Model Predictive Controller Tuning by Machine Learning and Ordinal Optimisation

Robert Chin
ORCID: 0000-0002-5679-4677

Submitted to THE UNIVERSITY OF MELBOURNE & UNIVERSITY OF BIRMINGHAM in fulfilment of the requirements for the jointly-awarded degree of

Doctor of Philosophy

Department of Electrical and Electronic Engineering
THE UNIVERSITY OF MELBOURNE
&
School of Computer Science
UNIVERSITY OF BIRMINGHAM

October 2021

Abstract

While for the past several decades model predictive control (MPC) has been an established control strategy in chemical process industries, more recently there has been increased collaboration in MPC research between academia and automotive companies. Despite the promising work thus far, one particular challenge facing the widespread adoption of MPC in the automotive industry is the increased calibration requirement. The focus of the research in this thesis is to develop methods towards reducing the calibration effort in designing and implementing MPC in practice. The research is tailored by application to offline tuning of quadratic-cost MPC for an automotive diesel air-path, to address the limited time-availability to perform online tuning experiments.

Human preferences can be influential in automotive engine controller tuning. Some earlier work has proposed a machine learning controller tuning framework (MLCTF), which learns preferences from numeric data labelled by human experts, and as such, these learned preferences can be replicated in automated offline tuning. Work done in this thesis extends this capability by allowing for preferences to be learned from pairwise comparison data, with monotonicity constraints in the features. Two methods are proposed to address this: 1) an algorithm based around Gaussian process regression; and 2) a Bayesian estimation procedure using a Dirichlet prior. These methods are successfully demonstrated in learning monotonicity-constrained utility functions in time-domain features from data consisting of pairwise rankings for diesel air-path trajectories.

The MLCTF also constitutes a plant model, yet there will typically be some uncertainty in an engine model, especially if it has been identified from data collected with

a limited amount of experimentation time. To address this, an active learning framework is proposed for selection of the next operating points in the design of experiments, for identifying linear parameter-varying systems. The approach is based on exploiting the probabilistic features of Gaussian process regression to quantify the overall model uncertainty across locally identified models, resulting in a flexible methodology which accommodates for various techniques to be applied for estimation of local linear models and their corresponding uncertainty. The framework is applied to the identification of a diesel engine air-path model, and it is demonstrated that measures of model uncertainty can be quantified and subsequently reduced.

To make the most of the limited availability for online tuning experiments, an ordinal optimisation (OO) approach is proposed, which seeks to ensure that offline tuned controllers can perform acceptably well, once tested online with the physical system. Via the use of copula models, an OO problem is formulated to be compatible with the tuning of controllers over an uncountable search space, such as quadratic-cost MPC. In particular, results are obtained which formally characterise the copula dependence conditions required for the OO success probability to be non-decreasing in the number of offline controllers sampled during OO.

A gain-scheduled MPC architecture was designed for the diesel air-path, and implemented on an engine control unit (ECU). The aforementioned non-decreasing properties of the OO success probability are then specialised to tuning gain-scheduled controller architectures. Informed by these developments, the MPC architecture was firstly tuned offline via OO, and then tested online with an experimental diesel engine test rig, over various engine drive-cycles. In the experimental results, it was found that some offline tuned controllers outperformed a manually tuned baseline MPC, the latter which has comparable performance to proprietary production controllers. Upon additional manual tuning online, the performance of the offline tuned controllers could also be further refined, which illustrates how offline tuning via OO may complement online tuning approaches.

Lastly, using an analytic lower bound developed for OO under a Gaussian copula model, a sequential learning algorithm is developed to address a probabilistically robust offline controller tuning problem. The algorithm is formally proven to yield a controller which meets a specified probabilistic performance specification, assuming that the underlying copula is not too unfavourably far from a Gaussian copula. It is demonstrated in a simulation study that the algorithm is able to successfully tune a single controller to meet a desired performance threshold, even in the presence of probabilistic uncertainty in the diesel engine model. This is applied to two case studies: 1) 'hot-starting' an online tuning procedure; and 2) tuning for uncertainty inherent across a fleet of vehicles.

Declaration

This is to certify that

1. the thesis comprises only my original work towards the PhD,
2. due acknowledgement has been made in the text to all other material used,
3. the thesis is less than 50,000 words in length, exclusive of tables, figures, bibliographies and appendices.

Robert Chin, October 2021

Acknowledgements

The foremost thanks goes towards my supervisors: Prof. Chris Manzie, Prof. Dragan Nešić and Prof. Iman Shames from Melbourne, and Prof. Jon Rowe from Birmingham. Their invaluable guidance and suggestions have shaped the contributions in this thesis for the better. It has also been useful to gain feedback from the perspectives of two departments (Computer Science and Electrical Engineering) across two universities in different countries.

The PhD research project has been part of an industry collaboration with Toyota Motor Corporation Japan, and I am incredibly fortunate to have worked with a number of accomplished people during the project. At the beginning of the PhD, postdoctoral researchers Dr. Alex Ira and Dr. Rohan Shekhar have been instrumental in laying the early foundations and ideas this thesis has built upon. Extended thanks goes to Gokul Sankar, who showed me the ropes on all aspects of the project, including working with with the diesel engine during my first visit to Japan. During the latter half of the PhD, I am privileged to have also collaborated with Dr. Saeed Ahmadizadeh, Dr. Nalika Ulapane and Dr. Alejandro Maass. The project would also not have been possible without the support of Dr. Hayato Nakada from Toyota, whom along with Mr. Takeshi Sano, generously hosted us during our trips to the Higashi-Fuji Technical Centre, and shared aspects of Japanese culture with us. A special mention also goes to the numerous engineering staff at Toyota who helped prepare and maintain the engine test bench during our experiments.

I am grateful for the assistance provided by the Elizabeth & Vernon Puzey scholar-

ship, and also the Priestly scholarship; the former as for a top-up scholarship, and the latter which enabled me to partake in a joint PhD program with Melbourne and Birmingham. Overall, it has been a fantastic opportunity being able to experience the best of what both universities have to offer under the Priestly program.

I wish to give a shout out to office mates from both Birmingham and Melbourne, for the pleasant discussions and interactions. Their PhD endeavours have inspired my own. In no particular order: Amr Elsharkawy, Chengbin Hou, Noam Olshina, Yaqi Zhu, Meng Yuan, Chenyang Liu, Hai Nguyen, Kaixiang Wang, Chwen-Kai Liao, Abdullah Alharbi, Will Clarke and Fuad Mire Hassan. In addition, Dr. Henry Reeve helped introduce me to the techniques used in statistical learning theory.

Lastly, a very appreciative thanks goes towards friends and family, for their unwavering support over my years as a PhD candidate.

Contents

Nomenclature

List of Abbreviations

CDF Cumulative Distribution Function

CMA-ES Covariance Matrix Adaptation Evolution Strategies

DFO Derivative Free Optimisation

ECU Engine Control Unit

EGR Exhaust Gas Recirculation

EUDC Extra Urban Drive-Cycle

GPR Gaussian Process Regression

i.i.d. (Mutually) Independent and Identically Distributed

IAE Integral of Absolute Error

LCB Lower Confidence Bound

LPV Linear Parameter-Varying

LQR Linear Quadratic Regulator

MAP Maximum a Posteriori

MCMC Markov Chain Monte-Carlo

MIMO Multiple-Input Multiple-Output

MLCTF Machine Learning Controller Tuning Framework

MLE Maximum Likelihood Estimation

MPC Model Predictive Control

MVEM Mean-Value Engine Model

OCBA Optimal Computing Budget Allocation

OO Ordinal Optimisation

PDF Probability Density Function

PID Proportional Integral Derivative

QP Quadratic Program

RA Randomised Algorithms

SI Stochastically Increasing

SISO Single-Input Single-Output

TMC Toyota Motor Corporation

UDC Urban Drive-Cycle

VARX Vector Autoregression with Exogenous Inputs

VGT Variable Geometry Turbine

WLTC Worldwide Harmonized Light Vehicles Test Cycle

List of Operators/Relations/Functions

$\lfloor \cdot \rfloor$ Integer floor

$\log(\cdot)$ Natural logarithm

$\mathbb{E}[\cdot]$ Expectation

$\mathbb{I}_{\{\cdot\}}$ Indicator variable

$\mathcal{GP}(\cdot, \cdot)$ Gaussian process

$\mathcal{N}(\cdot, \cdot)$ Gaussian distribution

$\mathfrak{m}(\cdot)$ GPR mean function

$\mathsf{A}^\top$ Matrix transpose of A

$\mathsf{J}(\cdot)$ Meta-cost function

$\mathtt{f}(\cdot)$ Learned function

$\mathtt{h}(\cdot)$ Ordinal utility function

$\mathtt{k}(\cdot,\cdot)$ GPR covariance function

$\odot$ Hadamard product between matrices

$\mathrm{Beta}(\cdot,\cdot)$ Beta distribution

$\cot(\cdot)$ Cotangent function

$\mathrm{diag}\{\cdot\}$ Diagonal matrix

$\mathrm{Dirichlet}(\cdot)$ Dirichlet distribution

$\mathrm{Exp}(\cdot)$ Exponential distribution

$\Pr(\cdot)$ Probability

$\mathrm{Uniform}(\cdot,\cdot)$ Uniform distribution

$\mathrm{Var}(\cdot)$ Variance

$\overline{J}(\cdot)$ Controller performance function

$\Phi(\cdot)$ Standard Gaussian CDF

$\phi(\cdot)$ Standard Gaussian PDF

$\Phi^{-1}(\cdot)$ Standard Gaussian quantile function

$\preceq$ Loewner ordering between positive semi-definite matrices

$\times$ Cartesian product (between sets)

$\underset{\text{pref}}{\preceq}$ Preference ordering

$\underset{\text{st}}{=}$ Equality in law

$\underset{\text{st}}{\preceq}$ Stochastic ordering between random vectors

$J(\cdot,\cdot)$ System performance function

$J_{\psi^*}(\cdot)$ Test performance function with respect to plant ψ^*

$Q(\cdot)$ Standard Gaussian complementary CDF

List of Symbols

α	OO goal softening parameter
β_1	Confidence setting for LCB on α
β_2	Confidence setting for LCB on ρ
$\boldsymbol{\theta}$	Gain-scheduled controller tuning variable
δ	Setting for high OO success probability
γ	High probability setting for sequential learning algorithm
κ	Population Kendall correlation
$\mathbb{N}$	Set of natural numbers
$\mathbb{R}$	Set of real numbers
$\mathbb{R}^{\mathsf{d}}$	d-dimensional Euclidean space
$\mathbb{R}^{\mathsf{m}\times\mathsf{n}}$	Set of $\mathsf{m} \times \mathsf{n}$ real-valued matrices
$\mathbb{R}_{>0}$	Set of positive reals
$\mathbb{U}$	Input constraint set
$\mathbb{X}$	State constraint set
$\mathbb{X}_{\mathsf{N}}$	Terminal state constraint set
$\mathbf{0}$	Zero matrix
$\mathbf{1}$	Vector of ones
$\mathcal{C}$	Copula
$\mathcal{D}$	Training data
$\mathcal{X}$	Feature space
$\mathfrak{P}$	Operating space
$\mathfrak{p}$	Operating point
$\mathrm{N_e}$	Engine speed

p_{em} Exhaust manifold pressure

p_{im} Intake manifold (boost) pressure

u_{EGR} EGR valve actuation signal

u_{thr} Throttle valve actuation signal

u_{VGT} VGT vane actuation signal

W_{comp} Compressor flow rate

W_{EGR} EGR flow rate

w_{fuel} Fuel rate

y_{EGR} EGR rate

A System matrix

B System matrix (input)

C Output function matrix

e MPC integrator error state

m Dimension of input u

N MPC prediction horizon

n Dimension of state x

P MPC weight matrix on terminal state

p Dimension of output y

Q MPC weight matrix on states

R MPC weight matrix on inputs

T Trajectory number of samples

u System input

V MPC open-loop objective

x System state variable

$\mathtt{x}^+$	State successor of $\mathtt{x}$
$\mathtt{y}$	System output
$\mathtt{b}$	Coefficients of linear utility function
$\mathtt{C}$	GPR observation covariance
$\mathtt{d}$	Dimension of feature $\mathtt{x}$
$\mathtt{M}$	Number of pairwise comparisons
$\mathtt{OS}$	Overshoot
$\mathtt{RT}$	Rise time
$\mathtt{ST}$	Settling time
$\mathtt{US}$	Undershoot
$\mathtt{x}$	Feature vector
ν	Upper bound on unfavourable deviation from Gaussian copula
Ψ	Space of plant parameters
ψ	Plant parameter
ρ	Gaussian copula correlation
τ	Sequential learning algorithm stopping time
Θ	Space of controllers
θ	Controller tuning variable
ξ	OO additive representation noise level
d	Number of local controllers in gain-scheduling
m	OO selection size
n	OO sample size
p_{success}	OO success probability
$p^{\mathcal{N}}_{\text{success}}$	OO success probability under Gaussian copula

X OO variable of interest

Y OO additive representation noise

Z OO observed variable

Chapter 1
Introduction

MODEL predictive control (MPC) is gaining traction in the automotive industry. While for the past several decades MPC has been an established control strategy in chemical process industries, more recently there has been increased collaboration in MPC research between academia and automotive companies [34, 96]. We highlight two factors that have driven this surge of research.

Increased computation power. MPC requires a numerical optimisation problem to be solved online, and has enjoyed decades of success in industry, owing to the relatively slow plant dynamics in chemical process control. As the amount of available computational power has increased over time, this has allowed for practical implementation of MPC on the engine control unit (ECU) in the embedded systems of vehicles, which can match the fast dynamics and sampling rates required in automotive control.

Regulatory requirements. With environmental concerns at the forefront of attention, increasingly strict regulatory requirements has led to the need for emissions constraint satisfaction, without compromising the performance of current controllers. As a framework for approximate optimal control with constraints, MPC is naturally suited for this problem.

MPC has been applied to virtually all aspects of automotive control in the literature, from diesel air-path control to cruise control. There has also been investigation into MPC for maturing automotive technologies, such as electric vehicles [79] and self-driving cars [139]. Despite the promising work thus far, there are several challenges facing the

widespread adoption of MPC in the automotive industry.

1.1 Barriers to Widespread MPC

The amount of available computational power will continue to progress over time, as will the development of efficient algorithms and solvers. Despite this, we discuss several barriers that must be overcome for MPC to be a widespread control strategy.

Cultural reasons. A staple control strategy is Proportional Integral Derivative (PID) control, which is deeply entrenched in industry. Surveys reveal typically in excess of 90% of all controllers use PID loops in particular industrial sectors [3, 57]. For tasks requiring low-level single-input single-output (SISO) actuation, PID control is often an adequate solution. However, the prevalence of PID control could also be attributed to patterns in control education, whereby not all academic programs may offer a state-space controls course [15] and students may need to attend specialised courses to learn about MPC [170]. In contrast, virtually all introductory controls courses that teach frequency-domain classical control will introduce PID control. Thus, engineers will be equipped to design PID controllers, but may not have sufficient familiarity with MPC. This trend results in an 'inertia' in industry that makes PID control difficult to displace.

Increased calibration requirement. Being a model-based control approach, MPC explicitly requires the dynamics of the system to be modelled in order to implement. In contrast, PID control has no such explicit requirement (however a controller design process may still involve a system identification step). Nevertheless, calibrating MPC can still be considered more cumbersome for the following reason. A standard PID control loop has at most 3 gains to tune. More importantly, these gains are intuitively linked to the time-domain characteristics (e.g. rise time, steady-state error, overshoot) of the system response. In comparison, the number of tuning variables for MPC with a quadratic-cost formulation scales quadratically with the order of the system. Furthermore, there is no obvious intuitive relation between each of the variables and the resulting time-domain characteristics in the response of the controlled system. Because of this, increased cali-

bration effort can be spent tuning MPC.

In this thesis, the main motivator of the work is to address the increased calibration requirement. We seek methods which reduce the calibration effort in designing and implementing MPC in practice. Our focus is restricted to application of offline tuning MPC on an automotive diesel air-path.

1.2 Challenges in Tuning MPC

We elaborate on several specific challenges of implementing and calibrating MPC, arising in the context of automotive diesel air-path control.

Limited experimentation budget. In a controller design process, we distinguish between **offline** tuning (i.e. with simulations done on computers using a model or *digital twin* [134]) and **online** tuning (i.e. through physical experimentation on the actual plant). When designing a controller in the automotive industry, availability of an *engine test bench* may be limited, due to the test bench being simultaneously shared and negotiated between other engine development teams. Thus, there is relatively more time for offline tuning, while time for online tuning is scarce. This presents the challenge of tuning controllers offline that perform well when tested online with the actual plant, within the limited budget for experimentation.

Engine dynamics. A model of the diesel air-path engine dynamics may be used for these two purposes: 1) to perform closed-loop simulation in offline tuning; and 2) to explicitly encode within MPC for performing predictions. These two models need not necessarily be the same; the model used in closed-loop simulation may be higher fidelity than the MPC-encoded model. However, modelling diesel engine air-path dynamics can be notoriously complicated, so collecting data to identify a high fidelity model will require further experimentation time. Thus, the challenge is to identify suitable models that are useful for both purposes mentioned above, with respect to the limited testing budget. A related concern is that there will inevitably be some plant-model mismatch in

the identified models, which degrades the concordance between performance observed offline versus performance observed online. So an auxiliary challenge is to make the offline controller tuning process be more robust to uncertainty in the engine dynamics.

Assessment of performance. Human preferences can be influential in automotive engine controller tuning. For instance, an engine calibrator may deem that there is 'too much' overshoot in the response of a controller during online testing, and subsequently decide to adjust the controller accordingly. These preferences also encode the degree of trade-off that the calibrator is willing to make (e.g. how much of a reduction in overshoot at the expense of longer settling time). A key advantage of offline tuning is that it can be implemented as a fully automated procedure. However, this necessitates there no longer being a human 'in-the-loop'. The challenge spawned by this is how the preferences of the human calibrator can be learned and replicated in automated offline tuning.

1.3 Outline of Thesis

This thesis contains research findings towards addressing the challenges listed in the previous section. The document is structured as follows.

- In Chapter 2, a thorough review of the relevant literature is presented, culminating in a statement of this thesis' research aims.
- In Chapter 3, we propose machine learning algorithms which learn preferences from pairwise comparison data, in order to assess the performance of controllers during automated offline tuning.
- In Chapter 4, an active learning approach is proposed to quantify the uncertainty in an identified Linear Parameter-Varying (LPV) model of the diesel air-path, and simultaneously reduce the experimentation time required to obtain the model.
- In Chapter 5, novel theoretical results are developed in the area of ordinal optimisation (OO), using copulae as a model for offline/online performances. These results feed into the tuning methodology featured in the subsequent chapters.

- In Chapter 6, the results in OO from the previous chapter are specialised to a framework of offline controller tuning, and extended to tuning of a gain-scheduled controller architecture. The implementation of gain-scheduled MPC on a diesel engine test rig is described, and the efficacy of the OO approach to offline tuning is demonstrated.

- In Chapter 7, the results in OO from Chapter 5 are applied to probabilistically robust controller tuning. In particular, we present a sequential learning algorithm for offline tuning of controllers to satisfy a performance specification, that is probabilistically robust to plant uncertainty.

- Lastly in Chapter 8, the key contributions of the thesis are summarised, and directions for future research are concluded.

A graph of the chapter dependencies is presented in Figure 1.1.

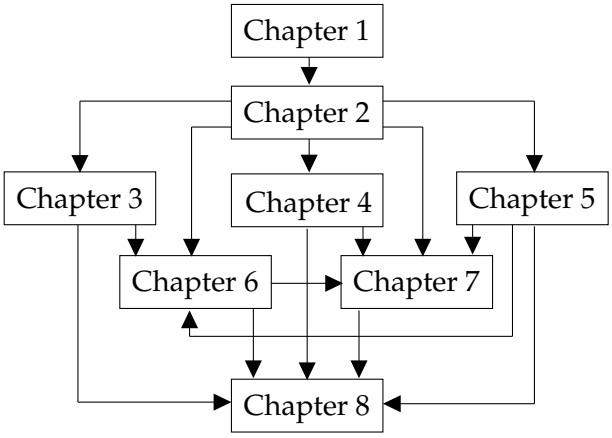


Figure 1.1: Reading dependencies on chapter in this thesis.

1.4 Notation

In this thesis, slanted serif symbols (e.g. x) are typically reserved for variables in an ordinal optimisation context, sans-serif symbols (e.g. x) are typically reserved for variables in a control context, Roman symbols (e.g. y) are typically reserved for variables in a diesel air-path context, and TrueType symbols (e.g. $\mathtt{x}$) are typically reserved for variables in a

machine learning context. The inequality $\leq$ or $<$ between vectors indicates a componentwise inequality. The dimensions of the identity matrix, zero vector and vector of ones (denoted I, $\mathbf{0}$, $\mathbf{1}$ respectively) are usually clear from context, or otherwise indicated. If required, context for a probability space is provided in the subscript to a probability $\Pr(\cdot)$ or expectation $\mathbb{E}[\cdot]$.

Chapter 2
Literature Review

In this chapter, we present preliminary material, and conduct a broad literature review in the relevant topics needed to setup the research questions addressed in this thesis. The research aims are stated at the end of the chapter, in Section 2.6.

2.1 Model Predictive Control

Model predictive control (MPC) is an intuitive control strategy about finding optimal control actions based on predictions generated by a model. MPC can be considered to belong to a class of methods (along with *reinforcement learning*) for approximate dynamic programming [23]. There are several variants of MPC (and their corresponding formulations) such as *Dynamic Matrix Control* or *Generalised Predictive Control* [131, §1.7], however they all share the common idea of leveraging a model's predictions to generate optimal inputs in an open-loop fashion.

