type selection. The harshness of the selection is set by the tournament size. Setting $N_p = 7$, enforces that the upper 50 % of the current population accounts for the genetic content of 99 % of the next generation for $N_i = 100$.

7.4.3 Pre-evaluation

The initial generation is composed of randomly chosen individuals. Evidently, one can expect that most individuals are non-performing, probably performing worse than the zero-individual $b \equiv 0$. If random control laws would improve the cost function, then control design would be simple.

Hence, a pre-evaluation of initial individuals is advised. Zero-individuals, for instance, should be replaced by other random individuals. Constant individuals might also be excluded, depending on the control problem. Non-trivial individuals may be excluded if typical sensor readings lead to function values or function behaviors which are clearly out of bound in terms of amplitude or frequency content. The delicate aspect is that the sensor readings change under actuation and a good guess of the probability distribution of sensor readings $p(\mathbf{s})$ is required.

Another strategy to avoid the testing of non-promising individuals is to reduce the number of operations per individuals for instance by low initial tree depth for the first generation.

Advanced material 7.1 Pre-evaluation in `OpenMLC`.
A more proactive measure can be undertaken by the use of a pre-evaluation function. This function takes the individual as an argument and returns a simple logical. If 'false' is returned, the individual is discarded and a new one is generated (or evolved). Detection of zero-individuals can usually be easily achieved with a simple simplification of the LISP expression (`OpenMLC`provides the function `simplify_my_LISP.m`). Saturated control laws can be detected by providing random time series' covering the whole sensor range for all sensors and discarding the control laws that stay constant (or keep the same value more than a pre-determined percentage of the total samples).

7.5 The Imperfect Experiment

Finally, we discuss the performance of MLC under imperfections of the experiments, like high-frequency noise (Sect. 7.5.1), low-frequency drift (Sect. 7.5.2), and the need to monitor the experiment's health status (Sect. 7.5.3).

7.5.1 Noise

In turbulence, noise is not an imperfection, but an intrinsic feature. For the purpose of control, we consider noise as the high-frequency fluctuations which are not relevant for the actuation mechanism. Noise impacts two aspects of MLC, the actuation command and the cost function. The actuation command $\mathbf{b} = \mathbf{K}(\mathbf{s})$ is affected by noise in the sensor signals. Every operation in the expression tree for $\mathbf{K}$ can act as noise amplifier and at some complexity the control law may become a noise generator. Hence, the number of operations or the depth of expression tree needs to be limited.

Second, the impact of noise on the cost function is far less critical, as the corresponding integrand is generally simple and the noise is largely integrated out.

The use of filters may improve MLC performance. Examples are low-pass, band-pass or single-frequency Morlet filters. We emphasize, again, that any predetermined filter introduces a bias towards the expected actuation mechanism which is not necessarily the most effective one. The choice of the proper filter may also be integrated in the MLC search algorithm like in Chap. 4.

7.5.2 Drift

Low-frequency drifts may arise due to changing operating conditions, e.g. changing oncoming velocity, temperature change or varying hot-wire anemometer readings. Drifts imply that effective individuals may degrade in performance and ineffective individuals may improve. Drifts may also lead to the choice of different sensors in the best control laws. However, within one or few generations, MLC will adjust to the new operating conditions if the change in performance of a given individual is slow with respect to the testing time of one generation. If the change is significantly faster, the cost function values of a single generation are wrong and lead inevitably to a poor ordering.

The cure for drifts depends on the cause. If the sensor readings drift, a re-calibration in regular intervals is advised. If the cost function drifts, the cure may be a normalization with the flow response to a periodic forcing. If the drifts are measured, like the oncoming velocity, the parameter may be integrated into the control law. We shall not pause to give a more exhausting list of drifts and potential cures.

7.5.3 Monitoring

Finally, a MLC run may be a time-consuming experiment with complex installations. Apart from slow drifts, there may be sudden unintended changes of the equipment. One or several hot-wire sensors may break. A pressure sensor may get clogged. An actuator may fail. Or there may be a sudden drop in the pressure reservoir of the actuator leading to significantly reduced actuator efficiency. Parts of equipment may unintentionally vibrate for some control laws. And the list can be indefinitely continued. Hence, an automatic extensive health monitoring of the plant is strongly advised to avoid extensive time on unsatisfying results. The authors have experienced several times that MLC has exploited weaknesses of problem definition, the wind tunnel or the actuator setup. It is a common experience that control optimization exploits the weakest links of the theoretical problem formulation/constraints or imperfections of the plant.

Chapter 8
Future Developments

The most fruitful areas for the growth of the sciences were those which had been neglected as a no-man's land between the various established fields
Norbert Wiener Cybernetics: Or Control and Communication in the Animal and the Machine, 1948

The future of machine learning control is bright, bolstered by concrete success stories and fueled by steady advances in hardware and methodology. As pointed out by N. Wiener, there is particularly fertile ground between two well-developed fields [276], and it is likely that there will be many exciting new technological developments between the fields of machine learning and control theory. This field will remain vibrant as long as

(1) it addresses problems that are interesting, important and challenging,
(2) improving hardware facilitates increasingly ambitious demonstrations, and
(3) investments in fundamental research yield new paradigms and mathematical frameworks.

Machine learning methods are beginning to enter the control of complex systems with numerous demonstrations in closed-loop turbulence control. These machine learning methods already pervade technology and information infrastructure, with smartphones providing compelling examples. Potential applications in engineering and industry may have an equally transformative impact. Examples include drag reduction of cars, trucks, trains, ships, airplanes and virtually any ground-, air- and water-born transport vehicle, lift increase of airfoils, aerodynamic force control mitigating wind gusts, internal fluid transport in ventilation systems and oil pipelines, efficient operation of wind turbines, greener and stable combustion and efficient chemical processes, just to name a few. Other applications concern more broadly any fluid flows or networks with well defined inputs (actuators) and outputs (sensors), like water and electricity supply in civil engineering, drug dosage and interactions for medical treatments, or financial trading.

T. Duriez et al., *Machine Learning Control – Taming Nonlinear Dynamics and Turbulence*, Fluid Mechanics and Its Applications 116,
DOI 10.1007/978-3-319-40624-4_8

In this chapter, we outline promising methodological advances that are poised to transform machine learning control (Sect. 8.1). Section 8.2 is dedicated to promising enablers for systems with high-dimensional inputs and outputs. We also highlight a number of exciting future applications that we believe will be heavily impacted by MLC, resulting in innovative solutions to scientific, engineering, and technological challenges of the 21st century [43] (Sect. 8.3). These directions are not exhaustive, but merely offer a glimpse into the exciting future developments of machine learning in the control of complex nonlinear systems. This chapter concludes with an interview of Professor Belinda Batten (Sect. 8.5). Prof. Batten is an internally renowned scholar in reduced-order modeling and control and pushes the frontiers of renewable wind and water energy.

8.1 Methodological Advances of MLC

One of the many benefits of MLC using genetic programming is its simplicity and generality, requiring little or no knowledge of the system being controlled. However, there are a number of targeted improvements that may expand the power and applicability of genetic programming for MLC to a wider range of problems with little added complexity.

Exploiting closeness of controllers: A major goal of future efforts is to reduce the training time required for MLC using GP. Genetic programming is an extremely flexible framework for building complex input–output functions, although these representations are not unique, and it is often unclear whether or not two GP controllers are similar in function. Improving this notion of *closeness* of controllers is critical to avoid redundant testing and to reduce training time. In addition, having an induced metric on the space of GP controllers may result in effective controller reduction, so that complicated, high-performance controllers can be reduced to their simplest form. One exciting development in this direction is related to embedding the space of GP controllers in a simple metric space using hash inputs. The function of a GP controller can be uniquely identified based on its response to a random input sequence, likely sampled on the attractor. These hash functions then define an embedding which is much lower dimensional and Euclidean; moreover, if the sampling is truly random and incoherent with respect to the *structure* of the controllers, then it is likely that the embedding will preserve the controller geometry. This is closely related to compressed sensing theory (see below) and the Johnson-Lindenstrauss theorem [147].

Multi-dimensional scaling: The progress of an evolutionary algorithm may be visualized in a two-dimensional plane with a suitable metric of the control laws and multi-dimensional scaling. Thus, close (distant) control laws in the metric space are close (distant) in the visualization plane. Each control law is associated with a value of a cost function. Thus, the visualization plane may indicate the topology of the cost function, e.g. the number of populated minima, maxima

or saddle points. An example of such a visualization of an ensemble of control laws has been provided in Sect. 7.4.1. Multi-dimensional scaling may also be used to illustrate MIMO control laws. These visualizations may provide valuable information for on-line decisions during a control experiment, e.g. tuning the levels of exploration versus exploitation.

Library building and dictionary learning: Another straightforward modification to GP for MLC is the implementation of library building and dictionary learning. Throughout the GP training process, a tremendous range of control laws are explored, and typically, only information about the final controller performance and the GP function representation are used in future iterations. Although it is necessary to obtain a statistically averaged measure of performance, or fitness, for each controller, it is likely that data from individual control experiments will provide rich information about attractor control. In particular, there are transient events in turbulence control, and it is likely that controllers are locally exploring favorable regions of phase space. Understanding controller individual histories at a higher time resolution may provide higher performance, multi-resolution analysis and control.

Multiple operating conditions: In addition, library building may result in adaptive GP for MLC for systems with multiple operating conditions. In cases where there are slowly varying parameters that change the operating conditions of a system, such as temperature, altitude, chemical composition of feed stock, etc., there may be multiple optimal control strategies. Library building and dictionary learning provide a robust strategy to handle these cases with multiple attractors. First, the system parameters are characterized by comparing against a library of previously identified attractors; sparse identification algorithms are particularly promising, and will be discussed below [279, 34, 107, 46, 42]. After the parameters are roughly characterized, the controller jumps to the best GP controller that fits the situation; we refer to this as fast feedforward control. Subsequently, additional slow feedback control can be included to add robustness to un-modeled dynamics and disturbances. However, it is likely that the inherent feedback built into GP controllers will be sufficient for many situations.

Learning robustness from control theory: Finally, there is an opportunity to bridge the gap between the excellent performance obtained using MLC and the rigorous theoretical foundations provided by classical control theory. Investigating issues related to robust performance of MLC will continue to be important to provide theoretical guarantees. These guarantees are particularly important in many aerospace applications, such as boundary layer control on a wing or combustion control in a jet engine, since they allow for controller certification. Although MLC already provides impressive robustness to varying environmental conditions in practice, it may be possible to augment robustness using techniques from classical control theory. It will also be interesting to see how the introduction of filters and dynamic estimators into the MLC framework will impact the control of real engineering-relevant systems. Lastly, there is great interest in the community for *insight* gained from effective MLC.

Genetic programming is a powerful and flexible regression technique, but machine learning has many more techniques that may be used for MLC. There are a number of more general methodological advances that will likely lead to new innovations in machine learning control in the future. The fields of machine learning, control theory, and data science in general, are rapidly growing and steady progress is being made on many fronts. Additionally, innovations in sparse sensing and uncertainty quantification are resulting in improved sensor and actuator placement, which will undoubtedly impact MLC strategies. Here, we provide a high-level overview of some of these exciting recent techniques.

Machine learning applications: The field of machine learning is developing at an incredible pace, and these advances are impacting nearly every branch of the physical, biological, and engineering sciences [30, 92, 168, 190, 194]. This progress is fueled by a combination of factors, including concrete success stories on real-world problems, significant investment of resources by large corporations and governments, and a grass-roots enthusiasm in the domain sciences. Surrounding this movement is the promise of a better future enabled by data science.

Machine learning of the plant: In a sense, aspects of machine learning have already been in use in classical engineering design and control for decades in the form of system identification [150, 174]. The goal of machine learning, like system identification, is to train a model on data so that the model generalizes to new test data, ideally capturing phenomena that have not yet been observed. Recent techniques in machine learning range from the extremely simple and general to highly sophisticated methods. Genetic programming has been used to identify nonlinear dynamical systems from data [32, 220, 240], and sparse variants exist [275]. The sparse identification of nonlinear dynamics (SINDY) algorithm [44] uses sparse regression [266] for nonlinear system identification.

Neural network based control: Machine learning has already been used in a number of control schemes for decades. In fluid dynamics, neural networks are used to identify accurate input–output maps to describe phenomena such as the growth of structures in a boundary layer and reduce skin-friction drag [171] as an add-on to opposition control [66]. Other examples where neural networks have been used to model and control turbulence include [95, 189, 193]. Neural networks constitute a set of bio-inspired algorithms to model input–output behavior using a coupled network of individual computational units, or "neurons". These algorithms were widely celebrated in the 1990s, but fell out of mainstream use, when it became clear that they were unable to solve many challenging problems. For example, although neural networks may be tuned to approximate nearly any input–output function, they are prone to overfitting and local minima in optimization. However, recently, the advent of *deep* convolutional neural networks (CNNs) has brought these algorithms back into the forefront of research [78]. In particular, deep neural networks are being constructed to identify complex phrases to describe scenery in unstructured images [69] and process natural language [132]. Perhaps more famously, these networks have resulted in the Google *deep dream*.

Genetic algorithm based control: Genetic algorithms [76, 137, 122] have also been widely used to identify parameters of both models and controllers for complex systems, such as fluids. Unlike genetic programming, which identifies both the structure and parameters of a model, genetic algorithms only identifies parameters of a model with a fixed structure. However, both genetic algorithms and genetic programming are evolutionary algorithms, relying on advancing generations of individuals and evaluating their fitness. Genetic algorithms have been successful in many applications where the structure is known.

Clustering: In addition to algorithms that identify input–output maps, there is also wide use of machine learning algorithms for classification or clustering data. For example, many complex systems are non-stationary and may be characterized by multiple attractors in different parameter regimes. In these systems, it is often not necessary or even helpful to estimate the full high-dimensional state of the system, but instead it may be sufficient to characterize the coarse system behavior. In this case, classification algorithms [280] (K-means, K-nearest neighbors, LDA, QDA, SVM [242, 253, 259], Decision Trees, Random Forests, etc.) are particularly useful. These classification algorithms are generally categorized into *supervised* algorithms, where the training data is labeled, and *unsupervised*, where the relevant clusters are unknown ahead of time. Classification is especially important in control, as each classification may lead to a control decision. For example, in ultrafast laser systems with multiple operating regimes, sparse classification algorithms have been used to rapidly identify the operating conditions, after which the controller jumps to a pre-determined near-optimal control setting and a slower adaptive control algorithm adds stability and disturbance rejection [42, 107]. In fluids, classification of operating regimes is also being explored [14, 34, 46]

Reinforcement learning: In robotics, the combination of machine learning and control has been proceeding steadily for at least a decade. Techniques for autonomous learning and reinforcement learning [258] have been widely adopted to control robots, both autonomous and tethered. Iterative learning control is used for tracking control, for example of robot arm position [35]. Similar algorithms are also used in brain-machine-interface problems, such as to train prosthetic devices.

The field of machine learning, and data science more generally, relies on a number of key steps: (1) data scrubbing, (2) feature extraction, mining, or engineering, and (3) model building. Each stage is crucial. The argument that machine learning will replace highly skilled domain scientists and engineers is somewhat facetious, considering that these algorithms do not work without a notion of what is *important* in a problem. For example, determining which labels should be used for training and engineering relevant features to distinguish various aspects of the data remain highly creative and critical human tasks in many disciplines.

8.2 System-Reduction Techniques for MLC—Coping with High-Dimensional Input and Output

This textbook contains examples multi-input multi-output (MIMO) control with few inputs and outputs. The learning of control laws can be expected to take longer as the number of inputs and outputs increases. This is particularly true for high-dimensional actuation like surface motion or high-dimensional sensing like real-time particle image velocimetry, like in Sect. 6.1, or image-based feedback. In this section, we outline approaches coping with such high-dimensional input and output.

System reduction: Dynamical systems and control strategies have long benefited from dimensionality reduction, relying on the fact that even complex high-dimensional systems typically exhibit low-dimensional patterns that are relevant for models and control [125, 138]. More generally, it has been observed for decades that nearly all natural data signals, such as audio, images, videos, laboratory measurements, etc., are inherently low-dimensional in an appropriate coordinate system, such as Fourier or Wavelets. In this new coordinate system, many of the coordinates of the data will be negligibly small, so that the original signal can be accurately represented by a sparse vector, containing mostly zeros, in the new basis. This inherent sparsity of natural signals is the foundation for data compression, such as MP3 for audio and JPEG for images. PCA provides a tailored basis for optimal low-rank representation of data.

Compressed sensing: A recent mathematical breakthrough, called compressed sensing [18, 50, 52, 84, 267], has upended the traditional paradigm of collecting and analyzing data. Instead of painstakingly measuring every variable in a system, only to compress and discard the majority of the negligible information in a transform coordinate system, it is now possible to measure significantly less data at the onset and infer the relevant terms in the sparse transform basis.

In the past, solving for these relevant terms from subsampled measurement data would amount to a brute-force combinatorial search. This class of non-polynomial-time, or NP-hard, problem does not scale favorably with the size of the data, and Moore's law of exponentially growing computer power does not scale fast enough to help. A major breakthrough in compressed sensing is an alternative algorithm based on convex optimization using the sparsity-promoting ℓ_1 norm: $\|\mathbf{x}\|_1 = \sum_{k=1}^{N} |x_k|$. It has been shown that under certain reasonable conditions, solving for the vector $\mathbf{x}$ with the smallest ℓ_1 norm will approximate the vector with fewest nonzero entries. The measure of nonzero entries is often called the ℓ_0 "norm", although it does not satisfy the properties of a norm; it is also equivalent to the Hamming distance of the vector from the zero vector, as in information theory. This convex optimization does scale favorably with Moore's law, providing a clear path to solve increasingly large problems in the future.

Another innovation surrounding compressed sensing is the establishment of clear conditions on when the theory will work. First, there must be sufficient measurements to estimate the nonzero transform coordinates; this minimum number of measurements depends on the original size of the vector, the expected sparsity

of the vector in the transform basis, and the level of noise on the signal. Next, the samples must be sufficiently *incoherent* with respect to the vectors that define the sparsifying basis; in other words, measurements should be as orthogonal, as possible, to all of the basis directions. It was shown that random projection measurements of the state (i.e., taking the inner product of the state vector $\mathbf{x}$ with a vector of identical independently distributed Gaussian elements) provide nearly optimal compressed measurements, regardless of the sparsifying basis. This is truly profound, as this enables a *universal* sparse sampling strategy. However, random projections of the state are not necessarily physical measurements, since they still require having access to all of the information in $\mathbf{x}$. A more useful engineering measurement is often the spatially localized point measurement, corresponding to a single physical sensor. Fortunately, single point measurements are incoherent with respect to the Fourier transform basis, so that many signals may be reconstructed from few point measurements.

Regardless of our interest in full signal reconstruction, there are huge implications of sparsity promoting techniques and compressed sensing in engineering. Many of the concepts are applicable more generally to machine learning, modeling and control, including measurement incoherence, working on compressed subspaces of the data, and the structure and sparsity of data in general. For example, if a categorical or a control decision is desired, as opposed to a full-state reconstruction, it is often possible to use randomly sampled measurements and still achieve high classification performance. It is also useful to consider the effect of subsampling or projecting data onto a low-dimensional measurement space. The Johnson-Lindenstrauss theorem [147] and the restricted isometry property (RIP) [51] provide powerful quantification of the distortion of high-dimensional inner products after the data is compressed. These concepts are closely related to unitarity, and they may be used to embed high-dimensional data, or functions, as in the case of genetic programming, in a low-dimensional metric space. Hash function embedding is already being used in fluid flow control [183].

Sparse sensor and actuator placement: The placement of sensors and actuators is a critically important stage in control design, as it affects nearly every downstream control decision. However, determining the optimal placement of sensors and actuators is an NP-hard problem, which does not have an elegant solution, but rather involves a brute-force combinatorial search. In particular, this strategy does not scale well to even moderately large problems of interest. To compound the difficulty, often sensors may be quite expensive, as in the case of putting tracers in the ocean or human vaccinators in a foreign country. Even if sensors are inexpensive, processing the vast streams of data from a sensor network may be intractable, especially for mobile applications which are power and computationally limited.

Recent advances in compressed sensing (see above) are poised to revolutionize sensor and actuator placement for control problems. From an engineering perspective, the ability to solve NP-hard optimization problems with convex optimization routines is transformative. Fortunately, the sensor placement problem may be thought of as a compressed sensing problem under certain conditions, as we are

trying to find the few best locations to maximize a well-defined objective function. Sparse sensor optimization using compressed sensing techniques has already been implemented for categorical decision making in complex systems [14, 39]. Extending these methods to optimize sensors and actuators for a control objective is an important area of current research. Finally, extending the compressed sensing framework to dynamic point measurements, such as Lagrangian tracer elements, has huge potential for the fields of oceanographic and atmospheric sampling.

8.3 Future Applications of MLC

MLC can be expected to address many applications from everyday life to industrial production. Many control problems have a well-defined cost function, a finite number of sensor signals (outputs) which monitor the state and a finite number of actuation commands (inputs) which shall improve the system performance. In short, we have a multiple-input multiple-output (MIMO) plant and search for a control logic which optimizes a cost function.

Most factory processes fall in the category of a MIMO control problem. Åström and Murray [223] provide an excellent introduction in the world of control problems. The key strength of MLC comes into play when the plant behavior departs from linear dynamics and eludes current methods of model-based control.

Feedback control of turbulence is a grand challenge problem for control design as the plant incorporates three key difficulties: (1) high dimension, (2) nonlinearity, and (3) potentially large time-delays (see Chap. 1). Turbulence increases the power required to pump gas and oil in pipe-lines and is a major source of drag of ground, airborne, and maritime transport. Wind turbulence increases maintenance costs of wind turbines via gusts and creates dangerous situations for ground- and airborne transport. In contrast, turbulence has desirable mixing properties in heat exchangers, combustion and chemical processes.

Effective turbulence control can contribute to a key challenge of modern society: renewable energy production and reduced energy consumption. The world's primary energy supply has more than doubled from 6,106 m[1] in 1973 to 13,371 m in 2012 [3]. The consumption in 2012 was $1{,}555 \times 10^{17}$ J corresponding to an average rate of 17.75 TW or 2.5 kW per person on this planet. This corresponds to one powerful heater per person operating day and night. On the downside, the environmental cost of energy production is an increasing challenge. Part of this cost is immediately felt as smog in the city or as noise near streets, railroads and airports. Other costs are indirect long-term consequences: Coal and fuel consumption provided 60.4 % of the 2012 energy supply [3] and are particularly problematic because the CO_2 emissions affect our climate. Our very existence is threatened by man-made global warming. The Nobel committee has appreciated this fact by awarding the 2007 Peace Nobel Price to the International Panel on Climate Change and Al Gore "for their efforts to

[1] Mtoe means million ton of oil or equivalent (11.63 TWh).

build up and disseminate greater knowledge about man-made climate change, and to lay the foundations for the measures that are needed to counteract such change" [1].

On the demand side, 20 % of the worldwide energy consumption is used for transport, mostly by oil [82]. This demand corresponds to a cube of oil with each side measuring nearly 14 km. In 2014, the USA used 28 % of its energy consumption for transport with 92 % provided by fuel. The economic price of world transport amounts to 10 % of the world gross domestic product (GDP) in 2011 [82].

The relevance of sustainable traffic may be measured by the fact that the European Research Council supports over 11,000 research grants on this topic. In following, we describe several transport-related opportunities for MLC:

Drag reduction of trucks: About one third of the operating costs of trucking companies is for fuel. The average profit margin is 3–4 % in the USA. Thus, a 5 % reduction on fuel costs, for instance by active flow control, spells a substantial increase of the profit. Passive means, like vanes for the rear side are already for sale. Active control solutions have already been tested on the road and have proven to save 5 % fuel already under real-world traffic conditions. Experiments with small-scale vehicles demonstrate the possibility of a 20 % drag reduction [19, 212] at a fraction of the energy cost.

Drag reduction of cars: Personal cars are less subject to economic considerations as in the trucking industry. However, the European union has identified the development of sustainable transport as a cornerstone challenge for the next decades. To encourage efforts in that directions, European union legislation [202] has set mandatory emission reduction targets for the car industry. The new standards impose a reduction of 30 % of greenhouse gas emissions and corresponding fuel consumption for new cars by 2020. Meanwhile, the transportation industry is a strategic economic sector in Europe accounting for 4.5 % of the total employment (10 million people). However, the market for new vehicles in European countries is declining, whereas that of emerging countries is rapidly growing. To remain competitive in this market, European car manufacturers must develop disruptive technology concepts to produce safer and more sustainable vehicles.

Improving the aerodynamic performance of road vehicles by flow control can help fullfill these requirements, in particular at highway speeds. At highway speeds, overcoming aerodynamic drag represents over 65 % of the total power expense [140, 184]. This explains significant aerodynamic improvements on automobiles during the last decades. Most existing flow control approaches to reduce vehicle drag on commercial road vehicles are passive [118]. They rely on aerodynamic shaping, such as well-rounded contours (especially at the front), rear slope angle, and add-on devices [70]. Such passive approaches, however, are restricted by design and practical considerations and cannot be 'turned off' when not needed or modified to adapt to changing conditions.

Recently, significant research has been focused on active flow control (AFC) solutions. Cattafesta and Shelpack [54] give an extensive overview of possible actuation mechanisms, whereas [65] present the most common AFC approaches on bluff bodies. Of the many available approaches, a large subset is considered as

an academic exercise, such as rotary [27, 81], streamwise [58], and transverse [31, 53] oscillation of a bluff body. Of the practical and realistic AFC mechanisms, many were investigated on the Ahmed body [4], such as synthetic jets [208], pulsed pneumatic actuation [149, 226], microjects with steady blowing [10], and Coanda blowing [104, 116]. AFC on heavy truck vehicles were also investigated, such as synthetic jet actuators [96], and Coanda blowing [97, 98].
A drag reduction of 23 % has been achieved on a truck model with steady Coanda blowing [212]. The saved towing power was 3 times the invested actuation energy. A better actuation efficiency of 7 has been achieved with high-frequency Coanda blowing for a blunt-edged Ahmed body by [20] resulting in a 18 % drag reduction. Both studies were based on a working open-loop control. The implementation of MLC is the logical next step after a working AFC. Machine learning control has recently been shown to improve existing control of an Ahmed body and a small-scale Citroën model in OpenLab PPRIME/PSA, France.

Safety of cars and trucks under wind gusts: Side-wind stability is important for passenger comfort and safety of all ground vehicles. It is particularly critical and a safety issue for vehicles with large projected side area, such as trucks and buses. Accidents linked to crosswind have been reported by several governmental agencies [48]. Side-winds and side-gusts originate from different sources such as weather, surrounding traffic, or the topology of the terrain next to the road. Several numerical [126, 145] and experimental [59, 118] studies were conducted to understand the dynamics of such flows. Both steady [15, 139] and unsteady [59, 268] crosswind investigations have been researched. The findings of these aforementioned studies and others can be summarized as follows:

- The driving stability decreases with increasing speed.
- A reduction of the lateral projected area in the back will reduce the rear side force and thereby the yaw moment. However, this relation is not valid for well-rounded rear-end configurations (C- and D-pillars).
- A-pillar radiusing has a large influence on the yaw moment.
- The center of pressure of the aerodynamic forces has a large impact on the vehicle stability.
- Directional stability depends more on yaw moment than on the overall side force.
- For a wide range of vehicles, the yaw moment increases approximately linearly with the yaw angle up to 20°.

Despite the numerous studies on side-wind effects and driving stability, very few have tackled the issue using AFC. To our knowledge, only Englar [97, 98] and Pfeiffer et al. [212, 213] applied Coanda blowing on the 4 rear edges of a truck to reduce drag and to control the yaw moment. Whereas Englar [97, 98] only implemented AFC as an open-loop, Pfeiffer et al. [212, 213] successfully applied it as a closed-loop control. The latter were able to achieve a leading 23 % drag reduction and a complete authority over the yaw moment. However, further improvements on the previous results can be achieved:

- Distributed actuation comprising *all* locations with demonstrated effect on drag and yaw;
- Distributed sensing providing real-time flow information;
- Employing very general, nonlinear and robust control laws comprising periodic, multi-frequency, and adaptive in-time closed-loop control;
- Reduction in actuation power for both drag minimization and yaw control.