2.1.1 Quadratic-Cost MPC Formulation

In this thesis, we primarily consider tuning MPC with a quadratic-cost functional (plus terminal cost) for the regulation of a plant with state and constraints. This formulation is commonly found in textbooks, e.g. in [83, §23.5], and we provide the elements of the formulation here. The MPC uses a prediction model to predict the future trajectory of the states with respect to a sequence of inputs. While this prediction model can generally be non-linear and also incorporate noise, this thesis considers an MPC formulation which uses a linear discrete-time prediction model (with inputs $u \in \mathbb{R}^m$, outputs $y \in \mathbb{R}^p$, and

states $x \in \mathbb{R}^n$) given by a difference equation of the form

$$x^+ = Ax + Bu \tag{2.1}$$

and output equation

$$y = Cx, \tag{2.2}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ and $C \in \mathbb{R}^{p\times n}$, with polytopic state constraints

$$\mathbb{X} = \{x \in \mathbb{R}^n : Mx \leq f\} \tag{2.3}$$

where M, f have number of rows equal to the number of state constraints, and polytopic input constraints

$$\mathbb{U} = \{u \in \mathbb{R}^m : Eu \leq h\}, \tag{2.4}$$

where E, h have number of rows equal to the number of input constraints. Moreover, both $\mathbb{X}$ and $\mathbb{U}$ are assumed to contain the origin. Consider the task of choosing inputs to regulate the states to the origin $x = 0$, from an initial condition of x_0. At time k, the online MPC cost function is defined as a quadratic in the states and inputs, with prediction horizon $N \in \mathbb{N}$:

$$V_k = \sum_{i=0}^{N-1} \left(x_{k|i}^\top Q x_{k|i} + u_{k|i}^\top R u_{k|i} \right) + x_{k|N}^\top P x_{k|N}, \tag{2.5}$$

where $x_{k|i}$ denotes the predicted future state at time $k+i$ from current state x_k, and $u_{k|i}$ denotes the applied input at time $k+i$. Also, $Q \in \mathbb{R}^{n\times n}$, $P \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{m\times m}$ are symmetric positive definite weighting matrices used to penalise the deviation of the trajectory of states and sequence of inputs from the origin. These matrices Q, P, R affect the closed-loop performance of the controller and are thus the tuning variables under consideration. Although other variables such as the prediction horizon N and sample time also affects performance, this thesis treats these as fixed, and not tuning variables (because they are tied to hardware limitations in practice). The MPC operates in the following way. At each sampling instant k, the controller finds the optimal open-loop

sequence $\{u_k^*\} := \left\{u_{k|0}^*, \ldots, u_{k|N-1}^*\right\}$ by solving:

$$\begin{aligned} &\min_{u_{k|0},\ldots,u_{k|N-1}} V_k \\ &\text{subject to } x_{k|i+1} = Ax_{k|i} + Bu_{k|i},\ i = 0, \ldots, N-1 \\ &\qquad x_{k|i} \in \mathbb{X},\ i = 1, \ldots, N-1 \\ &\qquad x_{k|N} \in \mathbb{X}_N \\ &\qquad u_{k|i} \in \mathbb{U},\ i = 0, \ldots, N-1 \end{aligned} \tag{2.6}$$

where

$$\mathbb{X}_N = \{x \in \mathbb{R}^n : M_N x \leq f_N\} \tag{2.7}$$

is a terminal constraint set, which plays a role in the analysis of stability (discussed later in Section 2.1.1.1). After (7.87) is solved, the input $u_{k|0}^*$ is applied at time k. Hence the closed-loop sequence of controls for times $k = 0, 1, 2, \ldots$ will be given by

$$\mathbf{U} := \left\{u_{0|0}^*, u_{1|0}^*, u_{2|0}^*, \ldots\right\}. \tag{2.8}$$

The closed-loop trajectory of states/outputs is the result of feeding this sequence of inputs $\mathbf{U}$ into the plant with initial condition x_0. The optimisation problem (7.87) can be formulated as a quadratic program (QP) [106], hence it is a convex problem and there exist a variety of QP solvers that can aid in finding the global optimum.

The controller defined by (7.87) is known as a *regulator*, as the aim is drive the states (and consequently, the outputs) to the origin. On an intuitive level, the weights in Q penalise future predicted deviations of the state from the origin, while P has a similar role but for only the terminal state x_N. On the other hand, the weights in R penalise non-zero control inputs u. In some contexts, non-zero u may reflect the amount of control effort or energy required to control the system. The effect of the R matrix also manifests itself as a form of regularisation to keep the magnitude of the inputs low, analogous to the role of regularisation in least squares regression [72]. Despite these connections, the exact relationship between the elements of (Q, P, R) on features of interest of the closed-loop

trajectory (such as time-domain characteristics) is generally non-trivial.

2.1.1.1 Stability of MPC

Analysis can be conducted for the stability properties of the controller defined by (7.87), applicable when the MPC prediction model perfectly matches the plant dynamics. A core object in this analysis is the terminal constraint set $\mathbb{X}_\mathsf{N}$, which if chosen to be a *control invariant set* (i.e. if $\mathsf{x} \in \mathbb{X}_\mathsf{N}$ there exists a $\mathsf{u} \in \mathbb{U}$ such that $\mathsf{x}^+ \in \mathbb{X}$), then the controller is *recursively feasible* (i.e. if the problem (7.87) is initially feasible, then it will be feasible for all time thereafter) [26, Theorem 13.1]. Having recursive feasibility, in conjunction with showing that the optimal objective $\mathsf{V}_k^*(\mathsf{x}_k) := \min_{\mathsf{u}_{k|0},\dots,\mathsf{u}_{k|\mathsf{N}-1}} \mathsf{V}_k$ acts as a Lyapunov function, closed-loop stability can be proven [26, Theorem 13.2].

2.1.1.2 MPC Integrator Augmentation

Due to plant disturbances, a regulator may result in non-zero steady-state output (i.e. we have $\lim_{k\to\infty} \mathsf{y}_k \neq 0$). The quadratic-cost MPC formulation above (7.87) can be augmented with an integrator to achieve zero *offset* (i.e. zero steady-state output regulation error) [151]. The integrator formulation we consider, applicable to constant disturbances, introduces the augmented state space

$$\begin{bmatrix} \mathsf{x}_{k+1} \\ \mathsf{e}_{k+1} \end{bmatrix} = \begin{bmatrix} \mathsf{A} & 0 \\ -\mathsf{C} & I \end{bmatrix} \begin{bmatrix} \mathsf{x}_k \\ \mathsf{e}_k \end{bmatrix} + \begin{bmatrix} \mathsf{B} \\ 0 \end{bmatrix} \mathsf{u}_k. \tag{2.9}$$

Then MPC is performed the same way as (7.87), except in the augmented state. To see how this formulation can lead to zero offset, observe that a necessary condition for equilibrium is that $\mathsf{e}_{k+1} = \mathsf{e}_k$, i.e. $\mathsf{C}\mathsf{x}_k = \mathsf{y}_k = 0$.

2.1.2 Tuning of Model Predictive Control

A 'meta-cost' problem can be formulated for the tuning of the class of MPC discussed above [45]. When given the Q, P, R matrices, the QP (7.87) can be solved online using convex optimisation, and the controller can be implemented/simulated on a plant to

perform some designated task. This results in a closed-loop output trajectory sequence, which we denote its first T elements by $\mathbf{Y}_{\mathsf{Q},\mathsf{P},\mathsf{R}}$ (where the dependence on Q, P, R is explicit). The tuning problem involves a higher level objective of choosing Q, P, R to optimise some performance index in the closed-loop output response $\mathbf{Y}_{\mathsf{Q},\mathsf{P},\mathsf{R}}$, so the following meta-cost optimisation problem is introduced:

$$\begin{aligned} &\min_{\mathsf{Q},\mathsf{P},\mathsf{R}} \mathsf{J}\left(\mathbf{Y}_{\mathsf{Q},\mathsf{P},\mathsf{R}}\right) \\ &\text{subject to } \mathsf{Q} \succ \mathbf{0}, \mathsf{P} \succ \mathbf{0}, \mathsf{R} \succ \mathbf{0} \end{aligned} \tag{2.10}$$

where $\mathsf{J}(\cdot)$ is a meta-cost function of the closed-loop response. For example, a meta-cost could be constructed to penalise undesirable time-domain characteristics.

Other approaches to formulate the tuning problem are possible, and abundant in the literature. Three papers [154], [75], [8], written roughly one decade apart from one another, survey the landscape of MPC tuning at the time of their writing. This literature encompasses a variety of approaches for tuning the multitude of MPC formulations, and their associated tuning variables. Therefore, not every approach in the literature is applicable to the quadratic-cost MPC formulation that we are considering. Instead, we taxonomise different aspects of the numerous MPC tuning methods.

2.1.2.1 Thematic Approaches

The MPC tuning methods may be organised into various thematic strategies.

Heuristic Textbooks on MPC will occasionally contain 'rules-of-thumb' tuning guidelines [37, 131, 176]. These guidelines are often based on frequency domain analysis, or rules akin to the Ziegler-Nichols rules for PID controllers. Some heuristic tuning procedures based on expert rules also appeared in [62, 204].

Analytical Analytical approaches based on pole placement [18,74] and controller matching [33,69,116,188] have been investigated in the literature. However, the main limitation

of these methods is that the analysis is conducted in the case when constraints are inactive. Another limitation is scalability; the eigenvalue expressions in [74] are based on symbolic computation, which quickly become impractical to work with for larger horizon lengths.

Algorithmic Methods under this banner apply some algorithmic approach in tuning controllers. For example, an optimisation problem like (2.10) may be explicitly formulated with objective(s) being some function of the closed-loop response with respect to the tuning variables. Then, a numerical optimisation algorithm is employed to find a solution (e.g. [148, 200]). Several ways to deal with multi-objective optimisation formulations have been presented [66, 194, 206]. The use of evolutionary and metaheuristic algorithms (e.g. genetic algorithms or particle swarm optimisation) is a reoccuring tool in many of these papers, because the formulated optimisation problems are typically treated as black-box search [53, 56, 180].

Statistical In some approaches, the selection of tuning variables is informed by first performing multiple controller experiments with different tuning variables, in order to collect data. Models are then fitted to the data, from which tuning procedures may be derived. In [17], a mapping from plant parameters to optimal tuning variables is fitted, from which a tuning rule solely based on the plant model is obtained. In [161], statistical methods have been used to characterise the sensitivity of controller performance to the tuning variables. A strategy called *response surface methodology* is trialled in [108], which fits a mapping from tuning variables to controller performance, and from which optimal tuning variables can then be found.

2.1.2.2 Tuning Variables

The nature and role of the tuning variables will depend on the specific formulation of MPC being considered.

Weights Some authors focus on tuning purely the weights in the cost function [67,192]. The ambient space of the weights themselves (hence the number of tuning variables) can also vary based on the formulation. In the quadratic-cost MPC formulation, the weights are the Q, P, R symmetric positive definite matrices. In an SISO Dynamic Matrix Control formulation, there is only a single weight (called the *move suppression coefficient*) taking on an analogous role to the R matrix [174].

Structural parameters Some studies are dedicated to tuning the structural parameters of MPC. For example, [16] focuses on the tuning of the sampling time and prediction horizon. Some formulations of MPC separate the prediction and control horizons into two different tuning variables, e.g. [131, §2.2], whereby the inputs are fixed for the remainder of the horizon. This is also known as *move blocking*, and [171] rigorously studies the optimal selection of move blocking structures. A genetic algorithm is also considered by [140] for tuning of the prediction and control horizons exclusively.

Tuning the weights and structural parameters need not be mutually exclusive, as there are examples of studies where the tuning of weights and structural parameters occurs simultaneously [61, 62, 110]. However, allowing the structural parameters to be a free tuning variable may not always be desired, since these variables could be fixed a priori, due to limitations of the hardware that the controller is being implemented on.

There is also a qualitative difference between tuning the weights and structural parameters. The weights typically take values from an uncountably infinite set (e.g. the positive definite cone). On the other hand, the structural parameters are usually values from a countable set (e.g. horizon lengths must be positive integers).

2.1.2.3 Tuning Timescale

A distinction between some tuning methods is the timescale which the method operates, i.e. whether the method is intended for offline or online tuning.

Offline In an offline approach, the controller is specifically designated to be tuned through simulations of the actual plant. This requires a computer model of the plant (i.e. a digital twin). The authors in [9,52] propose separate software packages for developing MPC in simulation. A caveat to offline tuning is that the simulation performance will not always emulate performance on the actual plant. This could be attributed to modelling uncertainties, noise, disturbances, or error induced by discretisation in the simulation. Thus, it is not immediately clear whether a well-performing controller in offline tuning will also perform well on the actual plant. Despite this, assuming a reasonably representative plant, offline simulation is still recommended to be used beforehand to assess whether the designed controller meets prescribed performance specifications [119]. In the reinforcement learning literature, the *Sim-to-Real* paradigm attempts to address the mismatch by randomisation of dynamics while training control policies in simulation [152].

Online In [6,19,32,189], adaptive methods are proposed for tuning MPC, whereby the controller is tuned online on the actual plant. A drawback of this approach is that it may be more expensive or take longer to tune the controller, compared to offline tuning. This is especially true when iterations of a controller in simulation are much faster/cheaper than iterations on the actual plant.

Many tuning methods are ambiguous, in that they are suitable to be used in an online or offline setting. To illustrate, a proposed algorithmic approach could be applicable irrespective of whether it is performed in simulation or on the actual plant, provided: 1) the performance of any given controller can be evaluated by some oracle; and 2) the tuning variables can be updated in-between evaluations.

Offline and online methods can also complement one-another, i.e. an offline method can be used to design an initial controller in simulation, and then an online method fine-tunes the controller on the actual plant, as suggested by [119, §4.1].

2.1.2.4 Machine Learning Controller Tuning Framework

In the aforementioned MPC tuning literature, the performance indices used to assess the controllers (e.g. choice of $\mathsf{J}(\cdot)$ in (2.10)) are taken as pre-established. One such example of a performance index is the integral of squared error [68]. In a multi-objective case, a scalar performance index can be obtained from a weighted sum of the objectives (e.g. [4]); these weights implicitly encapsulate the trade-offs between competing objectives. However, as acknowledged in [73,84], human preferences play an import role in MPC tuning. Moreover, it may not be possible for human calibrators to directly write down a function $\mathsf{J}(\cdot)$ which captures the trade-offs that they would themselves make when tuning a controller. This is similar to the motivation behind *inverse optimal control* [147], being that for some tasks, it is difficult to specify a cost or reward function that makes the system behave as desired.

In [104], a machine learning controller tuning framework (MLCTF) was introduced for learning the preferences of expert human controller calibrators, which addresses the requirement to obtain a performance index $\mathsf{J}(\cdot)$ if one is not explicitly provided. The procedure was demonstrated for tuning MPC matrices Q, P, R in a quadratic-cost formulation. The paper used neural networks to learn a mapping from time-domain characteristics (e.g. overshoot, rise time, etc.) of closed-loop trajectories to scalar performance rating, using human-provided rating data. This mapping was then used in conjunction with a derivative-free optimiser to tune the weights in an offline setting.

Figure 2.1 shows a diagram of the framework, which in essence, optimises an objective function $J(\theta)$ with respect to the tuning variables θ, where J is the composition $J = \mathtt{f} \circ \mathtt{x} \circ \mathbf{Y}$ and the function $\mathtt{f}$ is learned from human rating data.

2.1.2.5 Parametrisations of MPC Cost Weight Matrices

When tuning quadratic-cost MPC via the meta-cost as in (2.10), one important consideration is how to parametrise the Q, P, R matrices. 'Off-the-shelf' optimisation algorithms

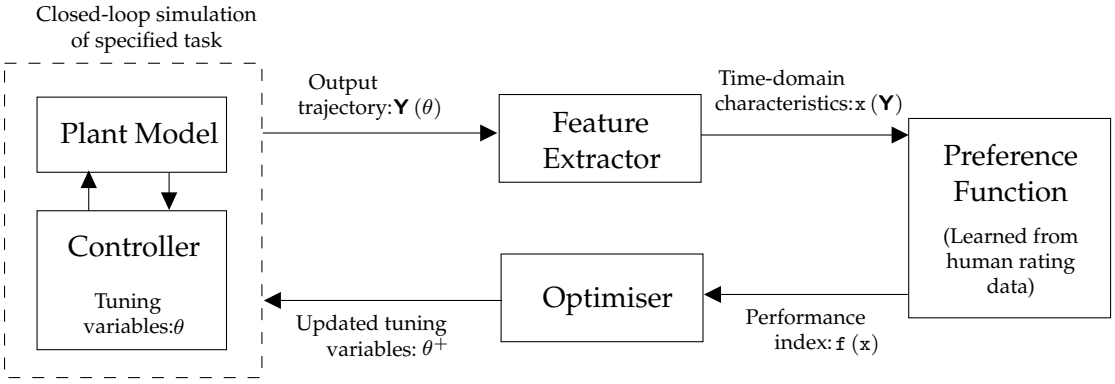


Figure 2.1: The machine learning controller tuning framework.

typically optimise variables over some subset of $\mathbb{R}^d$. In quadratic-cost MPC however, the matrices are $Q \succ \mathbf{0}$, $P \succ \mathbf{0}$, $R \succ \mathbf{0}$, so they belong to the set of positive definite matrices in the respective dimension. One option is to use diagonal weights, as in [148, 201], which reduces the parametrisation of Q, P, R to elements in $\mathbb{R}^n$, $\mathbb{R}^n$, $\mathbb{R}^m$ respectively. However, this reduces the flexibility of the controller and the opportunity to potentially find better-performing weights.

Alternatively, naively parametrising Q as a vector from $\mathbb{R}^{n^2}$ is not very productive, since Q is a symmetric matrix, and so can be uniquely determined by $\left(n^2+n\right)/2$ elements. However, even by parametrising the upper or lower triangular elements of Q as a vector in $\mathbb{R}^{\left(n^2+n\right)/2}$, one still needs to enforce the positive definiteness constraint. The following two approaches can achieve this.

Cholesky decomposition As suggested by [66], the matrix Q (and analogously for P, R) can be parametrised by the $\left(n^2+n\right)/2$ triangular elements of its Cholesky decomposition: $Q = L_Q L_Q^\top$, where L_Q is lower triangular. If the main diagonal of L_Q consists of all positive elements (i.e. this implies L_Q is invertible), then $L_Q L_Q^\top$ is positive definite. Moreover, every positive definite matrix has a unique Cholesky decomposition with all positive elements in the main diagonal [80, Theorem 4.2.7]. Hence in this parametrisation, n of the elements in the parametrisation will each be restricted to $\mathbb{R}_{>0}$, while the remaining $\left(n^2-n\right)/2$ elements will each take on values in $\mathbb{R}$.

Spectral decomposition For positive definite Q (and analogously for P, R), we can decompose $Q = W_Q D_Q W_Q^\top$ where W_Q is orthogonal, and D_Q is diagonal with positive main diagonal elements. Thus diagonal D_Q is parametrised by n positive numbers in $\mathbb{R}$, while orthogonal W_Q can be minimally parametrised by $\left(n^2 - n\right)/2$ values ([173] surveys various approaches to parametrise orthogonal matrices).

In [45], a parametrisation based on spectral decomposition is demonstrated for tuning MPC, motivated by the observation that if we multiply Q, P, R all by the same positive constant $c > 0$, the solution to the MPC problem (7.87) (hence the behaviour of the controller) remains unchanged. This suggests some redundancy, which can be exploited to further reduce the number of tuning variables. In addition, P is fixed with respect to Q, R by setting it as the solution to the discrete-time algebraic Riccati equation [83, Eq. (22.7.9)]:

$$Q + A^\top P A - A^\top P B \left(R + B^\top P B\right)^{-1} B^\top P A - P = \mathbf{0}. \tag{2.11}$$

Several reasons advocate for the P matrix to be chosen this way:

- it reduces the number of tuning variables of the controller,
- it causes the MPC law to be equivalent to the Linear Quadratic Regulator (LQR) law when constraints are not active, and
- it can be stated as one of the sufficient conditions for closed-loop stability [135, §3.7.4.1].

In all, this parametrisation of the MPC cost weights consists of $\left(n^2 + n\right)/2 + \left(m^2 + m\right)/2 - 1$ variables, as opposed to the $n^2 + n + \left(m^2 + m\right)/2$ variables required for parametrising each of Q, P, R directly. Furthermore, using hyperspherical coordinates to parametrise diagonal matrices and the *Givens rotations* parametrisation for orthogonal matrices, the tuple (Q, P, R) can be parametrised over a box-constrained search-space, which is compatible with many existing off-the-shelf solvers. By additionally introducing a parameter $r > 0$ that specifies the sum of squared diagonal elements in D_R relative to D_Q, this further allows for some level of control of the regularisation in the controller.

In the works [103,104] applying the MLCTF (Section 2.1.2.4), and also in [128], a variant of the Nesterov gradient free optimiser (originally from [146]) is tailored towards tuning weights in quadratic-cost MPC, based on the spectral decomposition of positive definite matrices. The algorithm requires random perturbations to the matrices, which are performed (in the case of Q, and analogously for the other matrices) by generating uniformly random orthogonal matrices W_Q, performing a perturbation to the diagonal elements of D_Q, followed by projecting $W_Q D_Q W_Q^\top$ onto the positive definite cone. By exploiting the symmetric structure, it is shown in [130] that this approach leads to lower complexity bounds in optimisation.

2.2 Diesel Engine Air-Path

In this thesis, the plant that we consider controlling is the air-path of a Toyota 1GD-FTV diesel engine (abbreviated to GD engine hereafter). The air-path is a subsystem within the wider engine control system governed by the ECU. The engine features exhaust gas recirculation (EGR) and variable geometry turbine (VGT) technology. Figure 2.2 contains a schematic of the diesel engine air-path. The relevant signals of interest for the modelling and control of the plant are described in this section.

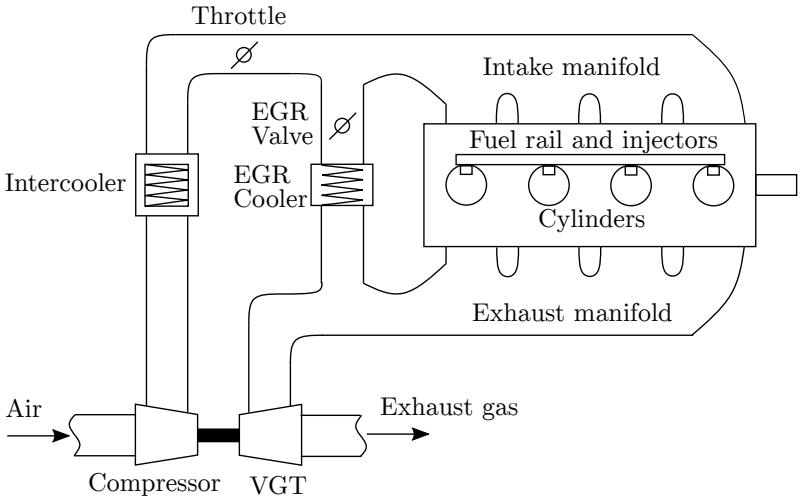


Figure 2.2: Schematic of an automotive diesel engine air-path.

Operating condition Two signals which the air-path takes as given are the **engine speed** $\mathrm{N_e}$ (in units rpm) and the **fuel rate** $\mathrm{w_{fuel}}$ (in units mm^3/stroke). Together, these two signals constitute the current operating condition of the plant, and we refer to the pair $\mathfrak{p} = (\mathrm{N_e}, \mathrm{w_{fuel}})$ as the engine *operating point*. The operating point is controlled through other subsystems of the diesel engine, yet the operating point itself will still influence the air-path dynamics.

Pressures Figure 2.2 depicts the different manifolds of the air-path (e.g. pre-throttle manifold, intake/exhaust manifolds) which may be treated as thermodynamic control volumes. Each of these manifolds will be associated with a signal for the pressure of the gas within the manifold. Of note, we have the **intake manifold pressure** (also called the 'boost' pressure) $\mathrm{p_{im}}$ (measured in kPa) and **exhaust manifold pressure** $\mathrm{p_{em}}$ (measured in kPa).

Mass flow rates Between thermodynamic control volumes, there will be flow of gas, quantified by mass flow rate signals. The flow rate of fresh air entering the system is given by the **compressor flow rate** $\mathrm{W_{comp}}$ (measured in g/s). The **EGR flow rate** is $\mathrm{W_{EGR}}$ (in g/s), which measures the flow rate of exhaust gas that is recirculated from the exhaust manifold back into the intake manifold for combustion. The **EGR rate** $\mathrm{y_{EGR}}$ is another signal defined as

$$\mathrm{y_{EGR}} := \frac{\mathrm{W_{EGR}}}{\mathrm{W_{comp}} + \mathrm{W_{EGR}}}. \tag{2.12}$$

This variable quantifies the mixture of fresh air and exhaust gas that is being supplied to the cylinders.

Actuators The air-path can be controlled via three actuators, shown in Figure 2.2. The actuation signal for the **throttle valve** is $\mathrm{u_{thr}}$ (specified in percentage closed), the signal for the **EGR valve** is $\mathrm{u_{EGR}}$ (percentage open) and the signal for the **VGT vane** is $\mathrm{u_{VGT}}$ (percentage closed).