All these new possible improvements have two themes in common, safety and efficiency. MLC can optimize open-loop, adaptive and closed-loop control by using harmonic functions as additional arguments of the control laws.

High-lift configuration of passenger aircraft: Civil aircraft are highly optimized for cruise conditions. Typical lift-to-drag ratios are around 17. This means 1 Newtons of thrust lifts 17 N of weight at a cruise speed around 850 km/h. During landing the main goal is not a good lift-to-drag ratio but a steep descent at low velocity to reduce the noise signature on ground. This is achieved with a high-lift configuration through a reeled-out flap. Active control can prevent early separation and reduce the size of these flaps for a given lift. Thus the weight of the aircraft is reduced during cruise resulting in lower propulsion power and thus reduced fuel consumption. This opportunity is being pursued by the two major passenger aircraft producers, Airbus and Boeing, and is the subject of intense research.
The economic aspect of reduced fuel consumption may be appreciated from the following data: Average profit margins are between 1–2 %. Fuel is with a 28 % contribution the most important operating cost of passenger airlines (2014, US airlines). Fuel may represent around 40 % of the maximum take-off weight. The airline not only saves fuel costs with active flow control. More importantly, it may replace 80 kg fuel by a paying passenger. In a 100 passenger airplane one passenger amounts to 1 % of the income and thus accounts for a significant change in the profit margin.

Nacelle of a passenger aircraft: During take-off, the engines produce about 6 times the thrust during cruise and can lift around 30 % of the passenger aircraft. The large nacelle prevents the recirculation bubble from extending to the compressor, which would result in a dangerous loss of thrust. The large nacelle is only needed for about one minute of maximum thrust during take-off. Active flow control at the inlet may reduce size the nacelle and thus reduce weight. This is another opportunity to save fuel.

Cavity of aircraft and trains: Wheel cavities of aircraft are a major noise source during landing. Similarly, cavities between train wagons produce noise. This noise can be significantly mitigated by feedback flow control [232]. The cavity noise reduction is a subject of intense research.

Drag reduction of ships: All moving ships create a wave pattern on the water. These waves require energy to be created and cause additional drag on the ship. For small ships, wave drag may be 80 % of the total drag! One may conceive active vanes at the front of the ship to mitigate part of the wave drag. For full-scale tankers, the wave drag reduces to 20 % of the total drag but is still substantial. An expected

5 % reduction of drag is considered as a threshold value for engaging in a new expensive ship design. The remaining portion of the drag is due to skin-friction.

Skin friction reduction: About 80 % of the propulsion power of tankers and ocean liners are due to skin friction drag. Future ships may profit from drag reduction due to hydrophobic surfaces. Similarly, skin friction is the major drag source of passenger airplanes. Riblets have been shown to reduce skin friction drag by up to 11 % and are actively pursued by aircraft manufacturers. Skin friction may also be mitigated with distributed local suction and blowing. Drag reductions of 25 % have been reported in low Reynolds number simulations [66]. At larger Reynolds numbers the reduction decreases. The distributed suction/blowing acts on plus unit scale to fight sweeps and ejections. An experimental realization would require myriad of actuators and sensors and will not be feasible for a long time. A more realistic active control is oscillatory wall motion. Numerical simulations [66] with high frequency oscillation show a 40 % drag reduction. In experiments at RWTH Aachen, a 5 % drag reduction was obtained. Feedback control with wall motion has hardly been explored.

We briefly mention other opportunities of feedback turbulence control and hence MLC.

Rockets: Solid fuel rockets may develop acoustic instabilities in the interior body which may destroy the rocket. Evidently, this is an opportunity for closed-loop control.

Wind turbines: Wind gusts create sudden loads on the rotor hub and bearings of a wind turbine. These forces reduce the mean-time between failure and thus make the maintenance effort more costly. Feedback controlled flaps at the trailing edge of the airfoil can reduce the unequal loading. The development of suitable hardware solutions and corresponding control logic is an active area of research.

Water turbines: Many underwater flows, such as in a river, may produce a more steady loading. Preliminary results of the authors show that accelerating and decelerating a cross-flow turbine to excite unsteady fluid forces can increase the energy efficiency by 80 % [256].

Gas turbines: Lean combustion reduces CO_X and NO_X emission. However, in the lean limit, the danger of blow out or combustion instabilities increases. Closed-loop mitigation of combustion instabilities via the change of fuel supply or fluid mechanic actuators are a demonstrated opportunity towards greener energy production [114].

Combustion engines: The internal combustion engine is ubiquitous in many modes of transportation, providing propulsion for most motorbikes, cars and trucks and convert chemical energy through combustion in mechanical energy via a periodic piston operation. Despite the periodic piston movement, the flow exhibits large cycle-to-cycle variations. Some cycles are good for combustion, particularly if the fuel is well mixed in the piston by turbulent mixing. Some cycles are less efficient. An ongoing effort of all engine producers is to reduce these cycle-to-cycle variations and to stabilize a uniformly good mixing. Fuel injection or

mechanical actuation may be elements of a feedback stabilization for a better operation.

Pipe-lines: The power needed to pump oil through pipe-lines can be significantly reduced by the addition of polymers. One polymer molecule per 1 million oil molecules may reduce drag by 40 %. Closed-loop turbulence control can hardly compete with this efficiency. However, there are still large opportunities for closed-loop control for gas-liquid-solid mixtures close to an oil plant.

Air conditioning: The goal of air conditioning is increased comfort. Ventilation shall refresh the air at appropriate temperature without unpleasantly felt air streams. MLC may learn an optimal air stream management based on sensor information.

Air conditioning causes other problems in hospitals. Airborne viruses need to be neutralized at active surfaces in the ducts. Such surfaces cause additional pressure losses and require thus a larger ventilation power. The mixing problem may be formulated as follows: a fluid particle with a potential virus must be sufficiently close to the active surface at least once with near certainty. However, multiple encounters with the active surface would lead to unnecessary pressure losses. This is an exciting mixing problem for closed-loop turbulence control.

Pharmaceutical and chemical processing: The industrial production of food, e.g. chocolate, may happen in vessels with diameters up to 5 m. Different constituents need to be uniformly distributed in these vessels. The mixing may be closed-loop controlled by the inlets and mechanical mixing and monitored by cameras. Numerous production processes require a good mixing in vessels or in pipes. This is another exciting closed-loop mixing problem for which MLC is predestined. The effect of mixing is highly deterministic yet hardly predictable. Mixing has largely eluded an intuitive understanding.

The previous examples have focused on turbulence control. The application of MLC requires only a finite number of actuators and sensors. In addition, the cost function needs to be strongly related to the control law. In other words, the controlled plant exhibits statistically stationary behavior. Many complex systems fall in this category. Examples include

Predictions for solar and wind energy: The operation of an electricity network shall satisfy the unsteady demand and prevent overproduction as the storage capacities are limited. Hence, the production of renewable energy needs to be predicted from weather and other data to identify potential supply-and-demand problems early in time. This is an estimation problem which can also be solved with genetic programming [220]. Here, the input is the weather and other data, the output the energy production and the cost function the difference between estimated and actual production.

Short-term trading at stock markets: Here, the plant is the stock market, the input the trading action of one player, the output the supply and demand curve, the control logic the automated trading, and the objective function the daily profit. Long-term trading is far more challenging for automated learning as the assumption of statistical stationarity is not satisfied. The Dow Jones index, for instance, has the

long-term tendency to increase. Companies may be founded, new economies may appear, or old economies may disappear, etc.

Dog training: All dogs, in fact all animals, are trained with rewards (and punishment). Intriguingly, maximum animal performance is not obtained with a monotonic performance-reward curve. There exist many education concepts how the reward should depend on animal performance. MLC might yield novel concepts.

These examples show, pars pro toto, the many MLC opportunities for MIMO plants which are of large relevance in industry or everyday life. As a disclaimer, we note that there also exist many problems which are more suitable for a model-based control strategy. For instance, the effect of control surfaces on the motion of an aircraft is well represented by a linear model for the usual operating envelope. Control design based on these models reliably cover many operating conditions. Thus, the robustness of control laws may be estimated. Continuing to establish a deeper connection between MLC and traditional control analysis will be essential to ensure broad adoption of these methods, as discussed in Chap. 4.

The flow around the airplane is turbulent but the time-averaged effect of myriad of vortices yields a nearly linear relationship between motion of the control surface and aerodynamic force. Taking the time scale of the vortices as reference, the model-based operation of the control surface can be considered as slow adaptive control. Mathematically, the situation might be compared to statistical thermodynamics where myriad of gas molecule collisions, i.e. strongly nonlinear events, can still lead to a linear relation between pressure and temperature in a finite volume by statistical averaging.

8.4 Exercises

Exercise 8–1: Investigate each of the following machine learning clustering algorithms and think about a representative data set that works well with the method and a representative data set that does not work with the method: K-means, linear discriminant analysis (LDA), support vector machines (SVM), decision trees.

Exercise 8–2: Consider the following signal, which is obtained as the sum of two sine waves:

$$f(t) = \sin(114\pi \text{ Hz } t) + \sin(1042\pi \text{ Hz } t).$$

The Shannon–Nyquist sampling theorem [201, 248] indicates that one must measure at 1042 samples per second to accurately reconstruct this signal. However, because the signal is *sparse* in the Fourier domain (i.e., only two Fourier modes are active at 57 and 521 Hz), we may reconstruct this signal from dramatically under-sampled measurements.

In this exercise, create a time-resolved signal $\mathbf{f}$ by sampling at 1042 samples per second for 10 s. Now sample this signal randomly for 10 s with an average sampling rate of 128 samples per second. You may think of the sampled vector $\mathbf{f}_s$ as the output after applying a measurement matrix $\mathbf{C}$ to the full time-resolved signal:

$$\mathbf{f}_s = \mathbf{C}\mathbf{f},$$

where $\mathbf{f}$ is a time-resolved vector of the signal: $\mathbf{f} = \left[f(\Delta t)\ f(2\Delta t) \cdots f(N\Delta t) \right]^T$, and $\mathbf{C}$ contains random rows of the $N \times N$ identify matrix.
Finally, we may solve for the active Fourier modes in the compressed sensing problem:

$$\mathbf{f}_s = \mathbf{C}\boldsymbol{\Psi}\hat{\mathbf{f}},$$

where $\boldsymbol{\Psi}$ is the inverse discrete Fourier transform. Use a convex optimization routine, such as `cvx` in Matlab®, to solve for the sparsest vector $\hat{\mathbf{f}}$ that solves the underdetermined system above.

Exercise 8–3: Consider the LQR stabilization task from Exercise 4–1. Here, we will define a metric on the space of controller functions in an attempt to improve the convergence time of genetic programming control. First, create one hundred random states $\mathbf{a}$ where each component a_1 and a_2 are Gaussian distributed about $a_1 = a_2 = 0$ with unit variance. Now, for each controller, $b = \mathbf{K}(\mathbf{a})$, evaluate the control law at these same one hundred random states, and store the value in a 100×1 vector $\mathbf{b}_{\text{hash}}$. It is now possible to use these vectors to construct a proxy distance between two controllers $b = \mathbf{K}_1(\mathbf{a})$ and $b = \mathbf{K}_2(\mathbf{a})$:

$$d(\mathbf{K}_1, \mathbf{K}_2) \triangleq \|\mathbf{b}_{\text{hash},1} - \mathbf{b}_{\text{hash},2}\|_2. \tag{8.1}$$

Using this induced metric, modify your genetic programming search strategy to improve the convergence rate by reducing the number of redundant controller functions tested. For instance, when mutating, you might check if a new individual is sufficiently similar to individuals from a previous generation, and impose additional distance criteria to improve exploration and exploitation.

Exercise 8–4: Pick a future application of MLC from Sect. 8.3 and explore the current control approaches being applied tho this system. What are the challenges with applying MLC to this problem?

8.5 Interview with Professor Belinda Batten

Belinda Batten is Professor of Mechanical Engineering at Oregon State University, OR, USA. Professor Batten is the Director of the Northwest National Marine Renewable Energy Center (NNMREC), a collaboration between Oregon State University, the University of Washington, and the University of Alaska Fairbanks. NNMREC supports wave, tidal, offshore wind, and in-river energy harvesting through research and testing. The consortium was established by the U.S. Department of Energy to facilitate the development of marine renewable energy technologies via research, education, and outreach.

Professor Batten is internationally renowned for her research in modeling and control of distributed parameter systems, especially for her development of computational algorithms for reduced-order controllers. Her current research projects include mathematical modeling and control of autonomous vehicles and wave energy devices. She has raised significant funding to support her research from DARPA, DoD, NSF, and DOE.

Professor Batten has been a Program Manager for dynamics and control at the Air Force Office of Scientific Research, after which she was elected member of the Scientific Advisory Board for the U.S. Air Force. She was also a professor of mathematics at Virginia Tech, and served as Department Head for Mechanical Engineering from 2003–2007 and as Head of the School of Mechanical, Industrial and Manufacturing Engineering from 2007–2011 at OSU. Her research was honored by national and international awards, for instance by the prestigious Alexander von Humboldt fellowship.

Authors: You are a leader in the field of marine renewable energy and have developed numerous innovative technologies and control solutions. What were the main trends in marine renewable energy, especially related to fluid modeling and control, in the past decade?

Prof. Batten: I will focus my comments on wave energy, as that is where my primary expertise lies. Prior to this past decade, there has been a good amount of research on developing models of wave energy converters, and some work on controlling them. The modeling work typically leveraged the great volume of

literature on hydrodynamics of ocean structures. The work on control has typically been model-based optimal control, and often has ignored the "control costs" that arise through actuation. That is, the results on the amount of energy produced through active control were often highly theoretical.

In the last ten years, more work has been done in experimental validation of computational models. Computational codes for simulating wave energy converters continue to be developed. Recently, codes are being developed for design of marine energy arrays, both wave and tidal current energy converters.

Some researchers are working on high fidelity fluid-structure interaction models for wave and tidal energy converters. These codes are not currently suitable for control design and computation as they require millions of state variables. As computational capabilities continue to progress, these codes may be more useful in the control regime, but at this point, model reduction of some type is required to use them for purposes other than simulation, e.g., for control or optimization.

Authors: You were an early advocate of merging control theory with fluid mechanics. Do you see any key similarities or differences in the current efforts to bring machine learning into the fluid dynamics community?

Prof. Batten: One of the challenges in merging control theory with fluid dynamics in the early years was learning each others' language. Basic terms like "control" were used differently by control theorists and by fluid dynamicists. I remember giving a kick-off talk at a meeting of the two groups, laying the foundation for the discussion—including the typical nomenclature for mathematically describing a fluid system with a control. One of the fluid dynamicists responded, "you mean my entire life's work in actuator modeling is reduced to a B matrix?" So, part of getting the two groups to work together was facilitating a common understanding of what problems were viewed as easy, hard, unanswered, unanswerable, tractable, etc. Once these communities began to interact with each other, great progress was made, and a lot was learned.

I hope that bringing machine learning into the community is a little easier because some of the same people in the controls and fluid dynamics communities are involved, and collaborative work across disciplines starts with relationships. That said, I remember years ago when a seminar speaker talked about using neural networks to develop a model for the Navier-Stokes equation; several in the audience muttered about why anyone would throw out the physics.

And that illustrates the crux of the issue. There are some really hard problems for which developing high fidelity physics based models for control is not feasible—at least not with today's computational limitations. The fluid-structure interaction modeling of a wave energy converter in the ocean is one of these problems. So, to develop machine learning approaches that we can test and verify on smaller tractable problems can lead to solutions and insights on the more complex ones for which developing physical models for control is infeasible.

Authors: Could you comment on the societal impact of marine renewable energy, and how you see fluid mechanics and control helping to address the key challenges?

Prof. Batten: The reality is that the world cannot continue to rely on fossil fuels to meet energy needs. The supply of fossil fuels will eventually be exhausted, and in the meantime, the impact of fossil fuels on our environment is increasingly destructive. While people may debate about climate change, the acidification of our oceans is measurable, and the consequences of delaying the move to renewable energy are sobering.

In my opinion, replacing fossil fuels will require a portfolio of renewables that address particular needs of place. Marine renewables could be an important addition to this portfolio; this energy source is highly predictable, always present, and located within 50 miles of 50 percent the world's human population. Tides are predictable 100 years in the future; waves are predictable 84 h out. Those horizons make it simple for a utility to manage electricity coming onto the grid, and one can control marine energy converter arrays to deliver what is needed. Unlike wind and solar, these energy sources are always present.

The key challenge for marine renewable energy is becoming cost competitive with other renewable energy sources. Control of marine energy converters has typically been directed toward maximizing energy extraction. Maximizing energy produced indeed contributes to a lower cost. However focusing solely on that objective ignores the reduction in life expectancy of a wave energy converter or increase in operations and maintenance costs due to fatigue or failure of converters under extreme events. It may be that for certain converters, control should also be used to minimize fatigue. At Oregon State University, my colleague Ted Brekken has been working with his students to develop life extending control to address this concept.

If computationally tractable models of fluid structure interactions of wave energy converters—especially under extreme wave conditions—could be developed, such models could be used to better predict the reliability and survivability of devices. Alternatively this topic could be fertile research ground for machine learning. Better understanding reliability and survivability of devices could be leveraged to understand how to lower the cost of energy of the wave energy converters, thus contributing to adoption of marine energy.

Authors: In other fields, such as astronomy and particle physics, there are well-funded grand-challenge problems that the community tackles in a coordinated effort. Is there a need for similar larger scale collaborative efforts in fluid flow control? If so, what do we need to do in order to get fluid dynamics to a similar position in terms of funding and support?

Prof. Batten: I think great progress has been made in fluid flow control since the early 2000 s when I was challenging the communities to come together and work collaboratively on these problems. A lot has been learned about what control can do and what kinds of models and computational codes are useful. I think the big challenge problem for fluid dynamics related to marine energy is the fluid-structure interaction models, suitable for control design and optimization. A workshop that pulls community experts together to define other such problems might be an important next step.

Authors: As the director of the Northwest National Marine Renewable Energy Center (NNMREC), could you comment on some of the grand challenges in the marine renewable energy sector, and how they may be impacted by machine learning methods in the coming years?

Prof. Batten: If there is a fundamental grand challenge in marine renewable energy, it is how to lower the cost of energy of these technologies to make them competitive. While there are many subsystems within a marine energy array with highly technical underpinnings and each contributes to the overall cost, it is necessary to take a system level viewpoint of this problem. Applying machine learning to a system level performance of a marine energy array could have interesting outcomes.

Authors: This chapter considers future developments and extensions to machine learning control. If you were giving advice to a new student, what direction would you point them in for important research problems, and what tools would you want them to be equipped with?

Prof. Batten: I'll leave the advice about new research directions in machine learning to the experts in that area. Regarding the tools that I want students to know, I believe that it is important for students interested in controls, be it learning based methods or model based methods, to understand the value and limitations of the various approaches. When it comes down to it, any control approach is a tool, and it's important to know when to use a screw driver and when to use a hammer. There are times when you could use either, but one is probably better for the job than the other.

Authors: We look forward to your continued breakthroughs in control and renewable energy, and we thank you for this interview!

Glossary

I learned very early the difference between knowing the name of something and knowing something.

Richard Feynman

Active control A controller that expends energy to accomplish a control task. For example, an automobile cruise-controller will actively control the fuel and brakes to regulate forward velocity.

Adaptive control A controller that modifies its action to optimize performance or account for varying system parameters or externally varying conditions.

Actuator A device that modifies the system according to the control input. The actuation effect is typically modeled by the structure of the **B** matrix in a control system.

Crossover A genetic operation where two individuals exchange a portion of their expression, thereby increasing diversity of the future generation. Crossover tends to exploit successful patterns in the parent individuals to produce more fit offspring in future generations.

Closed-loop control The process of controlling actuators based on sensor measurements.

Clustering Identifying groups of similar data. If the data are labeled, this is called *supervised*, and if the data is not labeled, it is *unsupervised*.

Coherent structures A structure in a dynamical system that remains coherent, or spatially correlated, for some time; here spatial correlation typically refers to the state of the dynamical system. In fluids, coherent structures often refer to persistent vortical structures that stay intact despite turbulent fluctuations.

Control theory The theory of processes which modify a system for an engineering goal, often with actuators and sensors.

Cost function A function that quantifies the cost or penalty of a given control law or estimator.

T. Duriez et al., *Machine Learning Control – Taming Nonlinear Dynamics and Turbulence*, Fluid Mechanics and Its Applications 116,
DOI 10.1007/978-3-319-40624-4

Disturbance An external perturbation to the system that passes through the dynamics, also known as *process noise*. Disturbances are typically seen as unwanted perturbations that degrade performance, such as unreliable or unpredictable environmental conditions.

Dynamical system A model for how a state evolves in time, possibly in response to an actuation signal and external disturbances. The dynamical system may have an *output equation* that consists of a set of measurements of the state and actuation signal. A dynamical system may either be *nonlinear* or *linear*, and they are often represented as a system of ordinary differential equations.

Elitism A genetic operation whereby the best individual(s) from a generation are automatically copied to the next generation without probabilistic selection based on fitness.

Estimator A dynamical system that estimates the state of another dynamical system from a limited set of measurements. See *Kalman filter*.

Evolutionary algorithm An algorithm that adapts over time (generations) according to a fitness or cost function.

Exploitation The process in an evolutionary algorithm whereby successful patterns in individuals of a given generation are *exploited* to produce more fit individuals in the next generation. Crossover is a genetic operation that promotes exploitation.

Exploration The process in an evolutionary algorithm whereby new, unexplored patterns are sought out for individuals in future generations. Mutation is a genetic operation that promotes exploration.

Expression tree A function or expression that may be expressed as a tree, where each node represents a unary or binary mathematical operation, such as $+$, $-$, $\times$, $/$, sin, cos, etc. Function trees may be quickly evaluated using recursion.

Feedback control A closed-loop control architecture, whereby a downstream sensor measurement is fed back to an upstream actuator.

Feedforward control A control architecture, whereby an upstream sensor measurement is fed forward to a downstream actuator. Often feedforward control is used to measure an incoming disturbance and apply preventative control downstream; this is known as *disturbance feedforward control*.

Fitness function A function that measures the success of an individual expression in achieving some goal. Often inversely related to the *cost function*. In genetic algorithms and genetic programming, the fitness function determines the probability that an individual will be selected for the next generation.

Flow control The process of modifying a fluid system to achieve some engineering goal. This is often accomplished by *active* control, whereby energy is expended to actuate the flow. High-level goals often include lift increase, drag reduction, mixing enhancement, and these goals may be achieved by physical mechanisms such as relaminarizing a boundary layer or stabilizing an unstable shear layer.

Frequency crosstalk A phenomena in nonlinear dynamics where a signal or behavior at one frequency can effect or modify a signal or behavior at another frequency. In a linear system, input forcing at a single fixed frequency will result in an output response with the same frequency and a new magnitude and phase.

However, in a nonlinear system, forcing a system at a single fixed frequency may result in an output response where multiple frequencies are modified through nonlinear coupling mechanisms.

Generation A collection of individuals to be tested in a genetic algorithm or in genetic programming. The performance of these individuals are evaluated, and each individual's *fitness function* determines the probability of advancing to the next generation via the *genetic operations*.

Genetic algorithm An evolutionary algorithm to optimize the parameters of an expression with a pre-specified structure.

Genetic operation A set of operations to advance individuals from one generation to the next. These operations include *elitism*, *replication*, *crossover*, and *mutation*. Individuals are selected for these operations depending on their *fitness function*.

Genetic programming An evolutionary algorithm to optimize both the structure and parameters of an expression or a function.

Genetic programming control The process of discovering an effective control law by using genetic programming to construct functions relating sensor measurements to an actuation signal.

Individual A candidate expression in a genetic algorithm or genetic programming. Each individual is tested, resulting in a fitness function that determines its probability of propagating to the next generation.

Kalman filter A dynamical system that estimates the full-state of another dynamical system from measurements of the sensor outputs and actuation inputs. The Kalman filter is an optimal state estimator for a linear system with additive Gaussian process and measurement noise.

Linear system A dynamical system where superposition holds for solutions. This implies that doubling the initial condition and the control input signal will result in exactly twice the output. Often, the system will be a *linear time invariant* (LTI) system, so that the dynamics may be characterized entirely by linear operators (matrices).

Linear quadratic Gaussian (LQG) An optimal sensor-based feedback control law that consists of a linear quadratic regular feedback law applied to the full-state estimate from a Kalman filter. The LQG controller is optimal for a linear system with the same quadratic cost function as in LQR and additive Gaussian white process and measurement noise of known magnitude.

Linear quadratic regulator (LQR) An optimal full-state feedback control law to stabilize the state of a linear system while not expending too much actuation energy. LQR is optimal with respect to a quadratic cost function that balances deviation of the state and control expenditure.

Linearization The process of approximating a nonlinear dynamical system by a linear dynamical system near a fixed point or periodic orbit by truncating a Taylor series of the dynamics at first order. Linearization is valid for small state perturbations in a small neighborhood of the fixed point or periodic orbit.

Machine learning A set of techniques to automatically generate models from data that may be generalized and improve with more data. Machine learning is often applied to high-dimensional data where it is difficult to identify patterns and

relationships in the data. Common techniques include classification and regression tasks, and these may be either supervised by expert input or unsupervised algorithms.

Machine learning control The process of determining effective control laws through the use of machine learning methods. Controllers are *learned* through a guided process that is informed by measured performance data as opposed to being derived from first principles or optimization routines.

Mean-field model In fluid mechanics, a mean-field model is a low-order Galerkin model linking base-flow changes with fluctuations. In the most simple case, a mean-field model describes the soft onset of an oscillation via a supercritical Hopf bifurcation. This is also referred to as *Watson-Stuart model* or *weakly nonlinear theory* and implies the famous *Landau equation* for a supercritical Hopf bifurcation. Generalized models may incorporate several frequencies and do not require the closeness of a bifurcation.

Measurement noise Noise that is added to the output equation of a dynamical system, thus not being affected by the dynamics. Often simply referred to as *noise*.

Model A mathematical expression that describes a system. Often, a model is derived from first-principles by physical arguments, such as conservation of mass, momentum and energy. Alternatively, a model may be derived from observational data about the system, as in statistics, system identification, and machine learning. Dynamic models are often represented as a coupled system of differential equations relating the various quantities under observation.

Model reduction The process of approximating a high-fidelity model with a smaller, more computationally efficient model in terms of fewer states. Model reduction is an important step when controlling high-dimensional systems, since determining and evaluating control laws based on high-fidelity models is often computationally prohibitive. Moreover, control performance may be limited by the latency of a control decision, so faster decisions resulting from reduced-order models are often beneficial.

Mutation A genetic operation where a portion of an individual in the current generation is randomly altered to produce a new individual in the next generation. Mutation tends to promote exploration in the search space of possible individuals.

Open-loop control A method of control that specifies a pre-determined input sequence without correction or adaptation via sensors. A common method of open-loop control is periodic forcing.

Neural network A network representation of an input–output function that attempts to mimic the computational flexibility observed in biological networks of neurons. A neural network consists of a group of individual computational components, or neurons, that are connected in a network or graph structure to perform some computation. Neural networks are typically characterized by their *adaptability* and *trainability* to new stimulus.

Noise A quantity that varies randomly in time and is added to some variable in a dynamical system. If added to the state equation, it is also known as a *disturbance* or *process noise*, and if added to the output equation, it is also known as *mea-*

surement noise. Noise is often assumed to follow a Gaussian white noise process, although it may also be correlated or *colored*.

Nonlinear system A system of equations or a dynamical system that is characterized by nonlinear dynamics. As opposed to a linear system, a nonlinear system does not satisfy superposition of solutions, resulting in complex behavior, including frequency crosstalk and chaos.

Passive control A controller that modifies a system without energy expenditure. Examples include vortex generators on wings that passively delays flow separation over a wing.

Plant In control theory, the plant refers to the model system being controlled along with the actuator.