Further details about this particular diesel air-path may be found in [166, §2.2] and

[97, §1.3].

2.2.1 Modelling of Diesel Engine Air-Path

2.2.1.1 Mean-Value Engine Models

The operation of a diesel engine can be described through repeated thermodynamic cycles, wherein combustion occurs in each cycle. While these cycles do influence the air-path dynamics, they happen on a very short timescale. On this timescale however, the effects of combustion may not be particular relevant for air-path control. Thus, the purpose of a mean-value engine model (MVEM) is as a control-oriented model. That is, an MVEM will preserve the main dynamics of interest with acceptable fidelity required to control the air-path, without the excess complexity of needing to model the combustion process. In an MVEM, the signals are considered to be averaged over one or more engine cycles [65, Definition 7.1].

2.2.1.2 Identification Approaches

We discuss two schools of thought in the identification of control-oriented models for a diesel engine air-path.

Physics-based modelling In a physics-based modelling approach, physics principles (e.g. conservation of mass and ideal gas laws) are applied towards deriving continuous-time dynamic equations for the pressures and other states. This approach is followed in [198], whereby a non-linear MVEM is identified and validated over experimental data. Some components of the model (e.g. friction torque) are diffcult to derive based on phyics principles, and are instead assumed to take on the form of polynomial equations. Non-linear least squares is used to identify the model parameters. The advantage of a physics-based model is that it can potentially be used in a high fidelity closed-loop simulation. However, a drawback is needing to solve a potentially difficult non-linear least squares problem, which poses questions for how well the model can be identified. Furthermore, implementation of MPC requires a discrete-time model, so in order to acquire

a model suitable for MPC implementation, one then needs to discretise (and perhaps also linearise) the physics-based model. This is performed in [99], where local linear models are obtained by linearising a given physics-based non-linear model.

Pure data-driven modelling An alternative to physics-based modelling is a pure data-driven approach, in which phyics principles are discarded in favour of fitting classes of time-series models to the data. The authors in [98] initially attempted physics-based models, but found the identified models to give poor fit to data, thus instead relied on identifying non-linear autoregressive exogenous (NARX) models with a polynomial structure. In [172], discrete-time local linear models are also identified directly from data. One advantage of the data-driven approach, if fitting linear models, is that the data requirements for the identification process (i.e. convergence rates of parameter estimates in the number of samples) will be relatively well understood [137, 175]. A potential disadvantage of this approach is that the simulation-fidelity may not be as high as a well-calibrated physics based model, since the models will be valid only within the region local to where data was gathered. Also, generic time-series models will fail to adhere to physical constraints (e.g. non-negativity constraints) of some signals.

Despite the abundance of work in diesel air-path identification in the literature, reports into any uncertainty quantification (i.e. obtaining uncertainty intervals for model parameters) of the diesel air-path has been relatively scarce.

2.2.1.3 Linear Parameter-Varying Sytems

Recently, the use of linear parameter-varying (LPV) systems (where the matrices A and B in (2.1) are now functions of an operating point $\mathfrak{p}$) have emerged as an approach for model-based control of non-linear systems, whereby local linear controllers are designed for regions of an operating space in a *gain-scheduled* manner [185]. Thus, LPV models are highly suitable for the diesel air-path because the latter is parametrised in the operating point $\mathfrak{p} = (\mathrm{N_e}, \mathrm{w_{fuel}})$. In particular, when the operating point $\mathfrak{p}$ is itself determined as a function of the state x (e.g. via a feedback loop in the ECU), this is specifically termed a

quasi-LPV system [30, §1.2.1].

There are two broad approaches to the identification of LPV systems. In the *local* approach, several local linear models are identified at several fixed operating points (also called scheduling points), which are then interpolated over the operating space. In the *global* approach, an LPV model is identified from an experiment which excites the operating space as well [60].

2.2.1.4 Active Learning

Active learning, along with closely-related optimal experimental design, are subfields of machine learning and statistics, that are concerned with the determination of query points which to sample data [168]. The main rationale underpinning active learning is that **data collection is costly**, so these query points should be selected in a way such that it optimises some notion of accuracy for a model being identified. Thus, active learning carries the advantage of enabling either identification of a model that is more accurate for a fixed data collection budget, or identification to a specified accuracy within a smaller data collection budget.

Optimal experimental design for dynamical systems has been studied since the 1960s. [122] demonstrated that a white noise input signal to a SISO discrete-time linear system minimised the *A-optimality* criterion (trace of the covariance matrix) for the parameters of a finite impulse response model. [82] gave an A-optimality formulation for optimal design of input signals for a general class of discrete-time non-linear systems. Due to limited computational resources at the time of the paper, the method was exemplified on simpler systems.

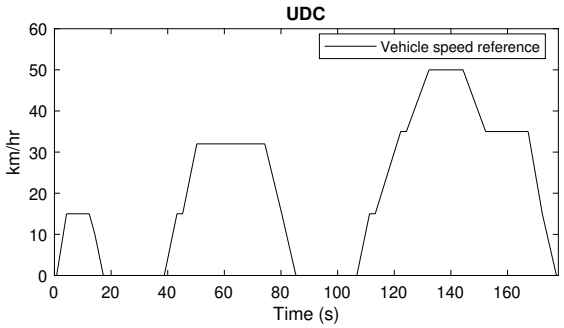
The optimal experimental design for local LPV identification has previously been investigated, where in [112], a technique was proposed to select new operating points to query for SISO systems with a univariate operating point. Their approach minimised a measure of anticipated overall accuracy, and assumed that each local linear model could

be identified perfectly. [143] relaxes this assumption, and provides an algorithm for the simultaneous selection of operating points and design of input signals (although still only valid for the class of SISO systems with univariate operating point). Their optimisation criterion is based on an A-optimality-like criterion.

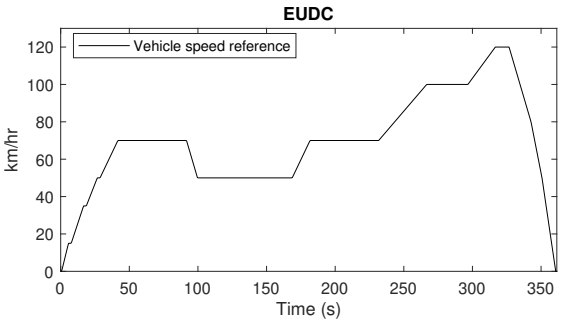
2.2.2 Control of Diesel Engine Air-Path

2.2.2.1 Drive-Cycles

A drive-cycle (or *driving cycle*) is a standardised trajectory for a vehicle speed reference over time, which used in the development, testing and benchmarking of vehicles. During transient testing of controllers performed over drive-cycles in this thesis, the vehicle speed reference is translated by the test hardware into a time-varying reference tracking problem, with reference trajectory denoted $\mathrm{y_{ref}}$. Examples of common drive-cycles include the Urban Drive-Cycle (UDC, also known as ECE-15), Extra-Urban Drive-Cycle (EUDC) [191] depicted in Figures 2.3a and 2.3b respectively, and the more recent Worldwide Harmonized Light Vehicles Test Cycle (WLTC, also abbreviated as WLTP) [47], which has four sections as shown in Figure 2.4.

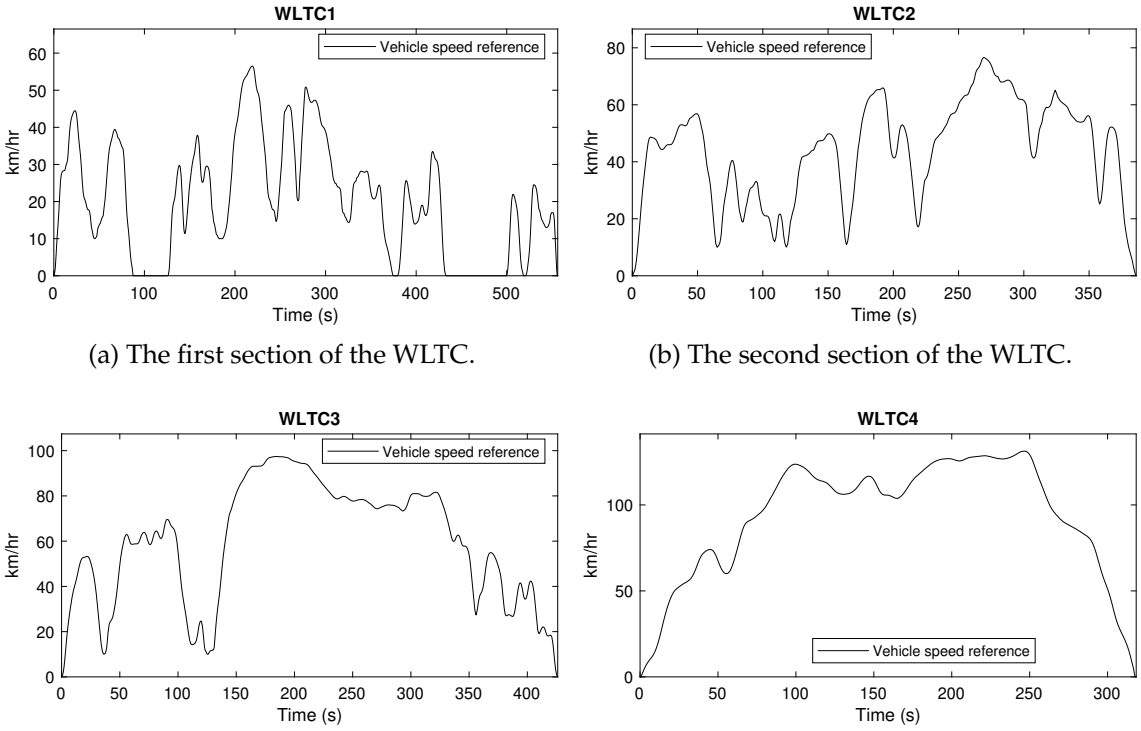


(a) Urban Drive-Cycle



(b) Extra-Urban Drive-Cycle

Figure 2.3: The Urban and Extra-Urban Drive-Cycles.



(a) The first section of the WLTC.

(b) The second section of the WLTC.

(c) The third section of the WLTC.

(d) The fourth section of the WLTC.

Figure 2.4: The Worldwide Harmonized Light Vehicles Test Cycle.

2.2.2.2 PID Control for Diesel Air-Path

PID (Proportional-Integral-Derivative) controllers have been standard in both the literature and industry for diesel air-path control [199]. An issue posed by controlling the diesel engine air-path is that the dynamics are highly non-linear, and also dependent on the operating point. This is typically addressed by an additional feed-forward/set-point control component in addition to feedback, using mappings which have been calibrated through experiments [50, §1.2.1]. An additional strategy in diesel air-path control is to apply *gain-scheduling [193]*. In gain-scheduling, different controller gains/weights are used in different regions of the operating space to account for dynamics which vary over the parameter space (e.g. an LPV system) [30, §3.1].

Software designed for general purpose calibration in the automotive industry, such as AVL CAMEO, may also be used to assist with the PID tuning workflow. The optimisa-

tion algorithms implemented by AVL CAMEO include some of those discussed in Section 2.1.2.1, such as genetic algorithms [211].

2.2.2.3 MPC for Diesel Air-Path

Several studies have also experimented with MPC for the diesel air-path. The same techniques of feed-forward control and gain-scheduling can be applied, featuring different local linear prediction models and cost weights active at different regions of the operating space [210].

Some earlier work [100, 103, 128, 129, 163–165, 172] has experimented with MPC on the same class of diesel engine that we consider in this thesis (the GD engine). In those works, the control objective is to track an output reference $\mathsf{y}_{\mathrm{ref}}$ for the boost pressure and EGR rate:

$$\mathsf{y} = \begin{bmatrix} \mathrm{p_{im}} & \mathrm{y_{EGR}} \end{bmatrix}, \tag{2.13}$$

using the inputs

$$\mathsf{u} = \begin{bmatrix} \mathrm{u_{thr}} & \mathrm{u_{EGR}} & \mathrm{u_{VGT}} \end{bmatrix}. \tag{2.14}$$

In [100], a rate-based MPC strategy (an alternative to integrator augmentation for offset-free tracking) was tested. A novel MPC formulation with exponentially decaying envelopes was proposed in [162], which aims to simplify and speed up the controller calibration process. This formulation was tested for diesel air-path control on the GD engine in [172], which found that tracking could be successfully performed with the four-state representation

$$\mathsf{x} = \begin{bmatrix} \mathrm{p_{im}} & \mathrm{p_{em}} & \mathrm{W_{comp}} & \mathrm{y_{EGR}} \end{bmatrix}. \tag{2.15}$$

The studies [163–165] incorporated an additional robust formulation into the MPC, via constraint tightening.

The MLCTF from [104], mentioned earlier in Section 2.1.2.4, has also been applied to tuning MPC for the diesel air-path. MPC was tuned offline and tested on the physical

engine over transient drive-cycles in [103]. In [128, 129], an online tuning approach was performed and tested over transient drive-cycles, using the integral of squared error as a performance index. The authors also used the same variant of the Nesterov gradient-free optimiser that originally featured in [104].

2.3 Utility Theory

As mentioned in Section 2.1.2.4, preferences can play a pivotal role in controller tuning. Our interest in utility theory is for the study of preferences, the latter which has attracted attention from a number of disciplines in the literature. Economists first introduced a rigorous treatment on the theory of preferences in the mid-20$^{\text{th}}$ century [54, 197]. Psychologists have also studied the impact of preferences on decision making [107]. The task of learning preferences from data has also gathered attention in the field of machine learning and computer science; an overview to preference learning from this corner of literature is given in [71].

2.3.1 Ordinal Utility Functions

We build a definition for utility functions, which act as a proxy for preferences. Let $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_{\mathtt{d}}$ denote a topological space, which we may call the *feature space*. The binary preference relation $\underset{\text{pref}}{\preceq}$ denotes a total ordering on $\mathcal{X}$, where for a pair of items $\mathtt{x_A}, \mathtt{x_B} \in \mathcal{X}$, the expression $\mathtt{x_A} \underset{\text{pref}}{\preceq} \mathtt{x_B}$ asserts that item $\mathtt{x_B}$ is preferred at least as much as $\mathtt{x_A}$. The strict preference relation is denoted by $\underset{\text{pref}}{\prec}$.

Definition 2.1 (Ordinal utility functions)**.** *An ordinal utility function* $\mathtt{h} : \mathcal{X} \to \mathbb{R}$ *for* $\underset{\text{pref}}{\preceq}$ *is a function representing the underlying preferences of an agent such that*

$$\mathtt{x_A} \underset{\text{pref}}{\preceq} \mathtt{x_B} \Leftrightarrow \mathtt{h}\left(\mathtt{x_A}\right) \leq \mathtt{h}\left(\mathtt{x_B}\right). \tag{2.16}$$

Note that there exists infinitely many ordinal utility functions that can represent a

single preference relation $\underset{\text{pref}}{\preceq}$, since for an ordinal utility function $\mathtt{h}(\mathtt{x})$ and any strictly monotonic (increasing) transformation $\mathtt{q}(\cdot)$, then $\mathtt{q}(\mathtt{h}(\mathtt{x}))$ is also an ordinal utility function for $\underset{\text{pref}}{\preceq}$.

An analogous definition and interpretation follows for *ordinal cost functions*, by reversing the direction of the inequality. That is, maximising the utility is equivalent to minimising the cost. Moreover, given a utility function, a cost function can be obtained by taking the negative of the utility.

2.3.2 Learning from Pairwise Rankings

Within the literature, there are various formulations of preference learning problems. A sub-class of these is the problem of learning preferences from pairwise rankings: where the data consists of pairwise comparisons (i.e. ordinal data), as opposed to numerical labels.

We first detail the data structure of this class of learning problem. An individual pairwise comparison (training example) consists of the pair $(\mathtt{x}, \mathtt{x}') \in \mathcal{X} \times \mathcal{X}$ and the label $\mathfrak{y} \in \{-1, 1\}$, where a label of 1 indicates that $\mathtt{x}$ has been rated preferred over $\mathtt{x}'$, while a label of -1 indicates that $\mathtt{x}'$ has been rated preferred over $\mathtt{x}$. Some formulations allow for a label to indicate ties, but this thesis does not. The structure of a pairwise comparison dataset $\mathcal{D}$ consists of $\mathtt{M}$ individual comparisons as training examples. Then, the goal is to learn the underlying preference relation $\underset{\text{pref}}{\preceq}$ (usually via a utility function), from the data $\mathcal{D}$ (in which the comparisons may have been corrupted with rating noise).

There are several motivating reasons for working with ranking data, in lieu of numeric ratings. In some cases, ranking data are the type of data that are naturally solicited, e.g. a user who selects a particular result from a list of search engine results conveys that their selection was ranked above all other alternatives. In other situations, ranking data may be more useful/reliable compared to numeric labels [190]. For instance, a human may be asked to rate on a scale of 1 to 10, but this scale will be subjective depending on

the human, and this subjectivity could inadvertently 'drift' over time (e.g. a mediocre object may begin with a rating of 5, but then the same could could be rated 6 by the same human in the future). A pairwise comparison rating scheme is a way to protect against these issues.

Work pertaining to learning from pairwise rankings has been shared across the psychometric, econometric and computer science literature. An early contribution proposed in psychometrics is the Thurstone-Mosteller model [142, 184], which assumes Gaussian rating noise. A variant known as the Bradley-Terry-Luce model [29] assumes Gumbel distributed rating noise. These models are the basis for several sports rating systems, such the Elo rating system used in competitive chess [64].

Discrete choice theory, originally from [136], is a Nobel prize-winning body of work from economics, which models the preferences and utility of decision makers based on choices made between a finite number of alternatives. In the case of a choice between two alternatives, this reduces to an archetypal problem of learning from pairwise rankings. The learned utility functions can be applied in the modelling and forecasting of consumer demand [22, 186], while estimation of utility functions themselves is traditionally performed via maximum likelihood or Bayesian methods [187].

Many contributions have arisen in the computer science literature, motivated by applications in recommender systems [81] and information retrieval [125]. In [138], theoretical relationships are drawn between pairwise ranking and the related problem of *bipartite ranking* (which is itself similar to binary classification). A Bayesian version of the Thurstone-Mosteller model is extended using Gaussian processes in [49]. An online ranking algorithm using Borda scores (which generalise the Thurstone-Mosteller and Bradly-Terry-Luce models) is considered in [91]. Learning from pairwise rankings has also been investigated for reinforcement learning, where [202] originally proposed a policy gradient algorithm. These reinforcement learning algorithms have been exhibited in robot locomotion [48] and Atari game playing [101].

2.3.3 Isotonic Preference Learning

In a feature space $\mathcal{X}$, suppose it is known that some features are *desirable* or *undesirable*, i.e. the more (less) of a particular feature, the better (worse), all else being equal. Hence the utility should always be non-decreasing (non-increasing) in these features. Equivalently, the cost should always be non-increasing (non-decreasing) in these variables. Placing this concept in the context of controller tuning, where `x` is a feature vector of time-domain characteristics, many variables would exhibit some desirability or undesirability. For example, many control engineers would agree that the faster the settling time, the better, all else being equal. In that case, we may wish to incorporate this knowledge into the learned utility function, via monotonicity constraints. Further justification is elaborated as follows.

Sensible Interpretation of Utility Functions To illustrate with a simple controller tuning example, consider the trade-off between two features: the overshoot `OS` and settling time `ST` of a single-output step response trajectory. One can reasonably assume that an engineer prefers less overshoot (holding settling time constant), and likewise for settling time (holding overshoot constant). Hence the utility over these features would be non-increasing. Without monotonicity constraints, there could exist a point where the learned utility function is increasing in `OS` and/or `ST`. At this point, this would then suggest that increasing `OS` and/or `ST` would improve the trajectory, which is not a sensible interpretation.

Regularisation Incorporating prior information is a commonly used technique for regularisation in learning (e.g. the Bayesian interpretation of regularisation [25, §1.2.5]). We can leverage prior knowledge of monotonicity to regularise the estimates and reduce the data requirements that lead to a desired quality of estimate, compared to methods which do not explicitly consider monotonicity. For the latter class of algorithms, monotonicity may not even be guaranteed except in the asymptotic case when there is an arbitrarily large amount of data.

If the preference data comprised of numeric labels (as in previous controller tuning work [104]), this would invoke the problem of *isotonic regression*, which could be solved, for example, through non-negative least squares [121, Chapter 3] or a non-parametric approach [117]. Monotonicity has also previously been considered for ordinal classification [20], where the output label belongs to a finite number of classes. When the data comprises pairwise comparisons however, there are seldom approaches in the literature which explicitly addresses this. The method proposed in [58] does produce monotonic utility functions, by virtue of making the utility function a product of beta cumulative distribution functions (CDFs). As a consequence though, the utility function is monotonic in all features; the proposed method does not allow for the monotonicity constraints to apply on only a subset of the features.

2.4 Ordinal Optimisation

Ordinal optimisation (OO) is an approach introduced in [92] for softening difficult problems in stochastic search and optimisation [177], and offered as a complementary approach to conventional optimisation techniques when there is 'little hope' of finding the global optimum solution. The outcome provided by OO is a high-probability guarantee that one or more out of a selected subset of candidate solutions is an acceptable sub-optimal solution, and its operation rests on two underlying principles: 1) by selecting the subset according to order, the selection is more 'robust' to noise; and 2) by 'goal softening' (i.e. increasing the degree of sub-optimality), chances of success can be improved.

OO was primarily introduced to the control theory community for the simulation-based optimisation of discrete-event dynamic systems [92], and has seen numerous successful applications in design/search problems across different disciplines. It was applied to stochastic optimal control in [55], where OO was used to find a heuristic solution to the Witsenhausen problem [203]. A proposed solution 50% better than Witsenhausen's own proposed solution was found in terms of performance cost. In [93], OO was applied

to rare event simulation of overflow probabilities in queuing systems. By embracing goal softening, computational requirements for simulation were reduced by approximately 3 orders of magnitude. An improvement by a factor of 100 was also reported by [209] in the time taken to generate optimal cloud computing schedules by an OO method, compared to a Monte-Carlo approach.

A research area with roots from OO is the optimal computing budget allocation (OCBA) framework, which addresses the problem of efficient allocation of simulation resources [39]. In the literature, the OCBA framework has also been referred to as ranking and selection [153], as well as "ordinal optimisation" [78]. The early results from the OCBA framework may also be regarded as a precursor of a pure exploration objective from the vast multi-armed bandit literature [120].