Process noise Noise that is added to the state equation of a dynamical system, thereby passing through the dynamics. Also called a *disturbance*.

Real-time control A control law that modifies the system on a time scale that is fast compared with the natural time scale. Also referred to as *in-time* control.

Reduced-order model An approximate model with fewer states than the full high-fidelity system. Reduced-order models are often desirable in the control of high-dimensional systems, such as fluids, to reduce computational overhead, leading to faster, lower-latency control decisions.

Regression A statistical model that relates multiple variables from measurement data. The method of least squares is a simple *linear* regression that determines a best-fit line relating data. Least-squares regression may be generalized to higher dimensions in what is known as the principal components analysis (PCA). More generally, nonlinear regression, dynamic regression, and functional or semantic regression are used to determine complex and possibly time-varying relationships between variables. Regression is commonly used in both *system identification*, *model reduction*, and *machine learning*.

Regulator A control law that maintains a set-point in the state variable. See *linear quadratic regulator*.

Replication A genetic operation where individuals are copied directly from one generation to the next. These individuals are selected probabilistically based on their fitness, so that the most fit individuals are more likely to advance.

Reynolds number A dimensionless quantity that measures the ratio of inertial and viscous forces in a fluid. The Reynolds number may also be thought of as a rough measure of the ratio of the size of the largest vortices and the smallest vortices in a flow. Thus, a volcanic eruption will constitute an extremely high Reynolds number flow, as there are both very large and very small eddies.

Robust control The field of control theory where controllers are designed to be inherently robust to model uncertainty, unmodeled dynamics, and disturbances. Often referred to as $\mathscr{H}_\infty$ *optimal control*.

Selection The process of selecting individuals from one generation for the next generation via a genetic operation. The individuals are selected randomly but with a bias for individuals with a higher fitness, and these individuals are advanced using one of the *genetic operations*.

Sensor A device that measures the system, producing an output. The sensor effect is typically modeled by the structure of the **C** matrix in a control system.

Stability A property of a system, referring to how it behaves for long times or when it is perturbed. For example, a fixed point of a dynamical system is *stable* if small perturbations around this fixed point result in trajectories that stay near the fixed point and do not leave a neighborhood of the fixed point. A fixed point of a linear system is unstable if some initial conditions near the fixed point result in trajectories that grow and leave the neighborhood.

State-space system A model consisting of a coupled system of ordinary differential equations in terms of a collection of variables known as the *state variable*. The state variable represents the state of the system, and it is an element of a vector space or manifold, known as the *state space*.

System identification The process of determining a model for a physical process based on measurement data. Typically, system identification involves measuring the sensor output of a system in response to certain actuation inputs, and a model for the underlying state dynamics (i.e., hidden variables) is constructed. Most methods of system identification may be viewed as a form of dynamic regression of data onto models.

Turbulence A fluid phenomena characterized by multi-scale coherent vorticity in space and time and strongly nonlinear, chaotic dynamics. Turbulence is often a characteristic of real-world or industrial flows at high *Reynolds number*.

Matlab® Code: `OpenMLC`

This appendix describes `OpenMLC`, the employed implementation of MLC in Matlab®. All examples in the book have been performed with this software.

Installation

`OpenMLC` is a Matlab® toolbox. It can be added to Matlab® by downloading the toolbox from [94] or duplicate the master branch. The root directory of the toolbox, **OpenMLC**, needs to be added to the path with subdirectories. Detailed or alternative instructions will be available in [94] as the software is updated.

Content

`OpenMLC` contains a class defined by the file `MLC.m` in the folder **@MLC**. This class implements all methods discussed in the book. Additionally, a folder **MLCtools** is provided. It provides the *MLCparameters* class description files which implements all parameters. Also functions such as expression-tree interpreter, derivation function, common function overloading for protection are provided in this folder. Finally this folder also contains the **Examples** subfolder that contains all configuration files, evaluation function, and typical results discussed in this book.

T. Duriez et al., *Machine Learning Control – Taming Nonlinear Dynamics and Turbulence*, Fluid Mechanics and Its Applications 116,
DOI 10.1007/978-3-319-40624-4

The reader is referred to the documentation of the software:

```
help MLC              % class description and first steps. Also
                      % list all properties of the MLC class and
                      % its methods.
help MLC/parameters   % will list all configuration parameters
                      % and available options
```

for a quick starting guide. Contextual help is available for each method by typing:

```
help MLC/METHOD % will provide help for METHOD
```

A full package documentation is available [94]. Any bug report, feature request or participation can be brought to our attention through the Github repository.

References

1. Nobel Media AB. Nobel peace prize (2007), http://www.nobelprize.org/nobel_prizes/peace/laureates/2007/
2. Y.S. Abu-Mostafa, M. Magndon-Ismail, H.-T. Lin, *Learning from Data. A Short Course* (AMLBook, Pasadena, 2012)
3. International Energy Agency. Key world energy statistics (2014), http://www.iea.org/publications/freepublications/publication/keyworld2014.pdf
4. S.R. Ahmed, G. Ramm, G. Faltin, Some salient features of the time averaged ground vehicle wake. Soc. Automot. Eng. SAE Inc. **1**(840300), 1–31 (1984)
5. J.-L. Aider, Private communication (2014)
6. K. Aleksic, D.M. Luchtenburg, R. King, B.R. Noack, J. Pfeiffer, Robust nonlinear control versus linear model predictive control of a bluff body wake, in *5th AIAA Flow Control Conference*, Chicago, USA, 2010. AIAA Paper 2010-4833, pp. 1–18
7. M.R. Allen, L.A. Smith, Monte Carlo SSA: detecting irregular oscillations in the presence of colored noise. J. Clim. **9**(12), 3373–3404 (1996)
8. M. Amitay, A. Glezer, Role of actuation frequency in controlled flow reattachment over a stalled airfoil. AIAA J. **40**(2), 209–216 (2002)
9. B.F. Armaly, F. Durst, J.C.F. Pereira, B. Schönung, Experimental and theoretical investigation of backward-facing step flow. J. Fluid Mech. **127**, 473–496 (1983)
10. S. Aubrun, J. McNally, F. Alvi, A. Kourta, Separation flow control on a generic ground vehicle using steady microjet arrays. Exp. Fluids **51**(5), 1177–1187 (2011)
11. S. Bagheri, L. Brandt, D.S. Henningson, Input-output analysis, model reduction and control of the flat-plate boundary layer. J. Fluid Mech. **620**, 263–298 (2009)
12. S. Bagheri, D.S. Henningson, Transition delay using control theory. Philos. Trans. R. Soc. A **369**(1940), 1365–1381 (2011)
13. S. Bagheri, J. Hoepffner, P.J. Schmid, D.S. Henningson, Input-output analysis and control design applied to a linear model of spatially developing flows. Appl. Mech. Rev. **62**(2), 020803–1..27 (2009)
14. Z. Bai, S.L. Brunton, B.W. Brunton, J.N. Kutz, E. Kaiser, A. Spohn, B.R. Noack, Data-driven methods in fluid dynamics: sparse classification from experimental data, in *Whither Turbulence and Big Data in the 21st Century*, ed. by A. Pollard. To appear in Springer (2016)
15. C.J. Baker, Ground vehicles in high cross winds. Part I: steady aerodynamic forces. J. Fluid Struct. **5**(1), 69–90 (1991)

T. Duriez et al., *Machine Learning Control – Taming Nonlinear Dynamics and Turbulence*, Fluid Mechanics and Its Applications 116,
DOI 10.1007/978-3-319-40624-4

16. B. Bamieh, L. Giarré, Identification of linear parameter varying models. Int. J. Robust Nonlinear Control **12**, 841–853 (2002)
17. W. Banzhaf, P. Nordin, R.E. Keller, R.D. Francone, *Genetic Programming: An Introduction* (Morgan Kaufmann, San Francisco, 1998)
18. R.G. Baraniuk, Compressive sensing. IEEE Signal Process. Mag. **24**(4), 118–120 (2007)
19. D. Barros, Wake and drag manipulation of a bluff body using fluidic forcing. Ph.D. thesis, École Nationale Supérieure de Mécanique et d'Aérotechnique, Poitiers, France, 2015
20. D. Barros, J. Borée, B.R. Noack, A. Spohn, T. Ruiz, Bluff body drag manipulation using pulsed jets and Coanda effect. J. Fluid Mech., in print (2016). arXiv:1507.02243 [physics.flu-dyn]
21. D. Barros, T. Ruiz, J. Borée, B.R. Noack, Control of a three-dimensional blunt body wake using low and high frequency pulsed jets. Int. J. Flow Control **6**(1), 61–76 (2014)
22. J.-F. Beaudoin, O. Cadot, J.-L. Aider, J.E. Wesfreid, Three-dimensional stationary flow over a backwards-facing step. Eur. J. Mech. B-Fluid **38**, 147–155 (2004)
23. N. Benard, J. Pons-Prats, J. Periaux, G. Bugeda, J.P. Bonnet, E. Moreau, Multi-input genetic algorithm for experimental optimization of the reattachment downstream of a backward-facing step with surface plasma actuator, in *46th AIAA Plasmadynamics and Lasers Conference*, Dallas, USA, 2015. AIAA Paper 2015-2957, pp. 1–23
24. P. Benner, S. Gugercin, K. Willcox, A survey of projection-based model reduction methods for parametric dynamical systems. SIAM Rev. **57**(4), 483–531 (2015)
25. E. Berger, M. Sastuba, D. Vogt, B. Jung, H.B. Amor, Estimation of perturbations in robotic behavior using dynamic mode decomposition. J. Adv. Robot. **29**(5), 331–343 (2015)
26. Z.P. Berger, M.G. Berry, P.R. Shea, B.R. Noack, S. Gogineni, M.N. Glauser, Active flow control for high speed jets with large window PIV. Flow Turbul. Combust. **94**, 97–123 (2014)
27. M. Bergmann, L. Cordier, J.-P. Brancher, Optimal rotary control of the cylinder wake using proper orthogonal decomposition reduced order model. Phys. Fluids **17**, 097101–1..21 (2005)
28. G. Berkooz, P. Holmes, J.L. Lumley, The proper orthogonal decomposition in the analysis of turbulent flows. Annu. Rev. Fluid Mech. **25**, 539–575 (1993)
29. T.R. Bewley, S. Liu, Optimal and robust control and estimation of linear paths to transition. J. Fluid Mech. **365**, 305–349 (1998)
30. C.M. Bishop, *Pattern Recognition and Machine Learning* (Springer, New York, 2006)
31. H.M. Blackburn, R.D. Henderson, A study of two-dimensional flow past an oscillating cylinder. J. Fluid Mech. **385**, 255–286 (1999)
32. J. Bongard, H. Lipson, Automated reverse engineering of nonlinear dynamical systems. Proc. Natl. Acad. Sci. USA **104**(24), 9943–9948 (2007)
33. L. Breiman, Random forests. Mach. Learn. **45**(1), 5–32 (2001)
34. I. Bright, G. Lin, J.N. Kutz, Compressive sensing and machine learning strategies for characterizing the flow around a cylinder with limited pressure measurements. Phys. Fluids **25**(12), 127102 (2013)
35. D. Bristow, M. Tharayil, A.G. Alleyne, A survey of iterative learning control. Control Syst. Mag. **3**(26), 96–114 (2006)
36. D.S. Broomhead, R. Jones, Time-series analysis. Proc. R. Soc. Lond. A **423**(1864), 103–121 (1989)
37. D.S. Broomhead, R. Jones, G.P. King, Topological dimension and local coordinates from time series data. J. Phys. A: Math. Gen. **20**(9), L563 (1987)
38. D.S. Broomhead, G.P. King, Extracting qualitative dynamics from experimental data. Physica D **20**(2–3), 217–236 (1986)
39. B.W. Brunton, S.L. Brunton, J.L. Proctor, J.N. Kutz, Optimal sensor placement and enhanced sparsity for classification. To appear in the SIAM J. Appl. Math. (2013). arXiv:1310.4217v1 [cs.CV]
40. B.W. Brunton, L.A. Johnson, J.G. Ojemann, J.N. Kutz, Extracting spatial-temporal coherent patterns in large-scale neural recordings using dynamic mode decomposition. J. Neurosci. Methods **258**, 1–15 (2016)