Some basic results in OO are summarised here, in the context of controller tuning. For some controlled system, let Θ be a finite set of possible values for the controller tuning variables, with cardinality $|\Theta| = N$. Suppose that, if we were to evaluate the performance of each of these controllers, there is a subset $\Theta^\star \subset \Theta$ of controllers with performance which we would consider 'acceptable', with cardinality $|\Theta^\star| = g$. Then if we were to sample a subset $\widehat{\Theta} \subset \Theta$ of m controllers (i.e. $\left|\widehat{\Theta}\right| = m$) uniformly randomly without replacement, then the probability that at least one of these selected m is in the top g controllers is

$$\Pr\left(\left|\widehat{\Theta} \cap \Theta^\star\right| \geq 1\right) = 1 - \Pr\left(\left|\widehat{\Theta} \cap \Theta^\star\right| = 0\right) \tag{2.17}$$

$$= 1 - \frac{\binom{N-g}{m}}{\binom{N}{m}}, \tag{2.18}$$

which is called the *blind pick* alignment probability. Now suppose that rather than selecting the m controllers uniformly randomly, we observe the performances of all N controllers (but with noise), and then select the best m observed (known as the *horse race* selection rule). Assume the observation process of each controller θ_i, for $i = 1, \ldots, N$, is

given by

$$\overline{J}\left(\theta_i\right) = J\left(\theta_i\right) + W_i, \tag{2.19}$$

where $J\left(\theta_i\right)$ is the 'true' performance of the system under the convention that lower is better (which determines the rankings between the θ_i an hence the top g), and W_i is i.i.d. observation noise. Denote the observed ordering within the subset of acceptable controllers:

$$J\left(\theta_{\{1\}}\right) + W_{\{1\}} \leq \cdots \leq J\left(\theta_{\{g\}}\right) + W_{\{g\}}, \tag{2.20}$$

i.e. $\theta_{\{i\}} \in \Theta^\star$ for each $i \in \{1, \ldots, g\}$, and the observed ordering outside the subset of observed controllers:

$$J\left(\theta_{\{g+1\}}\right) + W_{\{g+1\}} \leq \cdots \leq J\left(\theta_{\{N\}}\right) + W_{\{N\}}, \tag{2.21}$$

i.e. $\theta_{\{i\}} \notin \Theta^\star$ for each $i \in \{g+1, \ldots, N\}$. Then now for the top m selection $\widehat{\Theta}^*$, we have

$$\Pr\left(\left|\widehat{\Theta}^* \cap \Theta^\star\right| = 0\right) = \Pr\left(J\left(\theta_{\{g+m\}}\right) + W_{\{g+m\}} < J\left(\theta_{\{1\}}\right) + W_{\{1\}}\right). \tag{2.22}$$

It can be shown [94, §4.3.2] that the horse race alignment probability is lower bounded by the blind pick alignment probability:

$$\Pr\left(\left|\widehat{\Theta}^* \cap \Theta^\star\right| \geq 1\right) \geq 1 - \frac{\binom{N-g}{m}}{\binom{N}{m}}. \tag{2.23}$$

In the above, OO has been formulated as a search problem over a finite search space. That is, the variables which encode all the possible solutions take on values from a finite set Θ. There are of course problems where the search space will be infinite and possibly also uncountable (e.g. optimisation over the cone of positive definite matrices for tuning MPC). For these problems, OO can still be informally applied in practice by conditioning on the expected number of acceptable solutions in the sample, accompanied with the assurance that a large enough sample from the search space becomes 'representative' of the search space itself [123], and thus can be used as a good heuristic. However, a more stringent analysis and treatment of OO for uncountable search spaces is lacking.

2.5 Probabilistic Robust Control

The area of robust control is a richly developed sub-field of control for handling systems with uncertainties. Early on, the focus in the control community was on dealing with worst-case uncertainties, using set-theoretic descriptions of plants. Some time later, robust methods for dealing with probabilistic uncertainties, such as polynomial chaos expansion, began to emerge [114]. These methods can offer a trade-off between less-conservative bounds (which tend to be pessimistic in the worst-case) against some probabilistic risk [35].

2.5.1 Randomised Algorithms

When there is probabilistic plant uncertainty, the use of randomised algorithms (RA) is another well-established technique for finding approximate solutions to otherwise difficult computational problems [183]. A subset of these techniques goes by the name of the *scenario approach* to robust control [36]. Some key results from this area of probabilistic robust control pertain to the sample complexity, i.e. the number of simulations needed for the RA to perform analysis or design until desired probabilistic specifications are met. In control analysis, the techniques used to obtain sample complexities were originally pioneered in [113, 181], and based on well-known concentration inequalities such as the Chernoff bound. In control design, the first sample complexities for RA were provided in [195], using results from the paradigm of empirical risk minimisation in statistical learning theory (namely, using the Vapnik-Chervonenkis dimension). These randomised algorithms and variants thereof have seen numerous control applications. In fault detection, randomised algorithms have been used for finding a solution which ensures a low false alarm rate with high confidence [59]. In [10], randomised algorithms were used for the estimation and analysis for the probability of stability in high speed communication networks.

One basic randomised algorithm for probabilistically robust controller tuning is described as follows. Let $J(\psi, \theta) : \Psi \times \Theta \rightarrow [0, 1]$ be a system performance function with

a family of uncertain plants $\psi \in \Psi$ and a family of controllers $\theta \in \Theta$, under the convention that a smaller value of J is better. The plant ψ is treated as a random variable that is drawn when a test of the controller is conducted. One way to assess the controller performance is by the expected performance over the plant uncertainty

$$\overline{J}(\theta) := \mathbb{E}_{\psi}\left[J(\psi, \theta)\right]. \tag{2.24}$$

Unfortunately, evaluating this expectation is usually not tractable. However, it can still be estimated by drawing N i.i.d. samples $\psi_1, \ldots, \psi_N$ from the distribution of ψ and computing

$$\widehat{J}_N(\theta) := \frac{1}{N}\sum_{i=1}^{N} J(\psi_i, \theta). \tag{2.25}$$

A randomised algorithm for control design proceeds as follows. Sample M i.i.d. controllers $\theta_1, \ldots, \theta_M$ from Θ using some arbitrary mechanism. Denote

$$\widehat{\vartheta}_{N,M} := \underset{i=1,\ldots,M}{\operatorname{argmin}} \widehat{J}_N(\theta_i). \tag{2.26}$$

The following result [182, Theorem 10.3] says that for $\varepsilon, \epsilon, \delta \in (0,1)$, if the sample sizes are chosen sufficiently large with

$$M \geq \frac{\log(2/\delta)}{\log\left(1/(1-\epsilon)\right)} \tag{2.27}$$

and

$$N \geq \frac{\log \frac{4M}{\delta}}{2\varepsilon^2}, \tag{2.28}$$

then with confidence at least $1-\delta$,

$$\Pr_{\theta}\left(\overline{J}(\theta) < \widehat{J}_N\left(\widehat{\vartheta}_{N,M}\right) - \varepsilon\right) \leq \epsilon. \tag{2.29}$$

In the literature [196], the solution $\widehat{\vartheta}_{N,M}$ is known as an *approximate probable near minimiser*, and $\widehat{J}_N\left(\widehat{\vartheta}_{N,M}\right)$ is known as the *approximate probable near minimum*. This stipulates that given some risk δ, accuracy ε and level ϵ, then with high confidence (prescribed by

the risk δ), there is only a small chance (prescribed by the level ϵ) that a randomly sampled controller would outperform the **performance observed in simulation** of the obtained solution $\widehat{J}_N\left(\widehat{\vartheta}_{N,M}\right)$ (within slack prescribed by the accuracy ε). In other words, the empirical-averaged performance over plant uncertainty in simulation is 'close' to the non-empirical-averaged performance of an 'exceptional' subset of controllers.

If a controller $\widehat{\theta}^*$ has been obtained via offline tuning, it would seem useful to also have a bound that involves the **actual performance observed online** of the tuned controller, i.e. $J\left(\psi, \widehat{\theta}^*\right)$ from a realised plant ψ. However, such bounds do not yet appear to exist in the literature.

Several commonalities can also be observed between methods in OO and methods in RA for controller tuning. On the surface, they both seek to find approximate solutions to difficult design problems that are rendered intractable due to uncertainty. Moreover, they both employ a philosphy which can be roughly summarised as "randomly sample many candidate solutions, simulate their performances, and pick the best observed one". In OO, the optimality of this practice (the horse race rule) was formally shown in [207]. Additionally, [196] discusses that although this strategy is what seems intuitively to be the best thing to do (and what has been done for decades), the usage of RA is justified through rigorous sample complexity estimates. A further similarity shared by OO and RA is the notion of goal softening, which can be used to control the degree of suboptimality for the obtained solution. This apparent connection between OO and RA had been recognised and briefly touched on in [105, 113], but as of yet, has not been fully explored in the literature.

2.6 Research Aims

Before stating the research aims, we first summarise the literature presented above. In Section 2.1, a quadratic-cost formulation of MPC was introduced, and a meta-cost approach (as well as other approaches) to tuning were described in Section 2.1.2. In partic-

ular, the MLCTF [104] highlighted the use of preference learning to obtain a performance index for controllers. An account of preference learning and utility theory was given in Section 2.3.

Section 2.2 provided definitions for signals of interest in the diesel air-path, and summarised the current state of literature for diesel air-path modelling/control. Recounting from Chapter 1, physical engine experiments are costly/limited, so a diesel air-path model identified from a short amount of experimentation will lead to some model uncertainty. Sections 2.4 and 2.5 cover approaches from literature to address controller tuning in the presence of uncertainty, namely OO and RA.

Following this summary, we highlight several gaps in literature as they pertain to offline tuning of MPC, and propose research questions to address them.

2.6.1 Research Aim 1

As discussed in Section 2.1.2.4, a novelty proposed by the MLCTF is the learning of preferences from human experts. The original paper [104] considered this via a regression problem using numeric labels for ratings. In Section 2.3 however, motivation was provided for learning from pairwise ranking data, and the need to incorporate monotonicity constraints in learned utility functions in the case of desirable or undesirable features. Thus, there exists an opportunity to build upon the MLCTF, by learning preferences when data consists of pairwise comparisons, while also adhering to monotonicity constraints in the utility function.

Additionally, if a highly uncertain model is used in simulation for offline tuning, this may adversely impact the test performance of the tuned controller. Throughout Section 2.1.2, there have not been any studies which explicitly address the uncertainty in offline tuning of MPC, although the paradigms of OO and probabilistic robust control (covered in Sections 2.4 and 2.5 respectively) do offer a way to address the uncertainty in control design. However, their application firstly relies on having some characterisation of

the uncertainty in the plant model. Therefore, this forms an opportunity to augment the machine learning framework with uncertainty quantification, with respect to the plant model. This leads to the first research aim.

To augment the machine learning controller tuning framework with isotonic pairwise preference learning and plant uncertainty quantification.

This research aim is addressed throughout Chapters 3 and 4. Pairwise preference learning algorithms with monotonicity constraints are proposed in Chapter 3, and an active learning algorithm (which aims to reduce the number of experiments) for diesel air-path identification and uncertainty quantification is presented and demonstrated in Chapter 4.

2.6.2 Research Aim 2

In offline tuning of MPC, it would be desirable to have theoretical performance guarantees for the tuned controller on the actual plant, to the flavour of: "a well-performing MPC in simulation also performs well when tested on the actual plant". The existing theory not entirely suitable to provide such guarantees. The bounds and probabilities in OO are valid for tuning over finite space of controllers, however for quadratic-cost MPC this ideally requires tuning over an uncountable space of controllers. Also when using RA, the bound (2.29) involves the approximate probable near minimum, but not the actual test performance on a realised plant. Hence, an opportunity exists to develop an offline tuning approach for controllers over an uncountable space that provides some probabilistic performance guarantee of the tuned controller in a test on the actual plant. This leads to our second research aim.

To develop an ordinal optimisation approach valid for offline tuning of MPC, which is to be tested online.

This research aim is addressed throughout Chapters 5, 6 and 7. An analysis of OO

over uncountably many controllers is conducted in Chapter 6. A sequential learning algorithm for probabilistically robust controller tuning is also presented Chapter 7. The results in Chapters 6 and 7 rely on theoretical bounds and properties developed in Chapter 5.

2.6.3 Research Aim 3

The remaining research aim is an application of the proposed methods to tuning quadratic-cost MPC for a diesel air-path. Existing studies [128, 129] have already demonstrated the success of **online** MPC tuning approaches. Thus, the primary focus in this thesis is on the **offline** tuning application, which complements the online tuning method (e.g. by finding an initial good controller tuning for further online tuning). Our third research aim is stated as follows.

To implement and experimentally validate the efficacy of quadratic-cost MPC tuned offline from ordinal optimisation on a diesel engine test rig.

This research aim will involve designing an MPC architecture suitable for the control of the diesel air-path over transient drive-cycles, and implementing the architecture on an ECU. These experimental results arising from this research aim are presented in Chapter 6.

Chapter 3

Isotonic Preference Learning from Pairwise Comparisons

In this chapter, two methods are proposed for learning the preference function in the machine learning controller tuning framework (i.e. based on Figure 2.1), when the data comprises of pairwise comparisons, and monoticity constraints are to be enforced. The first method in Section 3.2 utilises Gaussian process regression. The second method in Section 3.3 relies on Bayesian estimation of a linear regression function using a Dirichlet prior. Both methods are demonstrated for learning preferences over features from diesel air-path trajectories, from pairwise comparison data.

Contributions in this chapter have been presented at the 2018 IEEE Conference on Decision and Control [43]

3.1 Problem Setup

Recall Section 2.3.1, where ordinal utility functions for a preference relation $\underset{\text{pref}}{\preceq}$ over a feature space $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_\mathtt{d}$ was presented in Definition 2.1. In what follows, we treat $\mathcal{X} \subset \mathbb{R}^\mathtt{d}$ and work with the following assumptions on the preferences of the human providing the rating data (whom we call the *user*).

Assumption 3.1. *(Debreu preferences) The users' preferences $\underset{\text{pref}}{\preceq}$ satisfy the conditions of Debreu's theorem [54, Theorem I] (i.e. $\mathcal{X}$ is a completely ordered, connected and separable space, and for each $\mathtt{x}' \in \mathcal{X}$ the sets $\left\{\mathtt{x} \in \mathcal{X} : \mathtt{x} \underset{\text{pref}}{\preceq} \mathtt{x}'\right\}$ and $\left\{\mathtt{x} \in \mathcal{X} : \mathtt{x}' \underset{\text{pref}}{\preceq} \mathtt{x}\right\}$ are both closed), so that there exists a continuous ordinal utility function that can be used to represent $\underset{\text{pref}}{\preceq}$.*

Remark 3.1. *Assumption 3.1 is not overly restrictive, but does serve the purpose of excluding cases where there does not exist a continuous utility function to represent users' preferences. A*

well-known example of preferences which do not satisfy Assumption 3.1 are Lexicographic preferences [90].

The notions of desirability/undesirability of features as discussed in Section 2.3.3 is formalised with the following definition.

Definition 3.1 (Monotonic preferences)**.** *A differentiable ordinal utility function* $\mathtt{h} : \mathcal{X} \to \mathbb{R}$ *for preferences* $\underset{\text{pref}}{\preceq}$ *is strictly monotonic at* $\mathtt{x}$ *in direction* j *if*

$$\frac{\partial \mathtt{h}(\mathtt{x})}{\partial [\mathtt{x}]_j} > 0 \tag{3.1}$$

while weak monotonicity is defined if (3.1) *holds with* $\geq$*.*

Assumption 3.2. *(Monotonicity) The users' preferences are strictly monotonic along dimensions given by the index set* $\mathcal{J} \subseteq \{1, \ldots, \mathtt{d}\}$*.*

Remark 3.2. *Assumption 3.2 is not restrictive because it arises from preconditioned knowledge (e.g. common sense or otherwise) of features exhibiting monotonicity, which is a motivating factor for the need to learn a monotonic utility function in the first place. For example, it is natural to treat the negative amount of overshoot and settling time as being desirable (i.e. higher is better), hence they should be monotonic in their preferences.*

Suppose there is a set $\{\mathtt{X}\} := \{\mathtt{x}_1, \ldots, \mathtt{x}_\mathtt{n}\}$ of $\mathtt{n}$ distinct items where each $\mathtt{x}_i \in \mathcal{X}$ for $i = 1, \ldots, \mathtt{n}$, from which items to compare are sampled from. Assume the data generating process follows the rating model in Assumption 3.3.

Assumption 3.3. *(Rating model) The user generates comparisons using the data generating process*

$$\mathtt{v}(\mathtt{x}_\mathtt{A}) := \mathtt{g}(\mathtt{x}_\mathtt{A}) + \mathtt{e}_\mathtt{A} \tag{3.2}$$

$$\mathtt{v}(\mathtt{x}_\mathtt{B}) := \mathtt{g}(\mathtt{x}_\mathtt{B}) + \mathtt{e}_\mathtt{B} \tag{3.3}$$

such that when shown items $\mathtt{x}_\mathtt{A}, \mathtt{x}_\mathtt{B} \in \{\mathtt{X}\}$ *for comparison,* $\mathtt{v}(\mathtt{x}_\mathtt{B}) > \mathtt{v}(\mathtt{x}_\mathtt{A})$ *means the user rates* $\mathtt{x}_\mathtt{B}$ *preferred over* $\mathtt{x}_\mathtt{A}$*, where* $\mathtt{e}_\mathtt{A}, \mathtt{e}_\mathtt{B} \sim \mathcal{N}\left(0, \sigma^2_{\text{rating}}\right)$ *are i.i.d. rating noise terms and* $\mathtt{g}(\cdot)$ *is the underlying utility function of the user. The reverse analogously holds if* $\mathtt{v}(\mathtt{x}_\mathtt{B}) < \mathtt{v}(\mathtt{x}_\mathtt{A})$*.*

The rating noise may be interpreted as inaccuracy of the user's judgement (e.g. due to imperceptible differences in alternatives or other psychological factors such as fatigue). If there are multiple inhomogeneous users providing ratings (as considered in [142]), then any differences between users' opinions can also be modelled using rating noise. The data consists of $\mathtt{M}$ comparisons, denoted by $\mathcal{D} = (\mathtt{X_A}, \mathtt{X_B})$ where $\mathtt{X_A} := (\mathtt{x}_{\mathtt{A}_1}, \dots, \mathtt{x}_{\mathtt{A}_M})$ and $\mathtt{X_B} := (\mathtt{x}_{\mathtt{B}_1}, \dots, \mathtt{x}_{\mathtt{B}_M})$, with indices $\mathtt{A}_1, \dots, \mathtt{B}_M \in \{1, \dots, \mathtt{n}\}$ such that the agent has rated $\mathtt{x}_{\mathtt{B}_i}$ preferred over $\mathtt{x}_{\mathtt{A}_i}$ for each $i = 1, \dots, \mathtt{M}$.

The learning problem is posed as follows:

Problem 3.1. *Given pairwise comparison data $\mathcal{D}$, learn a continuous ordinal utility function that exhibits strict monotonicity over all $\mathcal{X}$ in the respective dimensions given by $\mathcal{J}$.*

Observe that the monotonicity-constrained preference learning problem is trivial if $\mathtt{x}$ is of dimension $\mathtt{d} = 1$, as then the ordinal utility function can be chosen to be any strictly monotonic continuous function. Thus we are primarily interested in the case where the feature vector is of dimension $\mathtt{d} > 1$.

3.2 Isotonic Preference Learning via Gaussian Process Regression

Throughout this section, we develop an algorithm for isotonic preference learning from pairwise comparisons, using Gaussian process regression (Appendix A). Note thay monotonicity constraints have previously been considered in Gaussian processes regression [158], where 'virtual' derivative observations were injected into the data, however a guarantee of monotonicity for the posterior mean was not provided there.

3.2.1 Preference Learning via Gaussian Process Regression

We summarise an approach for learning a utility function from pairwise comparisons using Gaussian processes. The method is adapted from [49], except a generally non-zero prior mean is employed here. We seek an estimate for the vector of *latent* (i.e. unknown)

utilities

$$\vec{\mathtt{f}} := \begin{bmatrix} \mathtt{f}(\mathtt{x}_1) & \dots & \mathtt{f}(\mathtt{x}_\mathtt{n}) \end{bmatrix}^\top, \tag{3.4}$$

which are the evaluations from our estimated utility function $\mathtt{f}(\cdot)$ at each of the elements in $\{\mathtt{X}\}$. Denote the (normalised) difference in utility for the i^{th} comparison by

$$\mathtt{Z}_i := \frac{\mathtt{f}(\mathtt{x}_{\mathtt{B}_i}) - \mathtt{f}(\mathtt{x}_{\mathtt{A}_i})}{\sqrt{2}\sigma_{\text{rating}}}. \tag{3.5}$$

The estimate of $\vec{\mathtt{f}}$ is obtained by maximum a posteriori (MAP) estimation. This estimator is a function of the data, denoted by $\vec{\mathtt{f}}_{\text{MAP}}(\mathcal{D})$. In what follows, this dependency on the data will be suppressed for brevity of notation. Given a kernel function $\mathtt{k}(\mathtt{x}, \mathtt{x}')$ and prior mean function $\mathfrak{m}(\mathtt{x})$, the MAP estimate via Gaussian process regression is

$$\vec{\mathtt{f}}_{\text{MAP}} = \underset{\vec{\mathtt{f}}}{\operatorname{argmin}} \left\{ -\sum_{i=1}^{\mathtt{M}} \log \Phi(\mathtt{Z}_i) + \frac{1}{2} \left(\vec{\mathtt{f}} - \mathfrak{m}(\mathtt{X})\right)^\top \mathbf{K}^{-1} \left(\vec{\mathtt{f}} - \mathfrak{m}(\mathtt{X})\right) \right\}, \tag{3.6}$$

where $\mathtt{X} := \begin{bmatrix} \mathtt{x}_1 & \dots & \mathtt{x}_\mathtt{n} \end{bmatrix}^\top$, $\mathfrak{m}(\mathtt{X}) := \begin{bmatrix} \mathfrak{m}(\mathtt{x}_1) & \dots & \mathfrak{m}(\mathtt{x}_\mathtt{n}) \end{bmatrix}^\top$, $\mathbf{K}$ is the $\mathtt{n} \times \mathtt{n}$ Gram matrix of $\mathtt{X}$ using the kernel $\mathtt{k}(\cdot, \cdot)$ and $\Phi(\cdot)$ is the cumulative distribution function of the standard Gaussian. In practice, a small but positively scaled identity matrix can be added to $\mathbf{K}$ if it is ill-conditioned. Note that the optimisation problem in (3.6) is strictly convex and twice differentiable (as shown in [49]) and therefore can be solved in a straightforward manner using Newton-like or gradient methods. Once $\vec{\mathtt{f}}_{\text{MAP}}$ has been obtained, an approximate predictive posterior mean (via a Laplace approximation [155, §3.4]) at a test location $\mathtt{x}_*$ is given by

$$\widehat{\mathtt{f}}_*(\mathtt{x}_*) = \mathfrak{m}(\mathtt{x}_*) + \vec{\mathtt{k}}_*^\top \mathbf{K}^{-1} \left(\vec{\mathtt{f}}_{\text{MAP}} - \mathfrak{m}(\mathtt{X})\right), \tag{3.7}$$

where $\vec{\mathtt{k}}(\mathtt{X}, \mathtt{x}) := \begin{bmatrix} \mathtt{k}(\mathtt{x}_1, \mathtt{x}) & \dots & \mathtt{k}(\mathtt{x}_\mathtt{n}, \mathtt{x}) \end{bmatrix}^\top$ and $\vec{\mathtt{k}}_* := \vec{\mathtt{k}}(\mathtt{X}, \mathtt{x}_*)$.

As later demonstrated via Figure 3.1b however, this approach does not enforce monotonicity of the learned utility function.

3.2.2 Maximum Likelihood of a Linear Utility Function

By following an empirical Bayes approach [133] (in which a Bayesian prior is obtained from data), a maximum likelihood estimator (MLE) is proposed to obtain a prior mean function $\mathtt{m}(\mathtt{x})$. Let $\mathtt{q}_i : \mathcal{X}_i \to \mathbb{R}$, $i = 1, \ldots, \mathtt{d}$ be strictly monotonic functions, and define the operator $\overrightarrow{\mathtt{q}} : \mathcal{X} \to \mathbb{R}^{\mathtt{d}}$ with $\overrightarrow{\mathtt{q}}(\mathtt{x}) := (\mathtt{q}_1([\mathtt{x}]_1), \ldots, \mathtt{q}_{\mathtt{d}}([\mathtt{x}]_{\mathtt{d}}))$. We can then construct a model of the utility function linear in the basis $\overrightarrow{\mathtt{q}}(\mathtt{x})$:

$$\mathtt{g}(\mathtt{x}) = \mathtt{b}^\top \overrightarrow{\mathtt{q}}(\mathtt{x}), \tag{3.8}$$

where $\mathtt{b}$ is the vector of parameters to be estimated. By applying the rating model from Assumption 3.3, the maximum likelihood estimate of b under the monotonicity hypothesis in Assumption 3.2 involves solving the constrained problem

$$\begin{aligned} \widehat{\mathtt{b}} = \underset{\mathtt{b}}{\operatorname{argmin}} & \left\{ -\sum_{i=1}^{\mathtt{M}} \log \Phi \left(\frac{\mathtt{b}^\top \overrightarrow{\mathtt{q}}(\mathtt{x}_{\mathtt{B}_i}) - \mathtt{b}^\top \overrightarrow{\mathtt{q}}(\mathtt{x}_{\mathtt{A}_i})}{\sqrt{2}\sigma_{\text{rating}}} \right) \right\} \\ \text{subject to } & \mathtt{b}_j > 0, \ \forall j \in \mathcal{J}. \end{aligned} \tag{3.9}$$

Using a linear-in-basis form for the utility function is the usual approach taken in discrete choice theory [187], however as we later demonstrate, by using this form only as a prior, the overall estimate can be improved upon.

Remark 3.3. *The convexity of* (3.9) *is readily shown using the log-concavity of the Gaussian distribution [28]. Twice differentiability also holds and Newton-like or gradient methods can be used to solve* (3.9).

Remark 3.4. *We can fix $\sigma_{\text{rating}} > 0$ without loss of generality as this only affects the scale of the learned utility function (as in [142]). To illustrate, notice that scaling both $\mathtt{g}(\cdot)$ and σ_{rating} from Assumption 3.3 by the same positive constant will not change the distribution of $\mathcal{D}$. The learning problem is unaffected as we are only concerned with learning an ordinal utility function. So for convenience, we may choose $\sigma_{\text{rating}} = 1/\sqrt{2}$.*

Remark 3.5. *In numerical implementation of* (3.9)*, it may be required to enforce monotonicity with the constraint $\mathtt{b}_j \geq \mathtt{c}$ with some small $\mathtt{c} > 0$.*

After $\widehat{\mathtt{b}}$ has been obtained, we can express the estimate of the vector of latent utilities as the matrix-vector multiplication $\mathtt{Q}(\mathtt{X})\widehat{\mathtt{b}}$, where $\mathtt{Q}(\mathtt{X}) := \left[\overrightarrow{\mathtt{q}}(\mathtt{x}_1) \quad \ldots \quad \overrightarrow{\mathtt{q}}(\mathtt{x}_n)\right]^\top$. We also choose the prior mean function $\mathtt{m}(\mathtt{x}) = \widehat{\mathtt{b}}^\top \overrightarrow{\mathtt{q}}(\mathtt{x})$ and our proposed utility function estimate takes the form

$$\widehat{\mathtt{f}}_*(\mathtt{x}_*) = \widehat{\mathtt{b}}^\top \overrightarrow{\mathtt{q}}(\mathtt{x}_*) + \overrightarrow{\mathtt{k}}_*^\top \Xi \overrightarrow{\mathtt{z}}, \tag{3.10}$$

with $\Xi := \mathbf{K}^{-1}$, $\overrightarrow{\mathtt{z}} := \overrightarrow{\mathtt{y}} - \mathtt{Q}(\mathtt{X})\widehat{\mathtt{b}}$, and $\overrightarrow{\mathtt{y}}$ is a choice of vector for the latent utilities. Note that this closely resembles the form of the approximate predictive posterior mean in (3.7).

3.2.3 Monotonicity Conditions on Gaussian Process Regression

In this section, we state conditions for strict monotonicity of the utility function estimate. As it produces differentiable sample functions of both the prior and posterior, the kernel function being considered is the squared exponential kernel

$$\mathtt{k}(\mathtt{x}, \mathtt{x}') = \mathtt{s}^2 \exp\left(-\frac{1}{2}(\mathtt{x} - \mathtt{x}')^\top \Lambda^{-1} (\mathtt{x} - \mathtt{x}')\right), \tag{3.11}$$

where $\Lambda = \operatorname{diag}\{\ell_1, \ldots, \ell_\mathtt{d}\}$ and $\mathtt{s} > 0$, $\ell_1, \ldots, \ell_d > 0$ are hyperparameters. We focus on conditions pertaining to the predictive function (3.10). Let $\left[\frac{\partial \overrightarrow{\mathtt{k}}_*^\top}{\partial \mathtt{x}_*}\right]_j$ denote the j^{th} row of the $\mathtt{d} \times \mathtt{n}$ matrix of partial derivatives $\frac{\partial \overrightarrow{\mathtt{k}}_*^\top}{\partial \mathtt{x}_*}$. From Definition 3.1, the predictive function (3.10) is strictly monotonic at $\mathtt{x}$ in direction j if

$$\widehat{\mathtt{b}}_j \left.\frac{\partial \mathtt{q}_j(\mathtt{x}_j)}{\partial [\mathtt{x}]_j}\right|_{\mathtt{x}_j = \mathtt{x}_{*j}} + \left[\frac{\partial \overrightarrow{\mathtt{k}}_*^\top}{\partial \mathtt{x}_*}\right]_j \Xi \overrightarrow{\mathtt{z}} > 0. \tag{3.12}$$

For the squared exponential kernel and a choice of prior mean function affine in $\mathtt{x}$ (i.e. $\mathtt{m}(\mathtt{x}) = \widehat{\mathtt{b}}^\top \mathtt{x}$), the condition (3.12) becomes

$$0 < \widehat{\mathtt{b}} - \Lambda^{-1}\left[(\mathtt{x}_* - \mathtt{x}_1)\,\mathtt{k}(\mathtt{x}_*, \mathtt{x}_1) \quad \ldots \quad (\mathtt{x}_* - \mathtt{x}_\mathtt{n})\,\mathtt{k}(\mathtt{x}_*, \mathtt{x}_\mathtt{n})\right] \Xi \overrightarrow{\mathtt{z}}. \tag{3.13}$$

3.2.4 Monotonicity Constrained Estimates

We propose an approach to conduct preference learning from pairwise comparisons with a guarantee of strict monotonicity along any amount of directions as desired. For simplicity, the form of the prior mean function is chosen to be affine in $\mathtt{x}$ so that $\mathtt{m}(\mathtt{x}) = \widehat{\mathtt{b}}^\top \mathtt{x}$ and $\overrightarrow{\mathtt{z}} = \overrightarrow{\mathtt{y}} - \mathtt{X}\widehat{\mathtt{b}}$. For the condition in (3.13), observe that when $\widehat{\mathtt{b}}_j > 0$, the inequality will eventually be satisfied as $\overrightarrow{\mathtt{z}} \to \mathbf{0}$ (or equivalently as $\overrightarrow{\mathtt{y}} \to \mathtt{X}\widehat{\mathtt{b}}$). We state the strict monotonicity guarantee formally as follows.

Theorem 3.1. *With choice of prior mean function affine in* $\mathtt{x}$*, let* $\overrightarrow{\mathtt{f}}_{\text{lin}} := \mathtt{X}\widehat{\mathtt{b}}$ *as obtained from solving* (3.9) *and* $\overrightarrow{\mathtt{f}}_{\text{MAP}}$ *as from solving* (3.6)*. There exists an interval* $(\mathtt{a}^*, 1]$ *with*

$$\mathtt{a}^* = \max_{j \in \mathcal{J}} \left\{ \frac{\widehat{\mathtt{b}}_j}{-\widehat{\mathtt{b}}_j + \mathfrak{g}_j} \right\} + 1 \tag{3.14}$$

$$\mathfrak{g}_j := \min \left\{ 0, \inf_{\mathtt{x} \in \mathcal{X}} \left\{ \left[\frac{\partial \overrightarrow{\mathtt{k}}(\mathtt{X}, \mathtt{x})^\top}{\partial \mathtt{x}} \right]_j \Xi \left(\overrightarrow{\mathtt{f}}_{\text{MAP}} - \mathtt{X}\widehat{\mathtt{b}} \right) \right\} + \widehat{\mathtt{b}}_j \right\} \tag{3.15}$$

such that for all $\mathtt{a} \in (\mathtt{a}^*, 1]$*, using the convex combination* $\overrightarrow{\mathtt{y}} = \mathtt{a}\overrightarrow{\mathtt{f}}_{\text{lin}} + (1 - \mathtt{a})\overrightarrow{\mathtt{f}}_{\text{MAP}}$ *in* (3.10) *satisfies* (3.12) *for all* $j \in \mathcal{J}$ *over all* $\mathtt{x} \in \mathcal{X}$*.*

Proof. We have strict monotonicity in direction j for all $\mathtt{x} \in \mathcal{X}$ if

$$\widehat{\mathtt{b}}_j + \inf_{\mathtt{x} \in \mathcal{X}} \left\{ \left[\frac{\partial \overrightarrow{\mathtt{k}}(\mathtt{X}, \mathtt{x})^\top}{\partial \mathtt{x}} \right]_j \Xi \left(\overrightarrow{\mathtt{y}} - \mathtt{X}\widehat{\mathtt{b}} \right) \right\} > 0. \tag{3.16}$$

For ease of notation, define $\mathfrak{g}_j(\mathtt{x}) := \left[\frac{\partial \overrightarrow{\mathtt{k}}(\mathtt{X}, \mathtt{x})^\top}{\partial \mathtt{x}} \right]_j \Xi \left(\overrightarrow{\mathtt{f}}_{\text{MAP}} - \mathtt{X}\widehat{\mathtt{b}} \right)$. Starting from the monotonicity condition (3.16) and using $\overrightarrow{\mathtt{y}} = \mathtt{a}\mathtt{X}\widehat{\mathtt{b}}_j + (1 - \mathtt{a})\overrightarrow{\mathtt{f}}_{\text{MAP}}$, rearranging for $\mathtt{a}$ in the case when $\widehat{\mathtt{b}}_j + \inf_{\mathtt{x} \in \mathcal{X}} \{\mathfrak{g}_j(\mathtt{x})\} \leq 0$ yields

$$\mathtt{a} > \frac{\widehat{\mathtt{b}}_j}{\inf_{\mathtt{x} \in \mathcal{X}} \{\mathfrak{g}_j(\mathtt{x})\}} + 1, \tag{3.17}$$

while noting that $\widehat{\mathtt{b}}_j > 0$ so $\inf_{\mathtt{x} \in \mathcal{X}} \{\mathfrak{g}_j(\mathtt{x})\} < 0$. Let $\mathtt{a}_j^* = \widehat{\mathtt{b}}_j / \inf_{\mathtt{x} \in \mathcal{X}} \{\mathfrak{g}_j(\mathtt{x})\} + 1 \geq 0$, which satisfies (3.17) for all $\mathtt{x} \in \mathcal{X}$ if $\mathtt{a} \in \left(\mathtt{a}_j^*, 1 \right]$. For the case when $\widehat{\mathtt{b}}_j + \inf_{\mathtt{x} \in \mathcal{X}} \{\mathfrak{g}_j(\mathtt{x})\} > 0$, it is

enough to let $\mathtt{a}_j^* = 0$. To combine both cases we write

$$\mathtt{a}_j^* = \frac{\widehat{\mathtt{b}}_j}{-\widehat{\mathtt{b}}_j + \mathfrak{g}_j} + 1, \tag{3.18}$$

where $\mathfrak{g}_j$ is given in (3.15). Then $\mathtt{a}^* = \max_{j \in \mathcal{J}} \mathtt{a}_j^*$ finds the value such that for all $\mathtt{a} \in (\mathtt{a}^*, 1]$, strict monotonicity is satisfied in all directions $j \in \mathcal{J}$ over all $\mathtt{x} \in \mathcal{X}$, yielding (3.14). □

This result shows that we can find values of $\mathtt{a}$ (and hence $\overrightarrow{\mathtt{y}}$) which will satisfy the monotonicity constraint. A value of $\mathtt{a} = 1$ is the most conservative, resulting in the predictive function being identical to the prior mean, but is a value that always ensures the monotonicity constraints are satisfied. However, there is merit in choosing an $\mathtt{a}$ as low as possible (whilst satisfying monotonicity constraints), as indicated by Corollary 3.1, which follows Theorem 3.2.

Theorem 3.2. *Denote* $\overrightarrow{\mathtt{y}}_\mathtt{a} := \mathtt{a}\overrightarrow{\mathtt{f}}_{\text{lin}} + (1 - \mathtt{a})\overrightarrow{\mathtt{f}}_{\text{MAP}}$*, and define* $\mathtt{Z}_{i,\text{MAP}}$, $\mathtt{Z}_{i,\mathtt{a}}$, $\mathtt{Z}_{i,\text{lin}}$ *as in* (3.5) *however using the vectors* $\overrightarrow{\mathtt{f}}_{\text{MAP}}$, $\overrightarrow{\mathtt{y}}_\mathtt{a}$ *and* $\overrightarrow{\mathtt{f}}_{\text{lin}}$ *respectively. Suppose* $\overrightarrow{\mathtt{f}}_{\text{MAP}} \neq \overrightarrow{\mathtt{f}}_{\text{lin}}$*. Then the negative log likelihoods (or equivalently, the Kullback-Leibler divergences from the empirical distribution of comparisons) satisfy*

$$-\sum_{i=1}^{\mathtt{M}} \log \Phi\left(\mathtt{Z}_{i,\text{MAP}}\right) < -\sum_{i=1}^{\mathtt{M}} \log \Phi\left(\mathtt{Z}_{i,\mathtt{a}}\right) < -\sum_{i=1}^{\mathtt{M}} \log \Phi\left(\mathtt{Z}_{i,\text{lin}}\right) \tag{3.19}$$

for any $\mathtt{a} \in (0, 1)$*. If* $\overrightarrow{\mathtt{f}}_{\text{MAP}} = \overrightarrow{\mathtt{f}}_{\text{lin}}$ *then* (3.19) *holds but with equality.*

Proof. The case with $\overrightarrow{\mathtt{f}}_{\text{MAP}} = \overrightarrow{\mathtt{f}}_{\text{lin}}$ is trivial. Let $\mathtt{J}\left(\overrightarrow{\mathtt{f}}\right)$ denote the cost function in (3.6). For the case $\overrightarrow{\mathtt{f}}_{\text{MAP}} \neq \overrightarrow{\mathtt{f}}_{\text{lin}}$, use by definition $\min_{\overrightarrow{\mathtt{f}}} \mathtt{J}\left(\overrightarrow{\mathtt{f}}\right) = \mathtt{J}\left(\overrightarrow{\mathtt{f}}_{\text{MAP}}\right) < \mathtt{J}\left(\overrightarrow{\mathtt{f}}_{\text{lin}}\right)$ along with the strict convexity property of $\mathtt{J}\left(\overrightarrow{\mathtt{f}}\right)$ to establish $\mathtt{J}\left(\overrightarrow{\mathtt{f}}_{\text{MAP}}\right) < \mathtt{J}\left(\overrightarrow{\mathtt{y}}_\mathtt{a}\right) < \mathtt{J}\left(\overrightarrow{\mathtt{f}}_{\text{lin}}\right)$ for $\mathtt{a} \in (0, 1)$. Then apply the fact that the weighted norm

$$\left\|\overrightarrow{\mathtt{y}}_\mathtt{a} - \mathtt{X}\widehat{\mathtt{b}}\right\|_{\mathbf{K}} = (1 - \mathtt{a})\left\|\left(\overrightarrow{\mathtt{f}}_{\text{MAP}} - \overrightarrow{\mathtt{f}}_{\text{lin}}\right)\right\|_{\mathbf{K}} \geq 0 \tag{3.20}$$

is strictly monotonically decreasing in $\mathtt{a}$ for $\mathtt{a} \in (0, 1)$. □

Corollary 3.1. *If* $\overrightarrow{\mathtt{f}}_{\mathrm{MAP}} \neq \overrightarrow{\mathtt{f}}_{\mathrm{lin}}$*, then for all* $\mathtt{a}' < \mathtt{a}$

$$-\sum_{i=1}^{\mathtt{M}} \log \Phi\left(\mathtt{Z}_{i,\mathtt{a}'}\right) < -\sum_{i=1}^{\mathtt{M}} \log \Phi\left(\mathtt{Z}_{i,\mathtt{a}}\right). \tag{3.21}$$

Remark 3.6. *It is not trivial to find a class of problems such that there is a reasonable probability that* $\overrightarrow{\mathtt{f}}_{\mathrm{MAP}} = \overrightarrow{\mathtt{f}}_{\mathrm{lin}}$*. To do so requires investigating the intersection of the supports for the sampling distributions of* $\overrightarrow{\mathtt{f}}_{\mathrm{MAP}}$ *and* $\overrightarrow{\mathtt{f}}_{\mathrm{lin}}$*, which may or may not be the empty set. The supports of the sampling distributions themselves have cardinality in the order of* $2^{\mathtt{M}}$*, so even if the intersection is not the empty set, the probability should still be very small for reasonably large* $\mathtt{M}$*. Hence we argue that for most problems of interest (i.e. where* $\mathtt{M}$ *is not small), the probability that* $\overrightarrow{\mathtt{f}}_{\mathrm{MAP}} = \overrightarrow{\mathtt{f}}_{\mathrm{lin}}$ *will either be zero or can be considered negligibly small.*

The revelation of results (3.19) and (3.21) is that we should choose $\mathtt{a}$ as low as possible (whilst satisfying monotonicity constraints, i.e. $\mathtt{a} > \mathtt{a}^*$) and that the resulting empirical fit as measured by the likelihood will be an improvement over using $\overrightarrow{\mathtt{f}}_{\mathrm{lin}}$. A procedure based on this principle which finds a weighting between the linear and MAP estimates to guarantee strict monotonicity is described in Algorithm 3.1.

Algorithm 3.1 Preference Learning with Strict Monotonicity Constraints

Require: Data set $\mathcal{D}$, minimal distinct items matrix $\mathtt{X}$, monotonicity constraint index set $\mathcal{J}$

1: Choose hyperparameters $\mathtt{s} > 0$, $\sigma_{\mathrm{rating}} = 1/\sqrt{2}$, $\ell_1, \ldots, \ell_d > 0$
2: Choose small $\mathfrak{e} \in (0, 1]$
3: Obtain estimate $\widehat{\mathtt{b}}$ via (3.9) for the model $\mathtt{b}^\top \mathtt{x}$
4: Choose $\mathtt{r} > 0$
5: $\widehat{\mathtt{b}} \leftarrow \mathtt{r}\widehat{\mathtt{b}}$, $\sigma_{\mathrm{noise}} \leftarrow \mathtt{r}\sigma_{\mathrm{noise}}$ ▷ Normalise to a nominated scale
6: $\overrightarrow{\mathtt{f}}_{\mathrm{lin}} \leftarrow \mathtt{X}\widehat{\mathtt{b}}$
7: Obtain $\overrightarrow{\mathtt{f}}_{\mathrm{MAP}}$ via (3.6) using the prior mean $\widehat{\mathtt{b}}^\top \mathtt{x}$
8: $\mathtt{a} \leftarrow \mathtt{a}^*$ from (3.14)
9: $\mathtt{a} \leftarrow \min\{\mathtt{a} + \mathfrak{e}, 1\}$ ▷ For strict monotonicity
10: $\overrightarrow{\mathtt{y}} \leftarrow \mathtt{a}\overrightarrow{\mathtt{f}}_{\mathrm{lin}} + (1 - \mathtt{a})\overrightarrow{\mathtt{f}}_{\mathrm{MAP}}$
11: Compute the estimated utility function with (3.10)

3.2.5 Application to Trajectory Rating Data

We apply Algorithm 3.1 to learning preferences over diesel air-path trajectories. The available dataset is the same that is used in [103], and consists of 240 step response trajectories for the pair of boost pressure and EGR rate output signals $(\mathrm{p_{im}}, \mathrm{y_{EGR}})$ with corresponding numeric rating labels from $[0, 10]$. To obtain a pairwise ranking dataset suitable for Algorithm 3.1, a bootstrapped dataset was generated by randomly selecting 120 pairs of data points without replacement, and synthesising a comparison using the provided label, plus some small rating noise (standard deviation 0.01). The feature vector was selected to be integral of absolute error for each trajectory:

$$\mathrm{IAE}_{\mathrm{p_{im}}} = \mathsf{t}_{\mathrm{sample}} \sum_{k=0}^{\mathsf{T}-1} \left| \mathrm{p_{im},}_k - \mathrm{p}_{\mathrm{im}}^{\mathrm{ref}} \right| \tag{3.22}$$

$$\mathrm{IAE}_{\mathrm{y_{EGR}}} = \mathsf{t}_{\mathrm{sample}} \sum_{k=0}^{\mathsf{T}-1} \left| \mathrm{y_{EGR},}_k - \mathrm{y}_{\mathrm{EGR}}^{\mathrm{ref}} \right|, \tag{3.23}$$

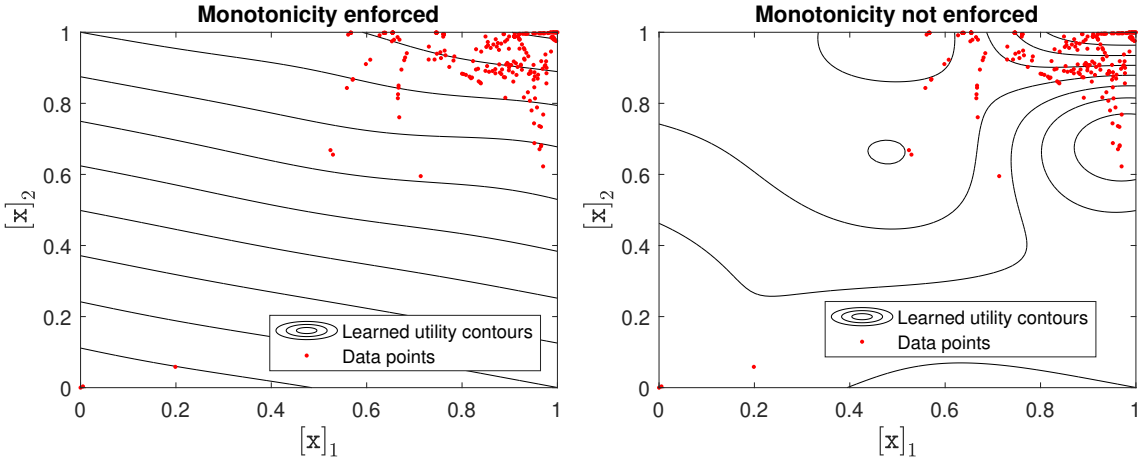
where $\mathsf{t}_{\mathrm{sample}}$ is the sample time and T is the total number of samples in the trajectory. The negative of the feature pair $(\mathrm{IAE}_{\mathrm{p_{im}}}, \mathrm{IAE}_{\mathrm{y_{EGR}}})$ was taken (so that the higher, the better) and normalised between $[0, 1]^2$ (by max/min normalisation across the whole dataset) so that the transformed feature pairs $([\mathtt{x}]_1, [\mathtt{x}]_2)$ were fed into Algorithm 3.1. Thus, Algorithm 3.1 is applied with $\mathtt{d} = 2$, $\mathtt{n} = 240$ and $\mathtt{M} = 120$.