41. S.L. Brunton, B.W. Brunton, J.L. Proctor, J.N. Kutz, Koopman observable subspaces and finite linear representations of nonlinear dynamical systems for control. PLoS One **11**(2), e0150171 (2016)
42. S.L. Brunton, X. Fu, J.N. Kutz, Self-tuning fiber lasers. IEEE J. Sel. Top. Quantum Electron. **20**(5), 464–471 (2014)
43. S.L. Brunton, B.R. Noack, Closed-loop turbulence control: progress and challenges. Appl. Mech. Rev. **67**(5), 050801:01–48 (2015)
44. S.L. Brunton, J.L. Proctor, J.N. Kutz, Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proc. Natl. Acad. Sci. USA **113**(15), 3932–3937 (2016)
45. S.L. Brunton, J.L. Proctor, J.H. Tu, J.N. Kutz, Compressed sensing and dynamic mode decomposition. J. Comput. Dyn. **2**(2), 165–191 (2015)
46. S.L. Brunton, J.H. Tu, I. Bright, J.N. Kutz, Compressive sensing and low-rank libraries for classification of bifurcation regimes in nonlinear dynamical systems. SIAM J. Appl. Dyn. Syst. **13**(4), 1716–1732 (2014)
47. M. Budišić, I. Mezić, Geometry of the ergodic quotient reveals coherent structures in flows. Physica D **241**(15), 1255–1269 (2012)
48. Statistisches Bundesamt. Straßenverkehrunfälle 1986 (transl: Road traffic accidents). Technical report, Fachserie 8, Reihe 7, Statistisches Bundesamt (1987)
49. J. Burkardt, M.D. Gunzburger, H.-C. Lee, Centroidal Voronoi tessellation-based reduced-order modeling of complex systems. Technical report, Florida State University (2004)
50. E.J. Candès, Compressive sensing, in *Proceedings of the International Congress of Mathematics* (2006)
51. E.J. Candès, T. Tao, Decoding by linear programming. IEEE Trans. Inf. Theory **51**(12), 4203–4215 (2005)
52. E.J. Candès, M.B. Wakin, An introduction to compressive sampling. IEEE Signal Process. Mag. **25**, 21–30 (2008)
53. J. Carberry, J. Sheridan, D. Rockwell, Controlled oscillations of a cylinder: a new wake state. J. Fluid Struct. **17**(2), 337–343 (2003)
54. L. Cattafesta, M. Shelpak, Actuators for active flow control. Annu. Rev. Fluid Mech. **43**, 247–272 (2011)
55. L. Cattafesta, D. Williams, C. Rowley, F. Alvi, Review of active control of flow-induced cavity resonance, in *33rd AIAA Fluid Dynamics Conference and Exhibit*, 2003. AIAA Paper 2003-3567
56. L.N. Cattafesta III, D. Shukla, S. Garg, J.A. Ross, Development of an adaptive weapons-bay suppression system, in *5th AIAA/CEAS Aeroacoustics Conference and Exhibit*, 1999. AIAA Paper 1999-1901
57. L.N. Cattafesta III, Q. Song, D.R. Williams, C.W. Rowley, F.S. Alvi, Active control of flow-induced cavity oscillations. Prog. Aerosp. Sci. **44**(7), 479–502 (2008)
58. O. Cetiner, D. Rockwell, Streamwise oscillations of a cylinder in a steady current. Part 1: locked-on states of vortex formation and loading. J. Fluid Mech. **427**, 1–28 (2001)
59. A. Chadwick, K. Garry, J. Howell, Transient aerodynamic characteristics of simple vehicle shapes by the measurement of surface pressures Technical report, SAE Technical Paper, (2000)
60. S. Chaturantabut, D.C. Sorensen, Nonlinear model reduction via discrete empirical interpolation. SIAM J. Sci. Comput. **32**(5), 2737–2764 (2010)
61. B. Cheng, D.M. Titterington, Neural networks: a review from a statistical perspective. Stat. Sci. **9**, 2–30 (1994)
62. N.J. Cherry, R. Hillier, M.E.M.P. Latour, Unsteady measurements in a separated and reattaching flow. J. Fluid Mech. **144**, 13–46, 7 (1984)
63. M. Chevalier, J. Hœpffner, E. Åkervik, D.S. Henningson, Linear feedback control and estimation applied to instabilities in spatially developing boundary layers. J. Fluid Mech. **588**, 163–187 (2007)
64. H. Choi, M. Hinze, K. Kunisch, Instantaneous control of backward-facing step flows. Appl. Numer. Math. **31**, 133–158 (1999)

65. H. Choi, W.-P. Jeon, J. Kim, Control of flow over a bluff body. Annu. Rev. Fluid Mech. **40**, 113–139 (2008)
66. H. Choi, P. Moin, J. Kim, Active turbulence control for drag reduction in wall-bounded flows. J. Fluid Mech. **262**, 75–110 (1994)
67. J.M. Chomaz, Global instabilities in spatially developing flows: non-normality and nonlinearity. Annu. Rev. Fluid Mech. **37**, 357–393 (2005)
68. K.-B. Chun, H.J. Sung, Control of turbulent separated flow over a backward-facing step by local forcing. Exp. Fluids **21**(6), 417–426 (1996)
69. D. Ciresan, U. Meier, J. Schmidhuber, Multi-column deep neural networks for image classification, in *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE (2012), pp. 3642–3649
70. K.R. Cooper, Truck aerodynamics reborn - lessons from the past. Technical report, SAE Technical Paper (2003)
71. L. Cortelezzi, K.H. Lee, J. Kim, J.L. Speyer, Skin-friction drag reduction via robust reduced-order linear feedback control. Int. J. Comput. Fluid Dyn. **11**(1–2), 79–92 (1998)
72. L. Cortelezzi, J.L. Speyer, Robust reduced-order controller of laminar boundary layer transitions. Phys. Rev. E **58**(2), 1906 (1998)
73. C. Cuvier, C. Braud, J.M. Foucaut, M. Stanislas, Flow control over a ramp using active vortex generators, in *Proceedings of the 7th International Symposium on Turbulence and Shear Flow Phenomena (TSFP-7)*, Ottawa, Canada, 2011
74. C. Cuvier, J.M. Foucaut, C. Braud, M. Stanislas, Characterisation of a high Reynolds number boundary layer subject to pressure gradient and separation. J. Turbul. **15**(8), 473–515 (2014)
75. J. Dandois, E. Garnier, P. Sagaut, Numerical simulation of active separation control by a synthetic jet. J. Fluid Mech. **574**(1), 25–58 (2007)
76. L. Davis, *Handbook of Genetic Algorithms* (Van Nostrand Reinhold Company, New York, 1991)
77. H. de Garis, Genetic programming: building artificial nervous systems using genetically programmed neural networks modules, in *Proceedings of the 7th International Conference on Machine Learning*, ed. by R. Porter, B. Mooney (Morgan Kaufmann, 1990), pp. 132–139
78. J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang, Q.V. Le, Large scale distributed deep networks. Adv. Neural Inf. Process. Syst. **2**, 1223–1231 (2012)
79. A. Debien, S. Aubrun, N. Mazellier, A. Kourta, Salient and smooth edge ramps inducing turbulent boundary layer separation: flow characterization for control perspective. C. R. Mecanique **342**(6–7), 356–362 (2014)
80. A. Debien, K.A.F.F. von Krbek, N. Mazellier, T. Duriez, L. Cordier, B.R. Noack, M.W. Abel, A. Kourta, Closed-loop separation control over a sharp-edge ramp using genetic programming. Exp. Fluids **57**(3), article 40 (2016)
81. S.C.R. Dennis, P. Nguyen, S. Kocabiyik, The flow induced by a rotationally oscillating and translating circular cylinder. J. Fluid Mech. **407**, 123–144 (2000)
82. N. Desbrosses, World energy expenditures (2011), http://www.leonardo-energy.org/blog/world-energy-expenditures
83. T.G. Dietterich, Ensemble methods in machine learning, in *Multiple Classifier Systems: Second International Workshop, MCS 2001 Cambridge, UK, 2–4 July 2001 Proceedings*, ed. by J. Kittler, F. Roli (Springer, Berlin, 2001), pp. 1–15
84. D.L. Donoho, Compressed sensing. IEEE Trans. Inf. Theory **52**(4), 1289–1306 (2006)
85. A.P. Dowling, A.S. Morgans, Feedback control of combustion oscillations. Annu. Rev. Fluid Mech. **37**, 151–182 (2005)
86. J.C. Doyle, Guaranteed margins for LQG regulators. IEEE Trans. Autom. Control **23**(4), 756–757 (1978)
87. J.C. Doyle, B.A. Francis, A.R. Tannenbaum, Feedback Control Theory. (Courier Corporation, 2013)
88. J.C. Doyle, K. Glover, P.P. Khargonekar, B.A. Francis, State-space solutions to standard H_2 and H_∞ control problems. IEEE Trans. Autom. Control **34**(8), 831–847 (1989)

89. J.C. Doyle, G. Stein, Multivariable feedback design: concepts for a classical/modern synthesis. IEEE Trans. Autom. Control **26**(1), 4–16 (1981)
90. D.C. Dracopoulos, *Evolutionary Learning Algorithms for Neural Adaptive Control, Perspectives in Neural Computing* (Springer, London, 1997)
91. D.C. Dracopoulos, S. Kent, Genetic programming for prediction and control. Neural Comput. Appl. **6**, 214–228 (1997)
92. R.O. Duda, P.E. Hart, D.G. Stork, *Pattern Classification* (Wiley-Interscience, New York, 2000)
93. G.E. Dullerud, F. Paganini, in A Course in Robust Control Theory: *A Convex Approach*. Texts in Applied Mathematics (Springer, Berlin, 2000)
94. T. Duriez, OpenMLC, Matlab implementation of MLC (2016), http://OpenMLC.com
95. M. Efe, M. Debiasi, P. Yan, H. Özbay, M. Samimy, Control of subsonic cavity flows by neural networks-analytical models and experimental validation, in *43rd AIAA Aerospace Sciences Meeting and Exhibit*, 2005. AIAA Paper 2005-0294
96. M. El-Alti, V. Chernoray, P. Kjellgren, L. Hjelm, L. Davidson, Computations and full-scale tests of active flow control applied on a VOLVO truck-trailer, in *Aerodynamics of Heavy Vehicles III: Trucks, Buses and Trains*, vol. 79, Lecture Notes in Applied and Computational Mechanics, ed. by A. Dillmann, A. Orellano (Springer, Berlin, 2016)
97. R.J. Englar, Advanced aerodynamic devices to improve the performance, economics, handling and safety of heavy vehicles. Technical report, SAE Technical Paper 2001-01-2072 (2001)
98. R.J. Englar, Pneumatic heavy vehicle aerodynamic drag reduction, safety enhancement, and performance improvement, in *The Aerodynamics of Heavy Vehicles: Trucks, Buses, and Trains*, ed. by R. McCallen, F. Browand (Springer, Berlin, 2004), pp. 277–302
99. N.B. Erichson, S.L. Brunton, J.N. Kutz, Compressed dynamic mode decomposition for real-time object detection (2015). Preprint arXiv:1512.04205
100. R. Everson, L. Sirovich, Karhunen-Loeve procedure for gappy data. J. Opt. Soc. Am. A **12**(8), 1657–1664 (1995)
101. N. Fabbiane, O. Semeraro, S. Bagheri, D.S. Henningson, Adaptive and model-based control theory applied to convectively unstable flows. Appl. Mech. Rev. **66**(6), 060801–1..20 (2014)
102. B.F. Farrell, P.J. Ioannou, State estimation using a reduced-order Kalman filter. J. Atmos. Sci. **58**(23), 3666–3680 (2001)
103. P.J. Fleming, R.C. Purshouse, Evolutionary algorithms in control systems engineering: a survey. Control Eng. Pract. **10**(11), 1223–1241 (2002)
104. J.B. Freund, M.G. Mungal, Drag and wake modification of axisymmetric bluff bodies using Coanda blowing. J. Aircr. **31**(3), 572–578 (1994)
105. Y. Freund, R.E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. **55**(1), 119–139 (1997)
106. U. Frisch, *Turbulence*, 1st edn. (Cambridge University Press, Cambridge, 1995)
107. X. Fu, S.L. Brunton, J.N. Kutz, Classification of birefringence in mode-locked fiber lasers using machine learning and sparse representation. Opt. Express **22**(7), 8585–8597 (2014)
108. M. Gad-el Hak, Flow control. Appl. Mech. Rev. **42**(10), 261–293 (1989)
109. B.G. Galerkin, Rods and plates: series occuring in various questions regarding the elastic equilibrium of rods and plates (translated). Vestn. Inzhen. **19**, 897–908 (1915)
110. S. Gaucel, M. Keijzer, E. Lutton, A. Tonda, Learning dynamical systems using standard symbolic regression, in *Genetic Programming: 17th European Conference, EuroGP 2014, Granada, Spain, 23–25 April 2014, Revised Selected Papers*, ed. by M. Nicolau, K. Krawiec, M.I. Heywood, M. Castelli, P. Garcia-Sánchez, J.J. Merelo, V.M. Rivas Santos, K. Sim (Springer, Berlin, 2014), pp. 25–36
111. N. Gautier, Flow control using optical sensors. Ph.D. thesis, Ecole doctorale: Sciences Mécaniques, Acoustique, Électronique & Robotique (UPMC), ESPCI, Laboratoire PMMH, 2014
112. N. Gautier, J.-L. Aider, Feed-forward control of a perturbed backward-facing step flow. J. Fluid Mech. **759**, 181–196 (2014)

113. N. Gautier, J.-L. Aider, T. Duriez, B.R. Noack, M. Segond, M.W. Abel, Closed-loop separation control using machine learning. J. Fluid Mech. **770**, 424–441 (2015)
114. G. Gelbert, J. Moeck, C.O. Paschereit, R. King, Feedback control of unstable thermoacoustic modes in an annular Rijke tube. Control Eng. Pract. **20**, 770–782 (2012)
115. J. Gerhard, M. Pastoor, R. King, B.R. Noack, A. Dillmann, M. Morzyński, G. Tadmor, Model-based control of vortex shedding using low-dimensional Galerkin models, in *33rd AIAA Fluids Conference and Exhibit*, Orlando, FL, USA, 2003. Paper 2003-4262
116. T.A. Ghee, G. Leishman, Drag reduction of motor vehicles by active flow control using the coanda effect. AIAA J. **20**(2), 289–299 (1992)
117. D. Giannakis, A.J. Majda, Nonlinear Laplacian spectral analysis for time series with intermittency and low-frequency variability. Proc. Natl. Acad. Sci. USA **109**(7), 2222–2227 (2012)
118. A.M. Gilhaus, V.E. Renn, Drag and driving-stability-related aerodynamic forces and their interdependence-results of measurements on 3/8-scale basic car shapes. Technical report, SAE Technical Paper 860211, (1986)
119. K. Glover, J.C. Doyle, State-space formulae for all stabilizing controllers that satisfy an h_∞-norm bound and relations to risk sensitivity. Syst. Control Lett. **11**, 167–172 (1988)
120. G. Godard, M. Stanislas, Control of a decelerating boundary layer. Part 1: optimization of passive vortex generators. Aerosp. Sci. Technol. **10**(3), 181–191 (2006)
121. G. Godard, M. Stanislas, Control of a decelerating boundary layer. Part 3: optimization of round jets vortex generators. Aerosp. Sci. Technol. **10**(6), 455–464 (2006)
122. D.E. Goldberg, *Genetic Algorithms* (Pearson Education India, 2006)
123. D. Greenblatt, I.J. Wygnanski, The control of flow separation by periodic excitation. Prog. Aerosp. Sci. **36**(7), 487–545 (2000)
124. J. Grosek, J.N. Kutz, Dynamic mode decomposition for real-time background/foreground separation in video (2014). Preprint arXiv:1404.7592
125. J. Guckenheimer, P. Holmes, *Nonlinear Oscillations, Dynamical Systems, and Bifurcation of Vector Fields* (Springer, New York, 1986)
126. E. Guilmineau, O. Chikhaoui, G. Deng, M. Visonneau, Cross wind effects on a simplified car model by a DES approach. Comput. Fluids **78**, 29–40 (2013)
127. D.P. Hart, High-speed PIV analysis using compressed image correlation. J. Fluids Eng. **120**, 463–470 (1998)
128. M.A.Z. Hasan, The flow over a backward-facing step under controlled perturbation: laminar separation. J. Fluid Mech. **238**, 73–96, 5 (1992)
129. S. Haykin, *Neural Networks: A Comprehensive Foundation* (Prentice Hall, Upper Saddle River, 2004)
130. L. Henning, R. King, Robust multivariable closed-loop control of a turbulent backward-facing step flow. J. Aircr. **44**, 201–208 (2007)
131. A.J.G. Hey, S. Tansley, K.M. Tolle, *The Fourth Paradigm: Data-Intensive Scientific Discovery*, (Microsoft Research Redmond, WA, 2009)
132. G. Hinton, L. Deng, D. Yu, G.E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath et al., Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process. Mag. **29**(6), 82–97 (2012)
133. B.L. Ho, R.E. Kalman, Effective construction of linear state-variable models from input/output data, in *Proceedings of the 3rd Annual Allerton Conference on Circuit and System Theory*, 1965, pp. 449–459
134. C.-M. Ho, P. Huerre, Perturbed free shear layers. Annu. Rev. Fluid Mech. **16**, 365–422 (1984)
135. M. Högberg, T.R. Bewley, D.S. Henningson, Linear feedback control and estimation of transition in plane channel flow. J. Fluid Mech. **481**, 149–175 (2003)
136. M. Högberg, D.S. Henningson, Linear optimal control applied to instabilities in spatially developing boundary layers. J. Fluid Mech. **470**, 151–179 (2002)
137. J.H. Holland, Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence (University of Michigan Press, Ann Arbor, MI, USA, 1975)