Figure 3.1a shows the contours of the estimated utility function using the vector $\overrightarrow{\mathtt{y}}_{\mathtt{a}^*+\mathfrak{e}}$, which is guaranteed to satisfy strict monotonicity. On the other hand Figure 3.1b shows the contours when the vector $\overrightarrow{\mathtt{f}}_{\mathrm{MAP}}$ was used, which does not exhibit monotonicity. The normalised coefficient estimates in the linear prior were

$$\widehat{\mathtt{b}} = \begin{bmatrix} 0.1997 & 0.8003 \end{bmatrix}^\top. \tag{3.24}$$

When converted back into its unnormalised scale, the coefficients become

$$\widehat{\mathtt{b}}_{\mathrm{scaled}} = \begin{bmatrix} 0.0019 & 0.4659 \end{bmatrix}^\top. \tag{3.25}$$



(a) A contour plot of the estimated utility function.

(b) A contour plot of the estimated utility function without enforced monotonicity constraints.

Figure 3.1: Comparison of learned utility functions.

3.3 Bayesian Isotonic Regression from Pairwise Comparisons

In the previous section, Algorithm 3.1 was successfully demonstrated in $\mathtt{d} = 2$ dimensions. However, the method may not scale well to higher dimensions of the feature space, since there may be no efficient way to find $\mathtt{a}^*$ in Algorithm 3.1, over brute force search. Here, we describe an alternative approach for isotonic regression from pairwise comparisons, based on a Dirichlet prior. The index set $\mathcal{J}$ is now required to be in all the dimensions $\{1, \ldots, \mathtt{d}\}$. The intention is to find a linear utility function of the form

$$\mathtt{f}(\mathtt{x}) = \mathtt{b}^\top \mathtt{x}, \tag{3.26}$$

where $\mathtt{b} \geq \mathbf{0}$. Since we aim to find an ordinal utility function, we can fix the scale and constrain $\mathtt{b}$ to lie on the *standard simplex* (i.e. the probability simplex), such that $\mathtt{b}^\top \mathbf{1} = 1$. Then an appropriate choice of prior to place over $\mathtt{b}$ is the Dirichlet prior denoted $\mathrm{Dirichlet}(\cdot)$; we write

$$\mathtt{p}_{\mathrm{prior}}(\mathtt{b}) = \mathrm{Dirichlet}(\mathtt{b}). \tag{3.27}$$

Re-using the likelihood implicit from (3.9) above, we have

$$\mathrm{p}_{\mathrm{lik}}\left(\mathcal{D}|\mathtt{b}\right) = \prod_{i=1}^{\mathtt{M}} \Phi\left(\frac{\mathtt{b}^{\top}\mathtt{x}_{\mathtt{B}_i} - \mathtt{b}^{\top}\mathtt{x}_{\mathtt{A}_i}}{\sqrt{2}\sigma_{\mathrm{rating}}}\right). \tag{3.28}$$

Thus the posterior up to a constant of proportionality is

$$\mathrm{p}_{\mathrm{post}}\left(\mathtt{b}|\mathcal{D}\right) \propto \mathrm{p}_{\mathrm{lik}}\left(\mathcal{D}|\mathtt{b}\right)\mathrm{p}_{\mathrm{prior}}\left(\mathtt{b}\right). \tag{3.29}$$

The analytical form of the exact posterior is intractable, however Markov Chain Monte-Carlo (MCMC) can be performed to draw samples from the posterior. The σ_{rating} is a hyperparameter, however it can be empirically chosen (e.g. similar to Algorithm 3.1, but with a normalisation constant that exactly makes $\mathtt{b}$ sum to one).

3.3.0.1 Application to Trajectory Rating Data

We use a pairwise comparison data bootstrapped in a similar manner to Section 3.2.5, however we now generate an eight-dimensional feature vector of undesirable (negated to be desirable) time-domain characteristics:

$$\mathtt{x} = -\left(\mathtt{RT}_1, \mathtt{ST}_1, \mathtt{OS}_1, \mathtt{US}_1, \mathtt{RT}_2, \mathtt{ST}_2, \mathtt{OS}_2, \mathtt{US}_2\right), \tag{3.30}$$

where the definition of each feature is given in Table 3.1. The Dirichlet prior is set to be

Table 3.1: Description of time-domain characteristics used in the meta-cost, for a trajectory starting from its initial condition relative to a final value of 0.

Characteristic	Description
$\mathtt{RT}_1$	10% to 90% rise time (seconds) of p_{im}
$\mathtt{ST}_1$	2% settling time (seconds) of p_{im}
$\mathtt{OS}_1$	Overshoot (proportion) of p_{im}
$\mathtt{US}_1$	Undershoot (proportion) of p_{im}
$\mathtt{RT}_2$	10% to 90% rise time (seconds) of $\mathrm{y}_{\mathrm{EGR}}$
$\mathtt{ST}_2$	2% settling time (seconds) of $\mathrm{y}_{\mathrm{EGR}}$
$\mathtt{OS}_2$	Overshoot (proportion) of $\mathrm{y}_{\mathrm{EGR}}$
$\mathtt{US}_2$	Undershoot (proportion) of $\mathrm{y}_{\mathrm{EGR}}$

the 'flat' Dirichlet prior, i.e. with all parameters equal to one. After forming the likelihood and the prior, we obtain 200 i.i.d. samples drawn approximately from the posterior using the Metropolis-Hastings MCMC algorithm [160, §7.3], and average out the samples to obtain an estimate of the posterior mean $\widehat{\mathtt{b}}$. A histogram of marginal posterior samples compared against is marginal prior is displayed in Figure 3.2.

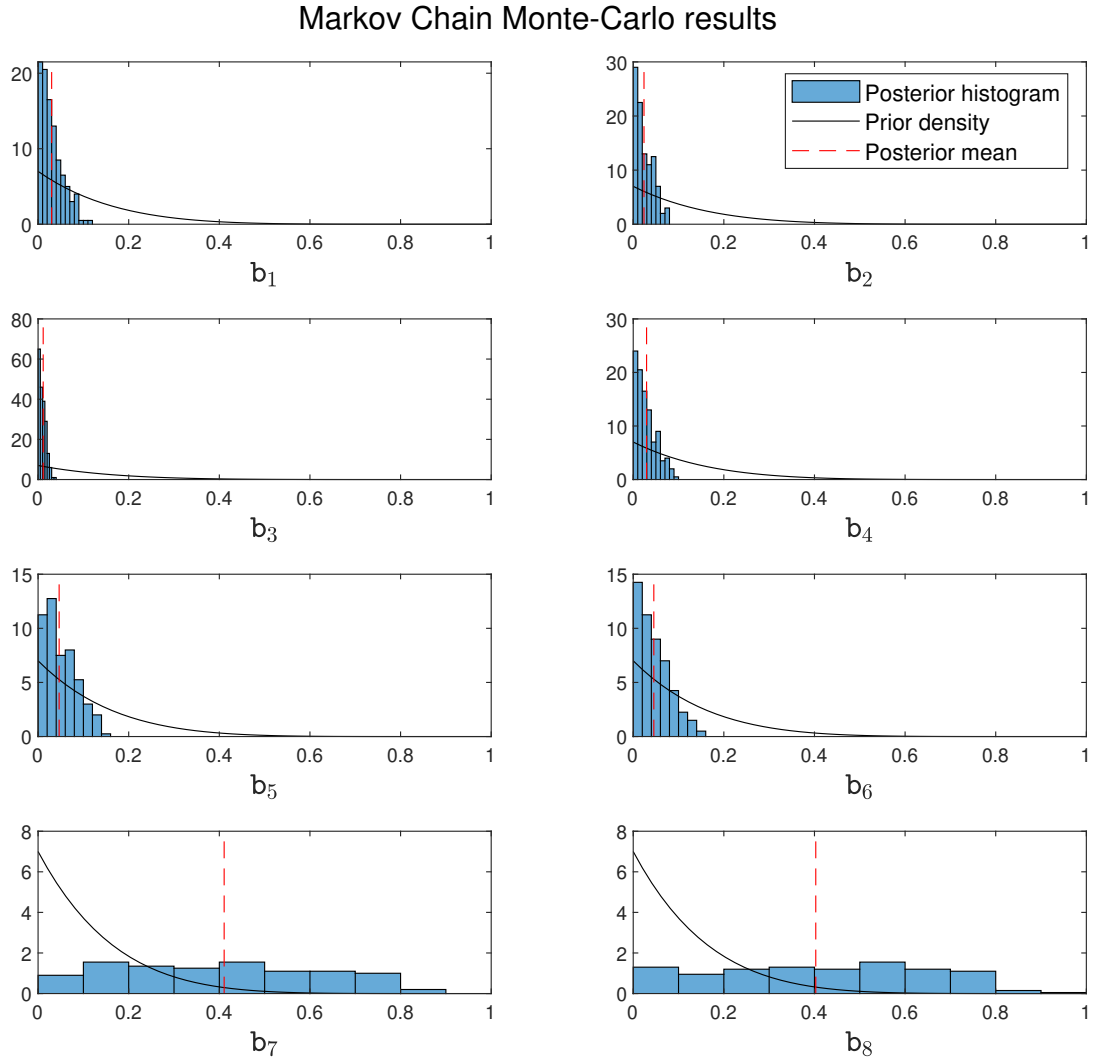


Figure 3.2: Histograms of samples approximately from the posterior, compared against density curves of the priors.

Plugging-in the posterior mean, the resulting ordinal utility function is

$$\mathtt{f}\left(\mathtt{x}\right) = -\left(0.03\mathtt{RT}_1 + 0.0238\mathtt{ST}_1 + 0.0113\mathtt{OS}_1 + 0.0295\mathtt{US}_1\right.$$

$$+0.0465\mathtt{RT}_2 + 0.0455\mathtt{ST}_2 + 0.4107\mathtt{OS}_2 + 0.4029\mathtt{US}_2)\,. \quad (3.31)$$

3.4 Summary

In this chapter, we demonstrated two methods for isotonic preference learning from pairwise comparisons. The first used Gaussian process regression and enforced monotonicity by finding a weighted average between the MAP and linear MLE estimates. The second used a Dirichlet prior for Bayesian estimation of a linear function, which potentially scales better with larger $\mathtt{d}$. Both methods were exemplified on diesel air-path trajectory rating data.

Chapter 4

Active Learning for Linear Parameter Varying System Identification

In this chapter, active learning is proposed for selection of the next operating points in the design of experiments, for identifying linear parameter-varying systems. Our approach is based on exploiting the probabilistic features of Gaussian process regression to quantify the overall model uncertainty across locally identified models. This results in a flexible methodology which accommodates for various techniques to be applied for estimation of local linear models and their corresponding uncertainty. In Section 4.3, we perform active learning with application to the identification of a diesel engine air-path plant model for the machine learning controller tuning framework, and demonstrate that measures of model uncertainty can be successfully reduced using the proposed methods.

Contributions in this chapter have been presented at the 2020 IFAC World Congress [42].

4.1 Problem Setup

Recall the discussion in Section 2.2.1.3 that linear parameter-varying (LPV) systems are suitable for modelling the diesel air-path, which are parametrised in the operating point. We thus consider noisy discrete-time LPV systems of the following form:

$$x_{k+1} = A(p)\,x_k + B(p)\,u_k + w_k \tag{4.1}$$

$$y_k = Cx_k, \tag{4.2}$$

with state $x_k \in \mathbb{R}^n$, input $u_k \in \mathbb{R}^m$, output $y_k \in \mathbb{R}^p$ and with noise/unmodelled disturbance sequence w_k. The operating point $p \in \mathcal{P} \subset \mathbb{R}^d$ parametrises the system matrices $A(p)$ and $B(p)$, the latter two which are the objects of interest to be identified. For the

identification problem, we make the following assumptions.

Assumption 4.1. *The operating space* $\mathfrak{P} \subset \mathbb{R}^{d}$ *is a compact region.*

Assumption 4.2. *The functions* $A : \mathfrak{P} \to \mathbb{R}^{n \times n}$ *and* $B : \mathfrak{P} \to \mathbb{R}^{n \times m}$ *are smooth.*

Assumption 4.3. *The matrix* C *is known and we have access to the full state measurement* x_k.

Assumption 4.4. *For all* $p \in \mathfrak{P}$, *the system* (4.1) *is stable and the noise* w_k *is an independent and identically distributed (i.i.d.) sequence with covariance matrix* $E(p)$.

In our formulation, Assumption 4.3 ensures the system order n is known and the state-space realisation is specified, so identification of (4.1) for fixed p becomes a special case of VARX (vector autoregression with exogenous inputs) regression, where identifiability issues arising from unknown state-space realisation do not become a concern. Also note by Assumption 4.4 that we do not necessarily require the noise to be Gaussian.

An implementation of MPC for the system (4.1) requires some knowledge about the matrices $A(p)$ and $B(p)$. As these are often not known in practice, they would be replaced by their estimates $\widehat{A}(p)$ and $\widehat{B}(p)$ which have been obtained from experiments. Doing so introduces some uncertainty in the predictions (in the form of variance), attributed to variance in the estimates for $A(p)$ and $B(p)$. This motivates our problem herein, which is to devise a method that quantifies the uncertainty in the estimates $\widehat{A}(p)$ and $\widehat{B}(p)$, and simultaneously leverages this to decide the next operating point to conduct an experiment at.

We use Gaussian process regression (Appendix A), which has previously been applied in active learning settings [31] and also in uncertainty quantification [24]. A Gaussian process on d-variate operating point $p \in \mathbb{R}^{d}$ may be denoted by:

$$f(p) \sim \mathcal{GP}\left(m(p), k\left(p, p'\right)\right) \tag{4.3}$$

with mean function $m(p) : \mathbb{R}^{d} \to \mathbb{R}$ and covariance function $k(p, p') : \mathbb{R}^{d} \times \mathbb{R}^{d} \to \mathbb{R}$. For

two collections of points $\mathfrak{p} = (\mathfrak{p}_1, \ldots, \mathfrak{p}_m)$ and $\mathfrak{p}' = (\mathfrak{p}'_1, \ldots, \mathfrak{p}'_n)$, denote the Gram matrix

$$K(\mathfrak{p}, \mathfrak{p}') := \begin{bmatrix} k(\mathfrak{p}_1, \mathfrak{p}'_1) & \ldots & k(\mathfrak{p}_1, \mathfrak{p}'_n) \\ \vdots & \ddots & \vdots \\ k(\mathfrak{p}_m, \mathfrak{p}'_1) & \ldots & k(\mathfrak{p}_m, \mathfrak{p}'_n) \end{bmatrix}, \tag{4.4}$$

and mean vector

$$\mathfrak{m}(\mathfrak{p}) := \begin{bmatrix} \mathfrak{m}(\mathfrak{p}_1) & \ldots & \mathfrak{m}(\mathfrak{p}_m) \end{bmatrix}. \tag{4.5}$$

Then for pre-specified prior mean and covariance functions $\mathfrak{m}(\cdot)$ and $k(\cdot, \cdot)$, the posterior predictive distribution at test points $\mathfrak{p}_*$ given input-output training data $\mathcal{D} = (\mathfrak{p}, f)$ subject to zero-mean Gaussian noise with covariance C on the output observations f, is given by:

$$[f_*|\mathfrak{p}_*, \mathcal{D}] \sim \mathcal{N}\left(\mathfrak{m}(\mathfrak{p}_*) + K(\mathfrak{p}_*, \mathfrak{p})\mathbf{K}^{-1}(f - \mathfrak{m}(\mathfrak{p})), K(\mathfrak{p}_*, \mathfrak{p}_*) - K(\mathfrak{p}_*, \mathfrak{p})\mathbf{K}^{-1}K(\mathfrak{p}, \mathfrak{p}_*)\right), \tag{4.6}$$

where

$$\mathbf{K} := K(\mathfrak{p}, \mathfrak{p}) + C. \tag{4.7}$$

4.2 Active Learning Framework

The work in this section relates to a framework of active learning for LPV system identification via a local approach, which extends previous work described in Section 2.2.1.4 since it is applicable to multiple-input multiple-output (MIMO) systems with multivariate operating point. The framework also quantifies the uncertainty associated with the LPV model in terms of the variance of the model parameters.

4.2.1 GPR-LPV Model Estimation

The active learning procedure is explained as follows. We presume there to be an initial selection of m operating points $\mathfrak{p} = (\mathfrak{p}_1, \ldots, \mathfrak{p}_m)$ for identification. For each of these points, a time-series data set has been collected by running a local experiment and measuring

the (x_k, u_k) pairs. From this, we have then subsequently identified local linear models with matrices $\left(\widehat{A}_{p_1}, \widehat{B}_{p_1}\right), \ldots, \left(\widehat{A}_{p_m}, \widehat{B}_{p_m}\right)$.

Moreover, suppose our estimation method also provides uncertainty estimates for the identified parameters in the form of estimated standard deviation for the estimator (called the *standard errors* of the estimates). For an arbitrary element $\widehat{\zeta}_{p_i}$ of $\left(\widehat{A}_{p_i}, \widehat{B}_{p_i}\right)$ for any $i \in \{1, \ldots, m\}$, denote its standard error by $\operatorname{se}\left(\widehat{\zeta}_{p_i}\right)$.

Now to conduct active learning, we fit Gaussian processes to each of the elements of $A(p)$ and $B(p)$. That is, we represent these matrices as

$$A(p) = \begin{bmatrix} a_{1,1}(p) & \ldots & a_{1,n}(p) \\ \vdots & \ddots & \vdots \\ a_{n,1}(p) & \ldots & a_{n,n}(p) \end{bmatrix} \tag{4.8}$$

$$B(p) = \begin{bmatrix} b_{1,1}(p) & \ldots & b_{1,m}(p) \\ \vdots & \ddots & \vdots \\ b_{n,1}(p) & \ldots & b_{n,m}(p) \end{bmatrix}, \tag{4.9}$$

where each element $a_{1,1}(p), \ldots, b_{n,m}(p)$ is a GPR model over p as introduced in Section 4.1. From our initial identified models, we form $n^2 + mn$ training datasets $\mathcal{D}_{a_{1,1}}, \ldots, \mathcal{D}_{b_{n,m}}$ from the m experiments. Each $\mathcal{D}_\zeta$ for $\zeta \in \{a_{1,1}, \ldots, b_{n,m}\}$ consists of m observations with feature-label pairs $\left(p_i, \widehat{\zeta}_{p_i}\right)$ for $i = 1, \ldots, m$. Then at this stage, GPR is applied to each training dataset. Note that this induces a distribution over LPV models, and is the primary mechanism used to quantify uncertainty, which we do so in the following novel way. Under standard conditions (these being (4.1) is stable, w_k is i.i.d. and u_k is quasi-stationary), the least squares parameter estimates are asymptotically normal as the length of time for the local experiment tends to infinity [27]. Hence it is reasonable to use those standard errors as the Gaussian output-error covariances for each of the GPR:

$$C_\zeta = \operatorname{diag}\left\{\operatorname{se}\left(\widehat{\zeta}_{p_1}\right)^2, \ldots, \operatorname{se}\left(\widehat{\zeta}_{p_m}\right)^2\right\} \tag{4.10}$$

for each $\zeta \in \{\mathtt{a}_{1,1}, \ldots, \mathtt{b}_{\mathtt{n},\mathtt{m}}\}$. In traditional GPR, the covariance $\mathtt{C}$ is typically treated as a hyperparameter that can be optimised (usually simplified to be a scaled identity matrix). Here, we expressly use $\mathtt{C}$ to incorporate uncertainty information about the local parameters into the resulting GPR-LPV model. Qualitatively, where there is greater uncertainty about the local parameter estimates, this carries through to greater uncertainty in that surrounding region on the GPR-LPV model, as will be illustrated later on in Section 4.3.

4.2.2 Uncertainty Criterion

As a probabilistic model, the utility of the fitted GPR-LPV is that it can be used to quantify the uncertainty of the model with respect to an operating point of interest $\mathfrak{p}_*$. Introduce $\mathtt{V}_{\mathcal{M}}(\mathfrak{p}_*) : \mathfrak{P} \to \mathbb{R}$ as an arbitrary objective function which quantifies a measure of uncertainty at operating point $\mathfrak{p}_*$ for identified GPR-LPV model $\mathcal{M}$. Following the well-known MacKay approach, new query points can be selected where there is currently the most uncertainty [132]. The decision of which operating point to conduct the $(\mathtt{m}+1)^{\text{th}}$ experiment at is obtained by solving

$$\mathfrak{p}_{\mathtt{m}+1} = \underset{\mathfrak{p}_* \in \mathfrak{P}}{\operatorname{argmax}}\, \mathtt{V}_{\mathcal{M}}(\mathfrak{p}_*)\,. \tag{4.11}$$

We focus on $\mathtt{V}_{\mathcal{M}}(\mathfrak{p}_*)$ being the sum of GPR-LPV variances:

$$\mathtt{V}_{\mathcal{M}}(\mathfrak{p}_*) = \sum_{\zeta \in \{\mathtt{a}_{1,1},\ldots,\mathtt{b}_{\mathtt{n},\mathtt{m}}\}} \operatorname{Var}(\zeta | \mathcal{D}_\zeta, \mathfrak{p}_*)\,, \tag{4.12}$$

which is a natural choice, since it is equivalent to the trace of the posterior covariance for the parameter vector $(\mathtt{a}_{1,1}, \ldots, \mathtt{b}_{\mathtt{n},\mathtt{m}})$. In general, the problem (4.11) can have multiple local optima. If $\mathtt{d} = 2$, global optima may be validated visually due to compactness in Assumption 4.1. However beyond $\mathtt{d} = 2$, the problem of finding global optima begins to suffer from the curse of dimensionality. This is a similar problem encountered in Bayesian active learning, whereby the practice is to resort to global optimisation and heuristic search techniques to find an approximate solution [31].

Note that the type of uncertainty we are quantifying is the *epistemic uncertainty* (i.e. the model uncertainty), because the epistemic uncertainty can in principle be reduced by collecting more data. Quantifying the *aleatoric uncertainty* (which would involve estimating the covariance of the noise w_k) is not within the main scope of the active learning framework because the aleatoric uncertainty by definition cannot be reduced (without modifying the system itself).

4.2.3 Active Learning Algorithm

The active learning procedure is detailed by the pseudocode in Algorithm 4.1, with the following components.

- Time-series datasets $\mathbb{D}_1, \ldots, \mathbb{D}_{\mathtt{m}}$ from local experiments conducted at the corresponding operating points $\mathtt{p}_1, \ldots, \mathtt{p}_{\mathtt{m}}$. Note that the experiments need not be all of the same length.
- A method `ilm`() which identifies a local linear model (with standard errors) from local experiment data.
- A method `gpr`() which fits a GPR-LPV model to the local linear models, as described in Section 4.2.1.
- A method `uc`() which computes the uncertainty criterion for a GPR-LPV model at a supplied operating point.

Specific implementation details of the methods `ilm`(), `gpr`(), `uc`() are up to the practitioner's choice, which allows for flexible variations of the active learning algorithm. We are also formally required to impose a basic assumption on the time-series data, so that identifiability is maintained.

Assumption 4.5. *The input signals in each of $\mathbb{D}_1, \ldots, \mathbb{D}_{\mathtt{m}}$ are quasistationary and satisfy persistency of excitation [13].*

We are able to state the following two results for our active learning framework, which characterise the performance of Algorithm 4.1 in terms of the posterior variance on the GPR-LPV model.