138. P. Holmes, J.L. Lumley, G. Berkooz, C.W. Rowley, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*, 2nd paperback edn. (Cambridge University Press, Cambridge, 2012)
139. J.P. Howell, Shape features which influence crosswind sensitivity. Technical report, The Institution of Mechanical Engineers, IMechE 1993-9, Vehicle Ride and Handling (1993)
140. W.-H. Hucho, *Aerodynamics of Road Vehicles* (Society of Automotive Engineers, Warrendale, 1998)
141. L. Hung, M. Parviz, J. Kim, Direct numerical simulation of turbulent flow over a backward-facing step. J. Fluid Mech. **330**, 349–374 (1997)
142. M. Ilak, C.W. Rowley, Modeling of transitional channel flow using balanced proper orthogonal decomposition. Phys. Fluids **20**, 034103 (2008)
143. S.J. Illingworth, A.S. Morgans, C.W. Rowley, Feedback control of flow resonances using balanced reduced-order models. J. Sound Vib. **330**(8), 1567–1581 (2010)
144. S.J. Illingworth, A.S. Morgans, C.W. Rowley, Feedback control of cavity flow oscillations using simple linear models. J. Fluid Mech. **709**, 223–248 (2012)
145. M. Islam, F. Decker, E. De Villiers, A. Jackson, J. Gines, T. Grahs, A. Gitt-Gehrke, J. Comas i Font. Application of Detached-Eddy Simulation for automotive aerodynamics development. Technical report, SAE Technical Paper 2009-01-0333 (2009)
146. G. James, D. Witten, T. Hastie, R. Tibshirani, *An Introduction to Statistical Learning* (Springer, Berlin, 2013)
147. W.B. Johnson, J. Lindenstrauss, Extensions of Lipschitz mappings into a Hilbert space. Contemp. Math. **26**(189–206), 1 (1984)
148. D.W. Jordan, P. Smith, *Nonlinear Ordinary Differential Equations* (Clarendon Press, Oxford, 1988)
149. P. Joseph, X. Amandolese, C. Edouard, J.-L. Aider, Flow control using MEMS pulsed micro-jets on the ahmed body. Exp. Fluids **54**(1), 1–12 (2013)
150. J.N. Juang, *Applied System Identification* (Prentice Hall PTR, Upper Saddle River, 1994)
151. J.N. Juang, R.S. Pappa, An eigensystem realization algorithm for modal parameter identification and model reduction. J. Guid. Control Dyn. **8**(5), 620–627 (1985)
152. J.N. Juang, M. Phan, L.G. Horta, R.W. Longman, Identification of observer/Kalman filter Markov parameters: theory and experiments. Technical Memorandum 104069, NASA (1991)
153. W.J. Jung, N. Mangiavacchi, R. Akhavan, Suppression of turbulence in wall-bounded flows by high-frequency spanwise oscillations. Phys. Fluids A **4**, 1605–1607 (1992)
154. E. Kaiser, B.R. Noack, L. Cordier, A. Spohn, M. Segond, M.W. Abel, G. Daviller, J. Östh, S. Krajnović, R.K. Niven, Cluster-based reduced-order modelling of a mixing layer. J. Fluid Mech. **754**, 365–414 (2014)
155. E. Kaiser, B.R. Noack, A. Spohn, L.N. Cattafesta, M. Morzyński, Cluster-based control of nonlinear dynamics. Theor. Comput. Fluid Dyn., under review (2016). arXiv:1602.05416
156. R.E. Kalman, A new approach to linear filtering and prediction problems. J. Fluids Eng. **82**(1), 35–45 (1960)
157. J. Karhunen, J. Joutsensalo, Representation and separation of signals using nonlinear PCA type learning. Neural Netw. **7**(1), 113–127 (1994)
158. W. Kerstens, J. Pfeiffer, D. Williams, R. King, T. Colonius, Closed-loop control of lift for longitudinal gust suppression at low Reynolds numbers. AIAA J. **49**(8), 1721–1728 (2011)
159. J. Kim, Control of turbulent boundary layers. Phys. Fluids **15**(5), 1093–1105 (2003)
160. J. Kim, Physics and control of wall turbulence for drag reduction. Philos. Trans. R. Soc. A **369**(1940), 1396–1411 (2011)
161. J. Kim, T.R. Bewley, A linear systems approach to flow control. Annu. Rev. Fluid Mech. **39**, 383–417 (2007)
162. B.O. Koopman, Hamiltonian systems and transformation in Hilbert space. Proc. Natl. Acad. Sci. USA **17**(5), 315–318 (1931)
163. B.O. Koopman, J.V. Neumann, Dynamical systems of continuous spectra. Proc. Natl. Acad. Sci. USA **18**(3), 255 (1932)

164. J.R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, vol. 1 (MIT Pess, Cambridge, 1992)
165. J.R. Koza, F.H. Bennett, M.A. Andre, M.A. Keane, F. Dunlap, Automatic synthesis of analog circuits by means of genetic programming. IEEE Trans. Evol. Comput. **1**(2), 109–128 (1997)
166. J.R. Koza, F.H. Bennett III, O. Stiffelman, Genetic programming as a Darwinian invention machine, in Genetic Programming: Second European Workshop, EuroGP'99 Göteborg, Sweden, 26–27, Proceedings. Springer **1999**, 93–108 (May 1999)
167. N. Kryloff, N. Bogoliubov, *Introduction to Non-Linear Mechanics, 3*, printing edn. (Princeton University Press, Princeton, 1952)
168. J.N. Kutz, *Data-Driven Modeling and Scientific Computation: Methods for Complex Systems and Big Data* (Oxford University Press, Oxford, 2013)
169. O.A. Ladyzhenskaya, *The Mathematical Theory of Viscous Incompressible Flow*, 1st edn. (Gordon and Breach, New York, 1963)
170. Y. Lan, I. Mezić, Linearization in the large of nonlinear systems and Koopman operator spectrum. Physica D **242**(1), 42–53 (2013)
171. C. Lee, J. Kim, D. Babcock, R. Goodman, Application of neural networks to turbulence control for drag reduction. Phys. Fluids **9**(6), 1740–1747 (1997)
172. H.W. Liepmann, G.L. Brown, D.M. Nosenchuck, Control of laminar instability waves using a new technique. J. Fluid Mech. **118**, 187–200 (1982)
173. J.C. Lin, Review of research on low-profile vortex generators to control boundary-layer separation. Prog. Aerosp. Sci. **38**(4), 389–420 (2002)
174. L. Ljung, *System Identification: Theory for the User* (Prentice Hall, Englewood Cliffs, 1999)
175. K.R. Low, Z.P. Berger, S. Kostka, B. El Hadidi, S. Gogineni, M.N. Glauser, Noise source identification and control in a Mach 0.6 turbulent jet with simultaneous time resolved PIV, pressure and acoustic measurements. Exp. Fluids **54**(4), 1–17 (2013)
176. D.M. Luchtenburg, B. Günter, B.R. Noack, R. King, G. Tadmor, A generalized mean-field model of the natural and actuated flows around a high-lift configuration. J. Fluid Mech. **623**, 283–316 (2009)
177. D.M. Luchtenburg, C.W. Rowley, Model reduction using snapshot-based realizations. Bull. Am. Phys. Soc. **56** (2011)
178. D.M. Luchtenburg, M. Schlegel, B.R. Noack, K. Aleksić, R. King, G. Tadmor, B. Günther, Turbulence control based on reduced-order models and nonlinear control design, in *Active Flow Control II*, vol. 108, Notes on Numerical Fluid Mechanics and Multidisciplinary Design, ed. by R. King, Springer, Berlin, 26–28 May 2010, pp. 341–356
179. J.L. Lumley, P.N. Blossey, Control of turbulence. Annu. Rev. Fluid Mech. **30**, 311–327 (1998)
180. D.L. Ly, H. Lipson, Learning symbolic representations of hybrid dynamical systems. J. Mach. Learn. Res. **13**, 3585–3618 (2012)
181. Z. Ma, S. Ahuja, C.W. Rowley, Reduced order models for control of fluids using the eigensystem realization algorithm. Theor. Comput. Fluid Dyn. **25**(1), 233–247 (2011)
182. D.G. Mabey, Analysis and correlation of data on pressure fluctuations in separated flow. J. Aircr. **9**(9), 642–645 (1972)
183. L. Mathelin, Private communication (2015)
184. R.C. McCallen, K. Salari, J.M. Ortega, L.J. DeChant, B. Hassan, C.J. Roy, W.D. Pointer, F. Browand, M. Hammache, T.Y. Hsu et al., DOE's effort to reduce truck aerodynamic drag - joint experiments and computations lead to smart design, in *AIAA Paper 2014-2249* (2004)
185. T. McConaghy, FFX: fast, scalable, deterministic symbolic regression technology, in *Genetic Programming Theory and Practice IX (Genetic and Evolutionary Computation)*, ed. by R. Riolo, E. Vladislavleva (Springer, Berlin, 2011), pp. 235–260
186. T.T. Medjo, R. Temam, M. Ziane, Optimal and robust control of fluid flows: some theoretical and computational aspects. Appl. Mech. Rev. **61**(1), 010801 (2008)
187. I. Mezić, Spectral properties of dynamical systems, model reduction and decompositions. Nonlinear Dyn. **41**(1–3), 309–325 (2005)
188. I. Mezić, Analysis of fluid flows via spectral properties of the Koopman operator. Annu. Rev. Fluid Mech. **45**, 357–378 (2013)

189. M. Milano, P. Koumoutsakos, Neural network modeling for near wall turbulent flow. J. Comput. Phys. **182**(1), 1–26 (2002)
190. T.M. Mitchell, *Machine Learning* (McGraw Hill, New York, 1997)
191. R. Mittal, R. Kotapati, L. Cattafesta, Numerical study of resonant interactions and flow control in a canonical separated flow, in *AIAA 43rd Aerospace Sciences Meeting and Exhibit*, Reno, NV, USA, 2005. AIAA Paper 2005-1261
192. B.C. Moore, Principal component analysis in linear systems: controllability, observability, and model reduction. IEEE Trans. Autom. Control **AC-26**(1), 17–32 (1981)
193. S.D. Müller, M. Milano, P. Koumoutsakos, Application of machine learning algorithms to flow modeling and optimization, in *Annual Research Briefs* (University of Stanford, Center for Turbulence Research, 1999), pp. 169–178
194. K.P. Murphy, *Machine Learning: A Probabilistic Perspective* (MIT Press, Cambridge, 2012)
195. A.G. Nair, K. Taira, Network-theoretic approach to sparsified discrete vortex dynamics. J. Fluid Mech. **768**, 549–571 (2015)
196. B.R. Noack, K. Afanasiev, M. Morzyński, G. Tadmor, F. Thiele, A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. J. Fluid Mech. **497**, 335–363 (2003)
197. B.R. Noack, M. Morzyński, G. Tadmor, *Reduced-Order Modelling for Flow Control*, vol. 528 (CISM Courses and Lectures (Springer, Vienna, 2011)
198. B.R. Noack, R.K. Niven, Maximum-entropy closure for a Galerkin system of an incompressible periodic wake. J. Fluid Mech. **700**, 187–213 (2012)
199. B.R. Noack, M. Schlegel, B. Ahlborn, G. Mutschke, M. Morzyński, P. Comte, G. Tadmor, A finite-time thermodynamics of unsteady fluid flows. J. Non-Equilib. Thermodyn. **33**, 103–148 (2008)
200. H. Nyquist, Certain topics in telegraph transmission theory. Trans. AIEE. **47**, 617–644 (1928)
201. K. Oberleithner, Private communication (2014)
202. Council of European Union. Regulation (ec) no 715/2007 of the European parliament and of the council of 20 June 2007 on type approval of motor vehicles with respect to emissions from light passenger and commercial vehicles (euro 5 and euro 6) and on access to vehicle repair and maintenance information (2007)
203. E. Oja, Principal components, minor components, and linear neural networks. Neural Netw. **5**(6), 927–935 (1992)
204. E. Oja, The nonlinear PCA learning rule in independent component analysis. Neurocomputing **17**(1), 25–45 (1997)
205. J. Östh, S. Krajnović, B.R. Noack, D. Barros, J. Borée, On the need for a nonlinear subscale turbulence term in pod models as exemplified for a high Reynolds number flow over an ahmed body. J. Fluid Mech. **747**, 518–544 (2014)
206. V. Parezanovic, J.C. Laurentie, T. Duriez, C. Fourment, J. Delville, J.-P. Bonnet, L. Cordier, B.R. Noack, M. Segond, M. Abel, T. Shaqarin, S. Brunton, Mixing layer manipulation experiment - from periodic forcing to machine learning closed-loop control. Flow Turbul. Combust. **91**(1), 155–173 (2015)
207. V. Parezanović, L. Cordier, T. Duriez, A. Spohn, B. R. Noack, J.-P. Bonnet, M. Segond, M. W. Abel, S. L. Brunton, Frequency selection by feedback control of a turbulent shear-flow. *J. Fluid Mech.***797**, 247–283 (2016)
208. H. Park, J.-H. Cho, J. Lee, D.-H. Lee, K.-H. Kim, Aerodynamic drag reduction of Ahmed model using synthetic jet array (Technical report, SAE Technical Paper, 2013)
209. M. Pastoor, Niederdimensionale Wirbelmodelle zur Kontrolle von Scher- und Nachlaufströmungen. Ph.D. thesis, Berlin Institute of Technology, Germany, (2008)
210. M. Pastoor, L. Henning, B.R. Noack, R. King, G. Tadmor, Feedback shear layer control for bluff body drag reduction. J. Fluid Mech. **608**, 161–196 (2008)
211. B. Peherstorfer, D. Butnaru, K. Willcox, H.-J. Bungartz, Localized discrete empirical interpolation method. SIAM J. Sci. Comput. **36**(1), A168–A192 (2014)
212. J. Pfeiffer, R. King, Multivariable closed-loop flow control of drag and yaw moment for a 3D bluff body, in *6th AIAA Flow Control Conference* (Atlanta, USA, 2012), pp. 1–14