Algorithm 4.1 Active Learning with GPR-LPV Models

1: **for** $i \in \{1, \ldots, \mathtt{m}\}$ **do**
2: Perform $\mathtt{ilm}\left(\mathbb{D}_i\right)$ to obtain $\left(\widehat{\mathsf{A}}_{\mathfrak{p}_i}, \widehat{\mathsf{B}}_{\mathfrak{p}_i}\right)$ and $\mathrm{se}\left(\widehat{\mathsf{A}}_{\mathfrak{p}_i}\right), \mathrm{se}\left(\widehat{\mathsf{B}}_{\mathfrak{p}_i}\right)$
3: **for** $\zeta \in \left\{\widehat{\mathsf{a}}_{1,1}, \ldots, \widehat{\mathsf{b}}_{\mathsf{n},\mathsf{m}}\right\}$ **do**
4: Construct $\mathcal{D}_\zeta$ from $\mathbb{D}_1, \ldots, \mathbb{D}_{\mathsf{m}}$
5: $\mathtt{C}_\zeta \leftarrow \mathrm{diag}\left\{\mathrm{se}\left(\zeta_{\mathfrak{p}_1}\right)^2, \ldots, \mathrm{se}\left(\zeta_{\mathfrak{p}_\mathtt{m}}\right)^2\right\}$
6: Perform $\mathtt{gpr}\left(\mathcal{D}_{\mathsf{a}_{1,1}}, \ldots, \mathcal{D}_{\mathsf{b}_{\mathsf{n},\mathsf{m}}}, \mathtt{C}_{\mathsf{a}_{1,1}}, \ldots, \mathtt{C}_{\mathsf{b}_{\mathsf{n},\mathsf{m}}}\right)$ to obtain GPR-LPV model $\mathcal{M}$
7: Solve (4.11) using $\mathtt{V}_{\mathcal{M}}\left(\mathfrak{p}_*\right) := \mathtt{uc}\left(\mathcal{M}, \mathfrak{p}_*\right)$
8: Return $\mathfrak{p}_{\mathtt{m}+1}$

Lemma 4.1. *Suppose the experiment at $\mathfrak{p}_{\mathtt{m}+1}$ is appended to the existing GPR-LPV which is identified from experiments at operating points $\boldsymbol{\mathfrak{p}}_\mathtt{m} = (\mathfrak{p}_1, \ldots, \mathfrak{p}_\mathtt{m})$. Then for each parameter $\zeta \in \{\mathsf{a}_{1,1}, \ldots, \mathsf{b}_{\mathsf{n},\mathsf{m}}\}$, the reduction $\mathcal{R}_{\zeta,\mathtt{m}+1}$ in posterior variance at $\mathfrak{p}_*$ is given by:*

$$\mathcal{R}_{\zeta,\mathtt{m}+1}\left(\mathfrak{p}_*\right) = \frac{\left(\mathtt{k}\left(\mathfrak{p}_*, \mathfrak{p}_{\mathtt{m}+1}\right) - \mathtt{k}_{\mathtt{m},\mathtt{m}+1}^\top \mathbf{K}_\zeta^{-1} \mathtt{K}\left(\boldsymbol{\mathfrak{p}}_\mathtt{m}, \mathfrak{p}_*\right)\right)^2}{\mathtt{k}\left(\mathfrak{p}_{\mathtt{m}+1}, \mathfrak{p}_{\mathtt{m}+1}\right) + \mathrm{se}\left(\widehat{\zeta}_{\mathfrak{p}_{\mathtt{m}+1}}\right)^2 - \mathtt{k}_{\mathtt{m},\mathtt{m}+1}^\top \mathbf{K}_\zeta^{-1} \mathtt{k}_{\mathtt{m},\mathtt{m}+1}}, \tag{4.13}$$

where

$$\mathtt{k}_{\mathtt{m},\mathtt{m}+1} := \mathtt{K}\left(\boldsymbol{\mathfrak{p}}_\mathtt{m}, \mathfrak{p}_{\mathtt{m}+1}\right), \tag{4.14}$$

$$\mathbf{K}_\zeta := \mathtt{K}\left(\boldsymbol{\mathfrak{p}}_\mathtt{m} \boldsymbol{\mathfrak{p}}_\mathtt{m}\right) + \mathtt{C}_\zeta \tag{4.15}$$

$$\mathtt{C}_\zeta := \mathrm{diag}\left\{\mathrm{se}\left(\widehat{\zeta}_{\mathfrak{p}_1}\right)^2, \ldots, \mathrm{se}\left(\widehat{\zeta}_{\mathfrak{p}_\mathtt{m}}\right)^2\right\}, \tag{4.16}$$

and $\widehat{\zeta}_{\mathfrak{p}_i}$ is the estimator for parameter $\zeta\left(\mathfrak{p}_i\right)$ via the data collected at experiment i.

Proof. The proof follows closely to the online supplement of [179], which relies on partitioned matrix inverse results. The main difference here is our inclusion of the standard errors (i.e. $\mathrm{se}\left(\widehat{\zeta}_{\mathfrak{p}_1}\right)$, $\mathrm{se}\left(\widehat{\zeta}_{\mathfrak{p}_2}\right)$, etc.) in the output observation covariances. □

Remark 4.1. *The reduction in posterior variance is non-negative since the denominator of* (4.13) *is the Schur complement of a positive definite matrix. Additionally, we can see that a smaller standard error $\mathrm{se}\left(\widehat{\zeta}_{\mathfrak{p}_{\mathtt{m}+1}}\right)$ results in a greater reduction in the posterior variance. When the term $\mathrm{se}\left(\widehat{\zeta}_{\mathfrak{p}_{\mathtt{m}+1}}\right)$ is computed using an asymptotic approximation [127, (10.3.8)], it behaves like*

$O\left(\mathsf{T}_{\mathtt{m}+1}^{-1/2}\right)$, *where* $\mathsf{T}_{\mathtt{m}+1}$ *is the length of the* $(\mathtt{m}+1)^{th}$ *experiment. This yields an intuitive conclusion that conducting a longer experiment will result in a greater reduction in posterior variance of the GPR-LPV.*

Next, we upper bound the posterior variance at the queried operating point in terms of the standard errors provided by `ilm()`.

Theorem 4.1. *Suppose the experiment at* $\mathfrak{p}_{\mathtt{m}+1}$ *is appended to the existing GPR-LPV which is identified from experiments at operating points* $\boldsymbol{\mathfrak{p}}_{\mathtt{m}}$*. Then for each parameter* $\zeta_* \in \{\mathsf{a}_{1,1}, \ldots, \mathsf{b}_{\mathsf{n},\mathsf{m}}\}$*, the posterior variance at* $\mathfrak{p}_* = \mathfrak{p}_{\mathtt{m}+1}$ *satisfies*

$$\operatorname{Var}\left(\zeta_* \middle| \mathcal{D}_\zeta, \mathfrak{p}_{\mathtt{m}+1}, \widehat{\zeta}_{\mathfrak{p}_{\mathtt{m}+1}}, \mathfrak{p}_* = \mathfrak{p}_{\mathtt{m}+1}\right) \leq \operatorname{se}\left(\widehat{\zeta}_{\mathfrak{p}_{\mathtt{m}+1}}\right)^2. \tag{4.17}$$

Proof. Begin from (4.13) and substitute $\mathfrak{p}_{\mathtt{m}+1}$ for $\mathfrak{p}_*$. Then from the structure for the posterior variance given in (A.5), we are able to show that the posterior variance takes the form:

$$\operatorname{Var}\left(\zeta_* \middle| \mathcal{D}_\zeta, \mathfrak{p}_{\mathtt{m}+1}, \widehat{\zeta}_{\mathfrak{p}_{\mathtt{m}+1}}, \mathfrak{p}_* = \mathfrak{p}_{\mathtt{m}+1}\right) = \mathfrak{a} - \frac{\mathfrak{a}^2}{\mathfrak{a}+\mathfrak{b}}, \tag{4.18}$$

where

$$\mathfrak{a} := \mathtt{k}\left(\mathfrak{p}_*, \mathfrak{p}_*\right) - \mathtt{K}\left(\mathfrak{p}_*, \boldsymbol{\mathfrak{p}}_*\right) \mathbf{K}_\zeta^{-1} \mathtt{K}\left(\boldsymbol{\mathfrak{p}}_*, \mathfrak{p}_*\right) \tag{4.19}$$

$$\mathfrak{b} := \operatorname{se}\left(\widehat{\zeta}_{\mathfrak{p}_{\mathtt{m}+1}}\right)^2. \tag{4.20}$$

Then it follows that

$$\operatorname{Var}\left(\zeta_* \middle| \mathcal{D}_\zeta, \mathfrak{p}_{\mathtt{m}+1}, \widehat{\zeta}_{\mathfrak{p}_{\mathtt{m}+1}}, \mathfrak{p}_* = \mathfrak{p}_{\mathtt{m}+1}\right) = \mathfrak{b} \cdot \frac{\mathfrak{a}}{\mathfrak{a}+\mathfrak{b}} \leq \mathfrak{b} \tag{4.21}$$

since $\mathfrak{a} \geq 0$ and $\mathfrak{b} \geq 0$. □

Remark 4.2. *If the uncertainty criterion is chosen as the sum of GPR-LPV variances as in* (4.12)*, then Theorem 4.1 implies that the total uncertainty at* $\mathfrak{p}_{\mathtt{m}+1}$ *post active learning will be upper bounded by the trace of the estimated covariance matrix for the local LPV model parameters. In*

this way, the active learning framework decouples the choice of operating point from the choice of input signals in the local experiment. Algorithm 4.1 can be seen as finding the operating point with greatest variance reduction potential, for which the resultant variance reduction can be controlled by the design of the local experiment with an A-optimality criterion. In general, this local design problem will depend on experimental constraints such as the allowable length of experimental time, as well as slew rate, saturation or power constraints on the input signals. This sub-problem is already well-addressed for linear systems in other literature, so we do not elaborate further here.

4.3 Active Learning for Diesel Engine Air-Path

We apply the active learning framework to the LPV system identification of a physical automotive diesel engine air-path, with exhaust gas recirculation (EGR) and variable geometry turbine (VGT). As mentioned in Section 2.2.1.2, a high-fidelity physics-based model for the diesel air-path has states for pressures and other physical signals of modelling relevance. This can lead to a relatively large number of states, for example eight in [198]. In [172], a reduced order model of four states was introduced to facilitate the online implementation of model predictive control.

4.3.1 Modelling

Following [172], the system is modelled using $\mathsf{n} = 4$ measured signals for the states:

$$\mathsf{x} = \begin{bmatrix} \mathrm{p_{im}} & \mathrm{p_{em}} & \mathrm{W_{comp}} & \mathrm{y_{EGR}} \end{bmatrix}^\top \tag{4.22}$$

and $\mathsf{m} = 3$ actuators:

$$\mathsf{u} = \begin{bmatrix} \mathrm{u_{thr}} & \mathrm{u_{EGR}} & \mathrm{u_{VGT}} \end{bmatrix}^\top, \tag{4.23}$$

where $\mathrm{p_{im}}$ is the intake manifold (boost) pressure, $\mathrm{p_{em}}$ is the exhaust manifold pressure, $\mathrm{W_{comp}}$ is the compressor mass flow rate and $\mathrm{y_{EGR}}$ is the EGR rate (which is the ratio of EGR mass flow rate to the sum of EGR and compressor mass flow rates, as defined in (2.12)). For the inputs, $\mathrm{u_{thr}}$ is the throttle valve, $\mathrm{u_{EGR}}$ is the EGR valve and $\mathrm{u_{VGT}}$ is the

VGT vane. A model is developed in the trimmed state and input:

$$\widetilde{\mathsf{x}} = \mathsf{x} - \mathsf{x}_{\mathrm{ss}}\left(\mathfrak{p}\right) \tag{4.24}$$

$$\widetilde{\mathsf{u}} = \mathsf{u} - \mathsf{u}_{\mathrm{ss}}\left(\mathfrak{p}\right), \tag{4.25}$$

where $\mathsf{x}_{\mathrm{ss}}\left(\mathfrak{p}\right)$ and $\mathsf{u}_{\mathrm{ss}}\left(\mathfrak{p}\right)$ are steady-state maps on the operating point $\mathfrak{p} = (\mathrm{N_e}, \mathrm{w_{fuel}})$, with $\mathrm{N_e}$ as the engine speed and $\mathrm{w_{fuel}}$ as the fueling rate. These maps have been previously obtained from a static calibration procedure as described in [164]. Thus, we can form an LPV model in the trimmed state and inputs with dynamics

$$\widetilde{\mathsf{x}}_{k+1} = \mathsf{A}\left(\mathfrak{p}\right)\widetilde{\mathsf{x}}_k + \mathsf{B}\left(\mathfrak{p}\right)\widetilde{\mathsf{u}}_k + \mathsf{w}_k. \tag{4.26}$$

The operating space $\mathfrak{P}$ is formed by box-constraints over $\mathfrak{p}$ (represented by high/low $\mathrm{N_e}$ and $\mathrm{w_{fuel}}$), and the outputs of interest for this system are $\mathsf{y} = \begin{bmatrix} \mathrm{p_{im}} & \mathrm{y_{EGR}} \end{bmatrix}^\top$. Normalisation of the states has been performed so that they are within the same order of magnitude.

4.3.2 Initial Training Data

An initial dataset was collected from 16 experiments at each of the operating points marked by the crosses in Figure 4.1. Each experiment constituted slightly over 6000 samples in duration, and was designed with a multisine input perturbation signal, due to slew rate considerations on the actuators.

For our choice of `ilm()` in the framework, the local linear estimates and their corresponding standard errors were identified using generalised least squares for VARX regression [127]. A GPR-LPV model is then fitted to these estimates. In our `gpr()` method, the covariance function we choose is the commonly-used squared exponential kernel:

$$\mathtt{k}\left(\mathfrak{p}, \mathfrak{p}'\right) = \mathtt{s}^2 \exp\left(-\frac{1}{2}\left(\mathfrak{p} - \mathfrak{p}'\right)^\top \Lambda^{-1}\left(\mathfrak{p} - \mathfrak{p}'\right)\right), \tag{4.27}$$

which is a justifiable choice by Assumption 4.2, since this kernel produces smooth sample

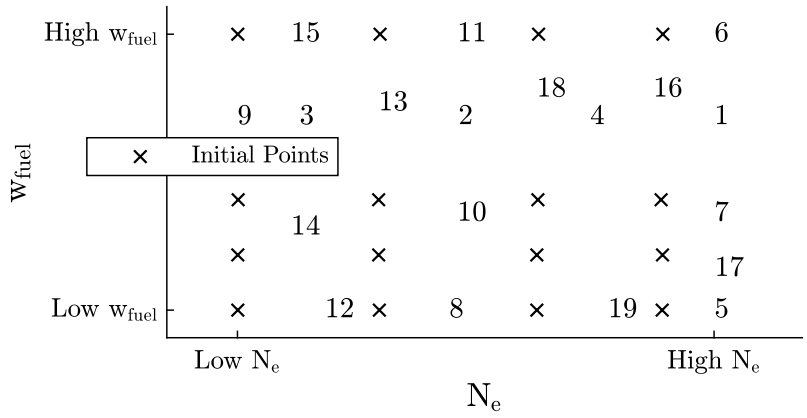


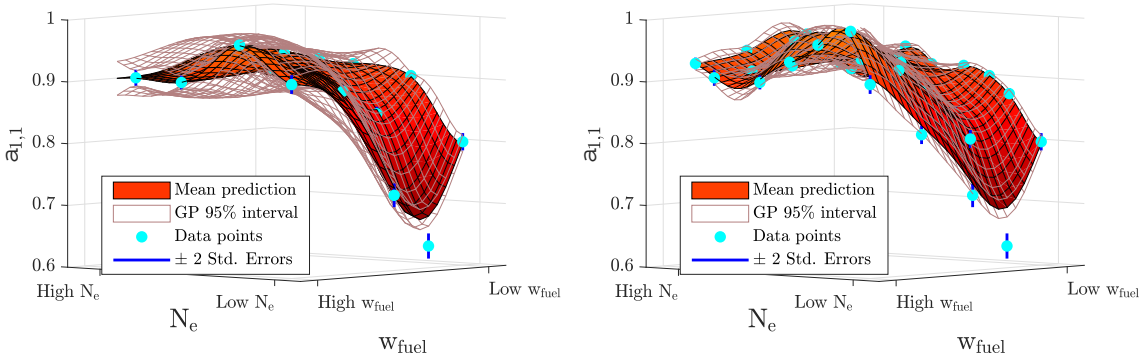
Figure 4.1: Operating points at which experiments were conducted. Points labelled with a number indicates the order in which the active learning experiment was performed beginning from the initial dataset.

paths of the posterior Gaussian processes. The matrix Λ is a diagonal matrix of length-scales, which we decide upon using domain knowledge, since the relative magnitudes of the units used in the operating point variables $\mathtt{p} = (\mathrm{N_e}, \mathrm{w_{fuel}})$ are understood. The hyperparameter $\mathtt{s}$ is chosen based on an empirical Bayes approach, where it is set to a factor of 2 of the maximum observed standard error for the respective parameter being fitted. As we suspect that $\mathsf{A}(\mathtt{p})$ has all eigenvalues inside the unit disk, we place a simple prior mean for $\mathsf{A}(\mathtt{p})$ which is a constant diagonal matrix with all elements less than one in magnitude. The prior mean for $\mathsf{B}(\mathtt{p})$ is taken as a constant matrix of zeros.

Figure 4.2a illustrates a GPR surface fitted to the $\mathtt{a}_{1,1}$ element from the initial training dataset, along with 95% credible intervals provided by the GPR and approximate 95% confidence intervals (2 standard errors) computed in the initial estimates.

4.3.3 Active Learning Results

We demonstrate the active learning framework for sequential selection of operating points. The uncertainty criterion (as given by the sum of GPR-LPV variances in (4.12)) for the GPR-LPV after the initial training dataset is displayed in Figure 4.3a. To extend Algorithm 4.1 for sequential operating point selection, we adopt a *greedy* approach, whereby the $(\mathtt{m}+1)^{\mathrm{st}}$ operating point is chosen at the point of maximum uncertainty after $\mathtt{m}$ ex-



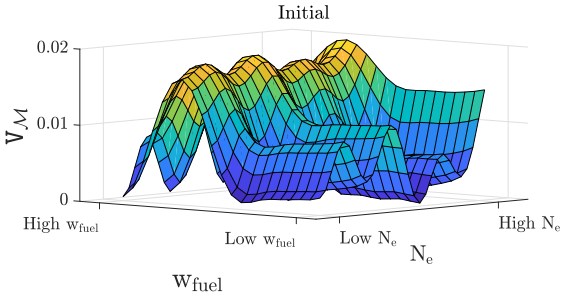
(a) Initial fitted GPR surface for the $a_{1,1}$ parameter. The GPR variance naturally increases the further away from the data points. Where the GPR surface lies above the particular data point; this is due to the effect of the prior regularisation. With a different selection of priors and also the hyperparameter Λ, a closer fit between the GPR estimate and the data point is possible.

(b) Final fitted GPR surface for the $a_{1,1}$ parameter after active learning. Compared to Figure 4.2a, the surface is more refined and the uncertainty intervals of the GP are narrower. Moreover by comparing the width of the GP 95% interval to the ± 2 standard errors interval, Theorem 4.1 is demonstrated.

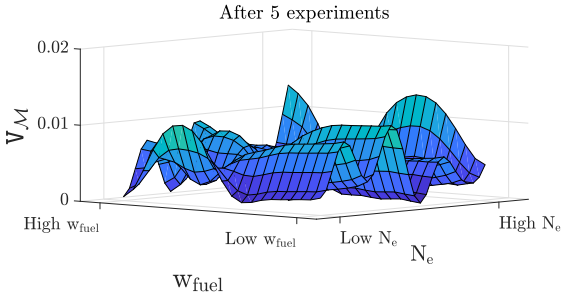
Figure 4.2: Comparison of fitted GPR surfaces for the $a_{1,1}$ parameter.

periments. We performed an additional 19 experiments using active learning with this greedy approach, to append on top of the initial training dataset for the GPR-LPV. The order and the locations at which these experiments were conducted are indicated in Figure 4.1. Figures 4.3a to 4.3d show the eventual reduction in variance over the operating space. The updated GPR surface for the $a_{1,1}$ element is presented in Figure 4.2b.

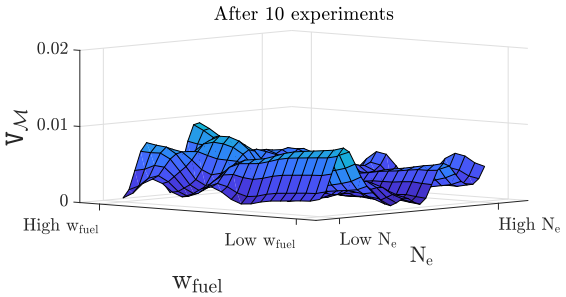
To assess the overall uncertainty of a GPR-LPV model $\mathcal{M}$ after a batch of experiments, we numerically evaluate the total integrated volume of the uncertainty criterion over the operating space, i.e. $\int_{\mathfrak{P}} \mathrm{v}_{\mathcal{M}}(\mathfrak{p})\, \mathrm{d}\mathfrak{p}$. Figure 4.4 plots the uncertainty volume as each subsequent experiment is added, and shows that using the active learning framework, most of the uncertainty can be reduced within the first few experiments.



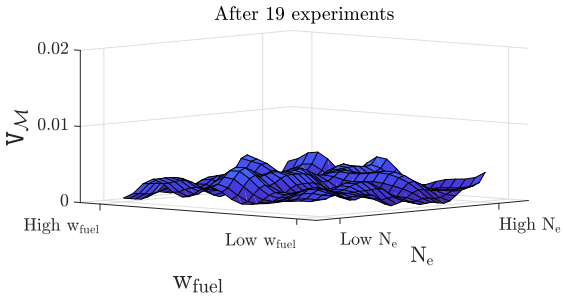
(a) Initial total uncertainty of GPR-LPV.



(b) Total uncertainty of GPR-LPV after 5 experiments. The total uncertainty is reduced compared to Figure 4.3a.



(c) Total uncertainty of GPR-LPV after 10 experiments. The total uncertainty is reduced compared to Figures 4.3a and 4.3b.



(d) Total uncertainty of GPR-LPV after 19 experiments. The total uncertainty is reduced compared to Figures 4.3a to 4.3c.

Figure 4.3: Total uncertainty surface as more experiments are added.

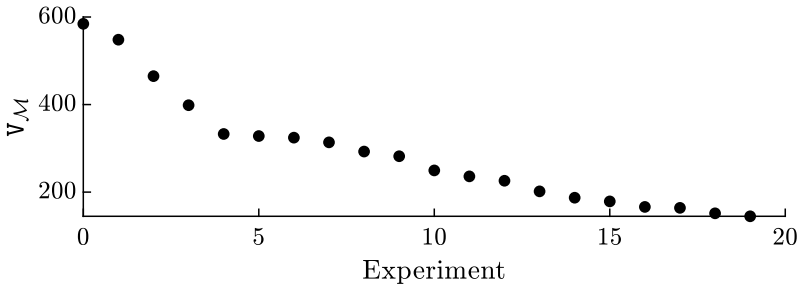


Figure 4.4: Decrease in uncertainty volume $\int_{\mathfrak{P}} \mathrm{v}_{\mathcal{M}}(\mathfrak{p})\,\mathrm{d}\mathfrak{p}$ via active learning.