213. J. Pfeiffer, R. King, Linear parameter varying active flow control for a 3d bluff body exposed to cross-wind gusts, in *32nd AIAA Applied Aerodynamics Conference, AIAA Aviation*. AIAA-Paper 2014–2406, pp. 1–15
214. M. Phan, L.G. Horta, J.N. Juang, R.W. Longman, Linear system identification via an asymptotically stable observer. J. Optim. Theory Appl. **79**, 59–86 (1993)
215. J.T. Pinier, J.M. Ausseur, M.N. Glauser, H. Higuchi, Proportional closed-loop feedback control of flow separation. AIAA J. **45**(1), 181–190 (2007)
216. J.L. Proctor, S.L. Brunton, B.W. Brunton, J.N. Kutz, Sparsity in complex systems. Eur. Phys. J. **223**, 2665–2684 (2014)
217. J.L. Proctor, S.L. Brunton, J.N. Kutz, Dynamic mode decomposition with control. SIAM J. Appl. Dyn. Syst. **15**(1), 142–161 (2016)
218. J.L. Proctor, P.A. Eckhoff, Discovering dynamic patterns from infectious disease data using dynamic mode decomposition. Int. Health **2**(7), 139–145 (2015)
219. B. Protas, Linear feedback stabilization of laminar vortex shedding based on a point vortex model. Phys. Fluids **16**(12), 4473–4488 (2004)
220. M. Quade, M. Abel, N. Shafi, R.K. Niven, B.R. Noack, Prediction of dynamical systems by symbolic regression. Phys. Rev. E., **94** (2016)
221. A. Quarteroni, G. Rozza, *Reduced Order Methods for Modeling and Computational Reduction*, vol. 9, MS&A - Modeling, Simulation and Appplications (Springer, Berlin, 2013)
222. J.R. Quinlan, Induction of decision trees. Mach. Learn. **1**(1), 81–106 (1986)
223. K.J. Åström, R.M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers* (Princeton University Press, Princeton, 2010)
224. I. Rechenberg, *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution* (Frommann-Holzboog, Stuttgart, 1973)
225. D. Rempfer, On boundary conditions for incompressible Navier-Stokes problems. Appl. Mech. Rev. **59**(3), 107–125 (2006)
226. M. Rouméas, P. Gilliéron, A. Kourta, Drag reduction by flow separation control on a car after body. Int. J. Numer. Methods Fluids **60**(11), 1222–1240 (2009)
227. K. Roussopoulos, Feedback control of vortex shedding at low Reynolds numbers. J. Fluid Mech. **248**, 267–296 (1993)
228. C.W. Rowley, Modeling, simulation, and control of cavity flow oscillations. Ph.D. thesis, California Institute of Technology (2002)
229. C.W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, D.S. Henningson, Spectral analysis of nonlinear flows. J. Fluid Mech. **645**, 115–127 (2009)
230. C.W. Rowley, D.R. Williams, T. Colonius, R.M. Murray, D.G. MacMynowski, Linear models for control of cavity flow oscillations. J. Fluid Mech. **547**, 317–330 (2006)
231. C.W. Rowley, Model reduction for fluids using balanced proper orthogonal decomposition. Int. J. Bifurc. Chaos **15**(3), 997–1013 (2005)
232. C.W. Rowley, D.R. Williams, Dynamics and control of high-Reynolds number flows over open cavities. Annu. Rev. Fluid Mech. **38**, 251–276 (2006)
233. W.J. Rugh, J.S. Shamma, Research on gain scheduling. Automatica **36**(10), 1401–1425 (2000)
234. M. Samimy, M. Debiasi, E. Caraballo, J. Malone, J. Little, H. Özbay, M.Ö. Efe, X. Yan, X. Yuan, J. DeBonis, J.H. Myatt, R.C. Camphouse, Strategies for closed-loop cavity flow control, in *42nd Aerospace Sciences Meeting and Exhibit*, Reno, NV, USA, 2004. AIAA Paper 2004-0576
235. M. Samimy, M. Debiasi, E. Caraballo, A. Serrani, X. Yuan, J. Little, Reduced-order model-based feedback control of subsonic cavity flows - an experimental approach, in *Active Flow Control*, vol. 25, Notes on Numerical Fluid Mechanics and Multidisciplinary Design (NNFM), ed. by R. King (Springer, Berlin, 2007), pp. 211–230
236. M. Samimy, J.-H. Kim, J. Kastner, I. Adamovic, Y. Utkin, Active control of high-speed and high-Reynolds-number jets using plasma actuators. J. Fluid Mech. **578**, 305–330 (2007)
237. R.E. Schapire, The boosting approach to machine learning: an overview, Nonlinear Estimation and Classification (Springer, Berlin, 2003), pp. 149–171

238. P.J. Schmid, Dynamic mode decomposition for numerical and experimental data. J. Fluid. Mech **656**, 5–28 (2010)
239. P.J. Schmid, L. Brandt, Analysis of fluid systems: stability, receptivity, sensitivity. Appl. Mech. Rev. **66**, 1–21 (2014)
240. M. Schmidt, H. Lipson, Distilling free-form natural laws from experimental data. Science **324**(5923), 81–85 (2009)
241. T.M. Schneider, B. Eckhardt, J. Vollmer, Statistical analysis of coherent structures in transitional pipe flow. Phys. Rev. E **75**, 66–313 (2007)
242. B. Schölkopf, A.J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond* (The MIT Press, Boston, 2002)
243. H.-P. Schwefel, Kybernetische Evolution als Strategie der experimentellen Forschung in der Strömunstechnik. Master's thesis, Hermann-Föttinger-Institut für Strömungstechnik, Technische Universität Berlin, Germany, Diplom thesis (1965)
244. A. Seifert, L.G. Pack, Effects of sweep on active separation control at high Reynolds numbers. J. Aircr. **40**(1), 120–126 (2003)
245. O. Semeraro, S. Bagheri, L. Brandt, D.S. Henningson, Feedback control of three-dimensional optimal disturbances using reduced-order models. J. Fluid Mech. **677**, 63–102 (2011)
246. O. Semeraro, S. Bagheri, L. Brandt, D.S. Henningson, Transition delay in a boundary layer flow using active control. J. Fluid Mech. **731**, 288–311 (2013)
247. J.S. Shamma, M. Athans, Guaranteed properties of gain scheduled control for linear parameter-varying plants. Automatica **27**(3), 559–564 (1991)
248. C.E. Shannon, A mathematical theory of communication. Bell Syst. Tech. J. **27**(3), 379–423 (1948)
249. T. Shaqarin, C. Braud, S. Coudert, M. Stanislas, Open and closed-loop experiments to identify the separated flow dynamics of a thick turbulent boundary layer. Exp. Fluids **54**(2), 1–22 (2013)
250. R.L. Simpson, Turbulent boundary-layer separation. Annu. Rev. Fluid Mech. **21**, 205–232 (1989)
251. D. Sipp, O. Marquet, P. Meliga, A. Barbagallo, Dynamics and control of global instabilities in open-flows: a linearized approach. Appl. Rev. Mech. **63**, 251–276 (2010)
252. S. Skogestad, I. Postlethwaite, *Multivariable Feedback Control: Analysis and Design*, 2nd edn. (Wiley, Hoboken, 2005)
253. A.J. Smola, B. Schölkopf, A tutorial on support vector regression. Stat. Comput. **14**(3), 199–222 (2004)
254. P.G. Spazzini, G. Iuso, M. Onorato, N. Zurlo, G.M. Di Cicca, Unsteady behavior of back-facing step flow. Exp. Fluids **30**(5), 551–561 (2001)
255. R.F. Stengel, Optimal Control and Estimation, (Courier Corporation, 2012). arXiv preprint arXiv:1605.01671
256. B. Strom, S.L. Brunton, B. Polagye, Intracycle angular velocity control of cross-flow turbines. Preprint arXiv:1605.01671 [physics.flu-dyn]
257. J.T. Stuart, Nonlinear stability theory. Annu. Rev. Fluid Mech. **3**, 347–370 (1971)
258. R.S. Sutton, A.G. Barto, *Reinforcement Learning: An Introduction* (MIT Press, Cambridge, 1998)
259. J.A.K. Suykens, J. Vandewalle, Least squares support vector machine classifiers. Neural Process. Lett. **9**(3), 293–300 (1999)
260. F. Takens, Detecting strange attractors in turbulence. Lect. Notes Math. **898**, 366–381 (1981)
261. J.A. Taylor, M.N. Glauser, Towards practical flow sensing and control via POD and LSE based low-dimensional methods. ASME J. Fluids Eng. **126**, 337–345 (2004)
262. H. Tennekes, J. Lumley, *A First Course in Turbulence* (MIT Press, Cambridge, 1972)
263. V. Theofilis, Global linear instability. Annu. Rev. Fluid Mech. **43**, 319–352 (2011)
264. B. Thiria, S. Goujon-Durand, J.E. Wesfreid, The wake of a cylinder performing rotary oscillations. J. Fluid Mech. **560**, 123–147 (2006)
265. V. Thirunavukkarasu, H.A. Carlson, R.D. Wallace, P.R. Shea, M.N. Glauser, Model-based feedback flow control development and simulation for a pitching turret. AIAA J. **50**(9), 1834–1842 (2012)

266. R. Tibshirani, Regression shrinkage and selection via the Lasso. J. R. Stat. Soc. B **58**(1), 267–288 (1996)
267. J.A. Tropp, A.C. Gilbert, Signal recovery from random measurements via orthogonal matching pursuit. IEEE Trans. Inf. Theory **53**(12), 4655–4666 (2007)
268. M. Tsubokura, T. Kobayashi, T. Nakashima, T. Nouzawa, T. Nakamura, H. Zhang, K. Onishi, N. Oshima, Computational visualization of unsteady flow around vehicles using high performance computing. Comput. Fluids **38**(5), 981–990 (2009)
269. J.H. Tu, C.W. Rowley, An improved algorithm for balanced POD through an analytic treatment of impulse response tails. J. Comput. Phys. **231**(16), 5317–5333 (2012)
270. J.H. Tu, C.W. Rowley, D.M. Luchtenburg, S.L. Brunton, J.N. Kutz, On dynamic mode decomposition: theory and applications. J. Comput. Dyn. **1**(2), 391–421 (2014)
271. A. Visioli, Tuning of PID controllers with fuzzy logic. IEEE Proc. Control Theory Appl. **148**, 1–8 (2001)
272. B. Vukasonivic, Z. Rusak, A. Glezer, Dissipative small-scale actuation of a turbulent shear layer. J. Fluid Mech. **656**, 51–81 (2010)
273. M. Wahde, *Biologically Inspired Optimization Methods: An Introduction* (WIT Press, Southampton, 2008)
274. R.D. Wallace, P.R. Shea, M.N. Glauser, V. Thirunavukkarasu, H.A. Carlson, Simulation-guided, model-based feedback flow control for a pitching turret. AIAA J. **50**(8), 1685–1696 (2012)
275. W.X. Wang, R. Yang, Y.C. Lai, V. Kovanis, C. Grebogi, Predicting catastrophes in nonlinear dynamical systems by compressive sensing. Phys. Rev. Lett. **106**, 154101-1–154101-4 (2011)
276. N. Wiener, *Cybernetics or Control and Communication in the Animal and the Machine*, 1st edn. (MIT Press, Boston, 1948)
277. K. Willcox, J. Peraire, Balanced model reduction via the proper orthogonal decomposition. AIAA J. **40**(11), 2323–2330 (2002)
278. C.E. Willert, M. Gharib, Digital particle image velocimetry. Exp. Fluids **10**(4), 181–193 (1991)
279. J. Wright, A. Yang, A. Ganesh, S. Sastry, Y. Ma, Robust face recognition via sparse representation. IEEE Trans. Pattern Anal. Mach. Intell. (PAMI) **31**(2), 210–227 (2009)
280. X. Wu, V. Kumar, J.R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G.J. McLachlan, A. Ng, B. Liu, S.Y. Philip et al., Top 10 algorithms in data mining. Knowl. Inf. Syst. **14**(1), 1–37 (2008)
281. K.B.M.Q. Zaman, A.K.M.F. Hussain, Turbulence suppression in free shear flows by controlled excitation. J. Fluid Mech. **103**, 133–159, 2 (1981)
282. Mezić, Igor and Banaszuk, Andrzej, Comparison of systems with complex behavior. Physica D: Nonlinear Phenomena. **197**(1), 101–133 (2004)

Index

T. Duriez et al., *Machine Learning Control – Taming Nonlinear Dynamics and Turbulence*, Fluid Mechanics and Its Applications 116,
DOI 10.1007/978-3-319-40624-4