4.4 Summary

In this chapter, we contributed an active learning framework for identifying LPV systems, and demonstrated the success of the approach via a reduction in total uncertainty of a GPR-LPV for a diesel-engine air-path. The ability of the GPR-LPV to quantify the model

uncertainty also provides an additional benefit, in analysing performance of controllers that were designed using the machine learning controller tuning framework.

Chapter 5

Ordinal Optimisation with Copula Models

In this chapter, we present results on the success probabilities for ordinal optimisation, with copula models on the joint distribution of performance values. After establishing some properties on the success probability for the general case in Section 5.2, we formally prove an analytic lower bound on the success probability under the Gaussian copula model in Section 5.3, and numerical experiments demonstrate that the lower bound yields a reasonable approximation to the actual success probability. Lastly, we showcase that the analytic lower bound can be used to invert the success probability, i.e. finding a sufficiently large sample size which yields a prescribed high success probability.

Contributions from this chapter also appear in the preprint publication [46].

5.1 Problem Setup

Recall in Section 2.4 that the original formulation of ordinal optimisation (OO) was where the set of possible controllers Θ was a finite set. Motivated by the need to extend the theory to cases where controller tuning variables belong to an uncountable set (e.g. quadratic-cost MPC), we develop a model for offline and online controller performances. To facilitate this, we work with the pair (Z, X) of random variables, which has an arbitrary continuous distribution with marginal cumulative distribution functions (CDFs) $F_Z(z)$, $F_X(x)$ respectively. For concreteness, suppose there exists some mechanism to randomly draw a candidate controller from uncountable Θ (e.g. the MLCTF from Section 2.1.2.4); then Z models the observed performance offline, while X models the tested performance if the same controller were evaluated online. As both performances involve the same controller, there will be some dependence between Z and X, which we model using a

bivariate copula defined as follows.

Definition 5.1 (Bivariate copula)**.** *A bivariate copula is a bivariate distribution, with both marginal distributions being the* $\mathrm{Uniform}\,(0,1)$ *distribution.*

Due to their ability to model dependence, and flexibility to describe wide classes of distributions, copula models see a host of applications in engineering, e.g. [208] for simulating communications channels, and finance, e.g. [40] for modelling returns. Through the *probability integral transform* (i.e. $F_Z(Z), F_X(X) \sim \mathrm{Uniform}\,(0,1)$ [11, Theorem 1]), every multivariate distribution can be represented with just its marginal distributions and a copula. Moreover, Sklar's theorem [109, Theorem 1.1.] asserts that if the distribution is continuous, then the choice of copula in this representation is unique. If continuous (Z,X) is represented with copula $\mathcal{C}_{Z,X}$, then we say that (Z,X) 'has' copula $\mathcal{C}_{Z,X}$ and write the joint copula CDF

$$\mathcal{C}_{Z,X}(z,x) := \Pr\left(F_Z(Z) \leq z, F_X(X) \leq x\right). \tag{5.1}$$

In addition, the conditional copula CDF of X given Z is denoted by

$$\mathcal{C}_{X|Z}(x|z) := \Pr\left(F_X(X) \leq x | F_Z(Z) = z\right). \tag{5.2}$$

Importing the setting from Section 2.4 and adapting it to the setup here thus far, suppose we have generated n i.i.d. candidate solutions and observed performance values $Z_1, \ldots, Z_n$. The horse race rule then stipulates to select the best m candidates out of the n. The actual performance of our selection depends on the associated X-values, which is quantified by the success probability formally defined as follows.

Definition 5.2 (Ordinal optimisation success probability)**.** *Consider n i.i.d. copies of (Z_i, X_i) drawn from the distribution of (Z,X), which is continuous. We observe $Z_1, \ldots, Z_n$, and order these observations from best to worst, denoted by $Z_{1:n} \leq \cdots \leq Z_{n:n}$. The best m are selected, given by $Z_{1:n} \leq \cdots \leq Z_{m:n}$, with respective X-values denoted as $X_{\langle 1\rangle}, \ldots, X_{\langle m\rangle}$, which are initially unobserved. More explicitly, we have selected the pairs $\left(Z_{1:n}, X_{\langle 1\rangle}\right), \ldots, \left(Z_{m:n}, X_{\langle m\rangle}\right)$.*

The success probability is defined as

$$p_{\text{success}}(n, m, \alpha) := \Pr\left(\bigcup_{i=1}^{m} \left\{X_{\langle i \rangle} \leq x_\alpha^*\right\}\right) \tag{5.3}$$

$$= \Pr\left(\min_{i \in \{1,\ldots,m\}} X_{\langle i \rangle} \leq x_\alpha^*\right), \tag{5.4}$$

where x_α^* with $\alpha \in (0, 1]$ is the 100α percentile of the distribution of X, i.e. $\Pr(X \leq x_\alpha^*) = \alpha$.

The value of α can be used to control the level of goal-softening (i.e. degree of sub-optimality). A condition that we might desire on (Z, X) is some notion of positive dependence, i.e. roughly speaking, low (high) Z is predictive of low (high) X. In offline controller tuning for example, this intuitively means that well-performing controllers in offline simulation are predicted to also perform well when tested online. One such formal notion of positive dependence is called stochastically increasing (SI) positive dependence, which we state as a regularity assumption.

Assumption 5.1 (Stochastically increasing positive dependence)**.** *The random variable X is stochastically increasing in Z [109, §2.8.2]. That is,*

$$\Pr(X > x | Z = z) \leq \Pr\left(X > x \middle| Z = z'\right) \tag{5.5}$$

for all z, z' in the support of Z such that $z \leq z'$.

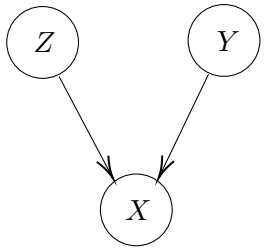
Regularity Assumption 5.1 may be satisfied, for instance, under an additive noise causal representation. Consider the following causal mechanism (depicted in Figure 5.1a) for generating (Z, X) by first generating Z, and then generating X given $Z = z$:

$$X = z + Y, \tag{5.6}$$

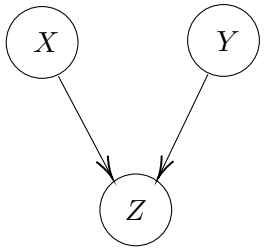
where Y is independent of Z. An alternative additive causal representation is the reverse (depicted in Figure 5.1b); first X is generated, and then Z is generated given $X = x$ by:

$$Z = x + Y, \tag{5.7}$$

where Y is independent of X.



(a) Causal additive noise representation for X.



(b) Causal additive noise representation for Z.

Figure 5.1: Causal graphs [141, §3.2.2] for additive noise representations.

Proposition 5.1 (Additive noise implies SI)**.** *If either of the following conditions hold:*

- (Z, X) *is generated via the causal representation* (5.6)*, or*
- (Z, X) *is generated via the causal representation* (5.7)*, and moreover the copula of* (Z, X) *is exchangeable (i.e. its distribution function is permutation symmetric),*

then X is stochastically increasing in Z.

Proof. Contained in Appendix B.3.1. □

Note that many classes of parametric copulae (with one or two parameters) are exchangeable [109, §2.15], as are all members of the Archimedean family of copulae [89], so regularity Assumption 5.1 is satisfied for a variety of models with additive noise. A stronger notion of positive dependence is *positive likelihood ratio dependence*, which implies SI positive dependence [145].

Definition 5.3 (Positive likelihood ratio dependence)**.** *The random variables Z and X are said to be positive likelihood ratio dependent [145, Definition 5.2.18] if their joint density $f_{ZX}(z, x)$ satisfies*

$$f_{ZX}(z, x)\, f_{ZX}\left(z', x'\right) \geq f_{ZX}\left(z, x'\right) f_{ZX}\left(z', x\right) \tag{5.8}$$

for all $z, x, z', x' \in \mathbb{R}$ such that $z \leq z'$ and $x \leq x'$.

5.2 Properties of OO Success Probability

In this section, we state several properties of the success probability when (Z, X) has a general copula model. The first property gives an expression to compute the success probability by evaluating an m-dimensional integral.

Theorem 5.1 (Expression for OO success probability)**.** *We have*

$$p_{\text{success}}(n, m, \alpha) = \frac{n!}{(n-m)!} \int_0^1 \int_0^{z_m} \cdots \int_0^{z_2} \left[1 - \prod_{j=1}^{m} \left(1 - \mathcal{C}_{X|Z}(\alpha|z_j)\right) \right] \times (1 - z_m)^{n-m} \, \mathrm{d}z_1 \mathrm{d}z_2 \ldots \mathrm{d}z_m. \quad (5.9)$$

Proof. Contained in Appendix B.3.2. □

Theorem 5.1 reveals that when calculating the success probability given n, m and α, only the copula of (Z, X) matters. Or alternatively, we can without loss of generality assume that (Z, X) is a copula distribution. Therefore, we see here why "ordinal" is a fitting qualifier - the success probability will be invariant to univariate monotonic (increasing) transformations of Z and X.

Under the SI positive dependence regularity condition, we can also confirm that selecting the best m observed is indeed the optimal choice.

Theorem 5.2 (Optimality of horse race selection)**.** *Under regularity Assumption 5.1, the selection of the first m order statistics maximises the success probability in Definition 5.2, compared to any other selection of size m from the sample.*

Proof. Contained in Appendix B.3.3. □

The success probability can also be shown to be non-decreasing in each of its arguments.

Theorem 5.3 (Monotonicity of OO success probability)**.** *The success probability p_{success} from Definition 5.2 satisfies the following monotonicity properties.*

(a) (Monotonicity in n) Under Assumption 5.1, for any $n \in \mathbb{N}$ and fixed $\bar{m} \leq n$ and fixed $\bar{\alpha} \in (0, 1]$, we have

$$p_{\text{success}}(n, \bar{m}, \bar{\alpha}) \leq p_{\text{success}}(n+1, \bar{m}, \bar{\alpha}). \tag{5.10}$$

(b) (Monotonicity in m) For fixed $\bar{n} \in \mathbb{N}$, fixed $\bar{\alpha} \in (0, 1]$ and any $m < n$, we have

$$p_{\text{success}}(\bar{n}, m, \bar{\alpha}) \leq p_{\text{success}}(\bar{n}, m+1, \bar{\alpha}). \tag{5.11}$$

(c) (Monotonicity in α) For fixed $\bar{n} \in \mathbb{N}$, fixed $\bar{m} \leq \bar{n}$ and any $\alpha, \alpha' \in (0, 1]$ such that $\alpha \leq \alpha'$, we have

$$p_{\text{success}}(\bar{n}, \bar{m}, \alpha) \leq p_{\text{success}}(\bar{n}, \bar{m}, \alpha'). \tag{5.12}$$

Proof. Contained in Appendix B.3.4. □

For any copula satisfying the SI positive dependence condition, we also have the following general upper and lower bounds.

Theorem 5.4 (General bounds for OO success probability)**.** *Under regularity Assumption 5.1, the success probability p_{success} from Definition 5.2 satisfies*

$$1 - (1-\alpha)^m \leq p_{\text{success}}(n, m, \alpha) \leq 1 - (1-\alpha)^n. \tag{5.13}$$

Proof. Contained in Appendix B.3.5. □

These bounds are tight, as some of the following limiting cases show.

Theorem 5.5 (Limiting forms of OO success probability)**.** *Under regularity Assumption 5.1, the success probability p_{success} from Definition 5.2 satisfies the following.*

(a) When $m = n$:

$$p_{\text{success}}(n, n, \alpha) = 1 - (1 - \alpha)^n . \tag{5.14}$$

(b) When $\alpha = 1$:

$$p_{\text{success}}(n, m, 1) = 1. \tag{5.15}$$

(c) When $\alpha \to 0^+$:

$$\lim_{\alpha \to 0^+} p_{\text{success}}(n, m, \alpha) = 0. \tag{5.16}$$

(d) If Z and X are comonotonic (i.e. perfect positive dependence, such that $F_Z(Z) = F_X(X)$ for every realisation of (Z, X)), then

$$p_{\text{success}}(n, m, \alpha) = 1 - (1 - \alpha)^n . \tag{5.17}$$

(e) If Z and X are independent, then

$$p_{\text{success}}(n, m, \alpha) = 1 - (1 - \alpha)^m . \tag{5.18}$$

(f) When $m \to \infty$ and $n \to \infty$ in any way such that $m \leq n$:

$$\lim_{m,n \to \infty} p_{\text{success}}(n, m, \alpha) = 1. \tag{5.19}$$

(g) When m is finite and $n \to \infty$:

$$\lim_{n \to \infty} p_{\text{success}}(n, m, \alpha) = 1 - \left(1 - \mathcal{C}_{X|Z}(\alpha|0)\right)^m . \tag{5.20}$$

Proof. Contained in Appendix B.3.6. □

Since n represents the number of offline simulations while m represents the number of online tests (for which the latter is more restricted), we are primarily interested in the regime of large n and small fixed m. Moreover, a useful result is to know for what sufficiently large n is $p_{\text{success}}(n, m, \alpha) \geq 1 - \delta$ for a given δ. Sadly, the property in Theorem 5.5(g) indicates that in general, for finite m we have $\lim_{n\to\infty} p_{\text{success}}(n, m, \alpha) < 1$, whenever $\mathcal{C}_{X|Z}(\alpha|0) < 1$. For example, the bivariate Frank copula [109, §4.5.1] with parameter $\varpi > 0$ has conditional CDF

$$\mathcal{C}_{X|Z}^{\text{Frank}}(\alpha|0;\varpi) = \frac{1 - e^{-\varpi\alpha}}{1 - e^{-\varpi}} < 1. \tag{5.21}$$

This means that in general we can only attain a high success probability with a combination of sufficiently large n and m. However, there do exist classes of copulae where $\lim_{n\to\infty} p_{\text{success}}(n, m, \alpha) = 1$ for all finite $m \geq 1$. In these cases, it is possible to invert the success probability for any m, i.e. for a prescribed high probability $1 - \delta$ with any $\delta \in (0, 1]$, to find an n^* such that $p_{\text{success}}(n^*, m, \alpha) \geq 1 - \delta$. In the next section, we study one such class of copula which satisfies this property.

5.3 Gaussian Copula Ordinal Optimisation

The results in this section specialise to the case when (Z, X) has a Gaussian copula, defined as follows.

Definition 5.4 (Bivariate Gaussian copula)**.** *Let (Z, X) be a bivariate standard Gaussian with correlation $\rho \in [-1, 1]$, i.e.*

$$\begin{bmatrix} Z \\ X \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix} \right). \tag{5.22}$$

Then the Gaussian copula with correlation ρ is the distribution of $(\Phi(Z), \Phi(X))$.

In the literature, the class of multivariate distributions with Gaussian copulae are known as *non-paranormal* distributions [124], and alternatively as *meta-Gaussian* distributions [178]. A bivariate Gaussian copula is entirely specified by the correlation parameter ρ, which also neatly summarises the dependence within the distribution. Thus, we denote the **OO success probability with a Gaussian copula** as $p^{\mathcal{N}}_{\text{success}}(n, m, \alpha, \rho)$, with analogous definition to (5.3), except the dependence on ρ is explicitly indicated. The bivariate Gaussian copula is known to satisfy the stronger condition of positive likelihood ratio dependence (Definition 5.3) for $\rho > 0$ [109, §4.3.1], which implies regularity Assumption 5.1 is also satisfied. Thus all the properties in Section 5.2 hold for $p^{\mathcal{N}}_{\text{success}}$. It is also known that the boundary CDF for the bivariate Gausssian is $\mathcal{C}^{\mathcal{N}}_{X|Z}(\cdot|0;\rho) = 1$ for all $\rho > 0$, so by Theorem 5.5(g), we immediately have

$$\lim_{n\to\infty} p^{\mathcal{N}}_{\text{success}}(n, m, \alpha, \rho) = 1 \tag{5.23}$$

when $\rho > 0$.

5.3.1 Additive Noise Representation

Suppose observations Z are generated by additive noise to X as in (5.7), analogous to the setup in Section 2.4. Specifically, let

$$Z = X + Y, \tag{5.24}$$

where $X \sim \mathcal{N}(0, 1)$ and $Y \sim \mathcal{N}\left(0, \xi^2\right)$. The variable ξ^2 may be interpreted here as the *noise-to-signal ratio*. Note that (Z, X) is then bivariate Gaussian with correlation $\rho = \left(1 + \xi^2\right)^{-1/2}$, and naturally, a bivariate Gaussian has a bivariate Gaussian copula. Hence this yields the success probability $p^{\mathcal{N}}_{\text{success}}\left(n, m, \alpha, \left(1 + \xi^2\right)^{-1/2}\right)$ in terms of the noise-to-signal ratio. This additive noise representation can also be used in the proof to show an additional monotonicity property in ρ, as follows.

Theorem 5.6 (Monotonicity in ρ)**.** *For fixed $\bar{n} \in \mathbb{N}$, fixed $\bar{m} \leq n$, fixed $\alpha \in (0, 1]$ and any*

$\rho, \rho' \in (0,1]$ *such that* $\rho \leq \rho'$*, we have*

$$p^{\mathcal{N}}_{\text{success}}\left(\bar{n}, \bar{m}, \bar{\alpha}, \rho\right) \leq p^{\mathcal{N}}_{\text{success}}\left(\bar{n}, \bar{m}, \bar{\alpha}, \rho'\right). \tag{5.25}$$

Proof. Contained in Appendix B.3.7. □

The intuitive takeaway from this result is that a 'good' model for (Z, X) is one with high ρ, i.e. strong positive dependence.

5.3.2 Approximation Formula

Motivated by the computational intractability in computing p_{success} by evaluating an m-dimensional integral using (5.9) for large n, we develop a more computationally tractable approximation for $p^{\mathcal{N}}_{\text{success}}$ under the Gaussian copula model. From the additive noise representation, the success probability may be written as

$$p^{\mathcal{N}}_{\text{success}}\left(n, m, \alpha, \rho\right) = \Pr\left(\min\left\{X_{\langle 1\rangle}, \ldots, X_{\langle m\rangle}\right\} \leq \Phi^{-1}\left(\alpha\right)\right). \tag{5.26}$$

Note the symmetry properties of the zero-mean Gaussian and its order statistics, i.e. we can relate the lower and upper extreme order statistics by

$$\left(Z_{1:n}, \ldots, Z_{m:n}\right) \underset{\text{st}}{=} \left(-Z_{n:n}, \ldots, -Z_{(n-m+1):n}\right). \tag{5.27}$$

Then it follows that (5.26) has a symmetry property, in the sense

$$\Pr\left(\min\left\{X_{\langle 1\rangle}, \ldots, X_{\langle m\rangle}\right\} \leq \Phi^{-1}\left(\alpha\right)\right) = \Pr\left(\max\left\{X_{\langle n-m+1\rangle}, \ldots, X_{\langle n\rangle}\right\} \geq \Phi^{-1}\left(1-\alpha\right)\right). \tag{5.28}$$

Hence, we develop the approximation of $p^{\mathcal{N}}_{\text{success}}$ using the upper variables $X_{\langle n-m+1\rangle}, \ldots, X_{\langle n\rangle}$ instead of the lower variables $X_{\langle 1\rangle}, \ldots, X_{\langle m\rangle}$ (analogous to considering a maximisation problem rather than a minimisation problem). As will be clear, the resulting approximation involves the usual multivariate Gaussian CDF rather than the complementary

CDF, for which software implementations of the former are more prevalent. Our approach is to approximate the joint extreme order statistics of the standard Gaussian Z from representation (5.22) with a multivariate Gaussian. The distribution of the random vector $\mathbf{Z}' = \left(Z_{(n-m+1):n}, \ldots, Z_{n:n}\right)$ is chosen to be approximated with the distribution of

$$\widehat{\mathbf{Z}}' \sim \mathcal{N}\left(\boldsymbol{\mu}_{\widehat{\mathbf{Z}}'}, C_{\widehat{\mathbf{Z}}'}\right), \tag{5.29}$$

with mean vector

$$\boldsymbol{\mu}_{\widehat{\mathbf{Z}}'} = \begin{bmatrix} \Phi^{-1}(p_1) & \ldots & \Phi^{-1}(p_m) \end{bmatrix}^\top \tag{5.30}$$

and covariance structure

$$\left[C_{\widehat{\mathbf{Z}}'}\right]_{ij} = \frac{p_i(1-p_j)}{n\phi\left(\Phi^{-1}(p_i)\right)\phi\left(\Phi^{-1}(p_j)\right)}, \quad i \leq j, \tag{5.31}$$

where $p_1 = \frac{n-m}{n}, \ldots, p_m = \frac{n-1}{n}$. This approximation is justified by the asymptotic normality of joint central order statistics [12, Theorem 8.5.2], with which we subsequently approximate the extreme order statistics. Note that the practice of approximating order statistics using asymptotic theory is not uncommon [157]. From (5.22), the conditional distribution of $\mathbf{X}' = \left(X_{\langle n-m+1\rangle}, \ldots, X_{\langle n\rangle}\right)$ given $\mathbf{Z}'$ can be computed using well-known Gaussian conditioning formulae [155, Equation (A.6)] to be the Gaussian

$$\left[\mathbf{X}' \middle| \mathbf{Z}' = \mathbf{z}\right] \sim \mathcal{N}\left(\rho\mathbf{z}, \left(1-\rho^2\right) I\right). \tag{5.32}$$

With a Gaussian approximation for $\mathbf{Z}'$ and a Gaussian form for $\mathbf{X}'$ conditioned on $\mathbf{Z}'$, one can analytically marginalise out $\mathbf{Z}'$ with well-known formulae [159, Equation (1.11)] and obtain a Gaussian approximation for $\mathbf{X}'$. Hence $\mathbf{X}'$ is approximated with a multivariate Gaussian vector $\widehat{\mathbf{X}}' \sim \mathcal{N}\left(\boldsymbol{\mu}_{\widehat{\mathbf{X}}'}, C_{\widehat{\mathbf{X}}'}\right)$, where

$$\boldsymbol{\mu}_{\widehat{\mathbf{X}}'} = \rho\boldsymbol{\mu}_{\widehat{\mathbf{Z}}'} \tag{5.33}$$

$$C_{\widehat{\mathbf{X}}'} = \rho^2 C_{\widehat{\mathbf{Z}}'} + \left(1-\rho^2\right) I. \tag{5.34}$$

We denote the CDF of this multivariate Gaussian by $\mathbf{\Phi}_{\boldsymbol{\mu}_{\widehat{\mathbf{X}}'}, C_{\widehat{\mathbf{X}}'}}(\cdot)$. The success probability is given by

$$p^{\mathcal{N}}_{\text{success}}(n, m, \alpha, \rho) = \Pr\left(\max\left\{X_{\langle n-m+1\rangle}, \ldots, X_{\langle n\rangle}\right\} \geq \Phi^{-1}(1-\alpha)\right) \tag{5.35}$$

$$= 1 - \Pr\left(\mathbf{X}' \leq \Phi^{-1}(1-\alpha)\,\mathbf{1}\right). \tag{5.36}$$

Using the Gaussian approximation $\widehat{\mathbf{X}}' \sim \mathcal{N}\left(\boldsymbol{\mu}_{\widehat{\mathbf{X}}'}, C_{\widehat{\mathbf{X}}'}\right)$ for $\mathbf{X}'$, the right-hand side of the above equation is approximated as

$$\widehat{p}^{\mathcal{N}}_{\text{success}}(n, m, \alpha, \rho) := 1 - \Pr\left(\widehat{\mathbf{X}}' \leq \Phi^{-1}(1-\alpha)\,\mathbf{1}\right) \tag{5.37}$$

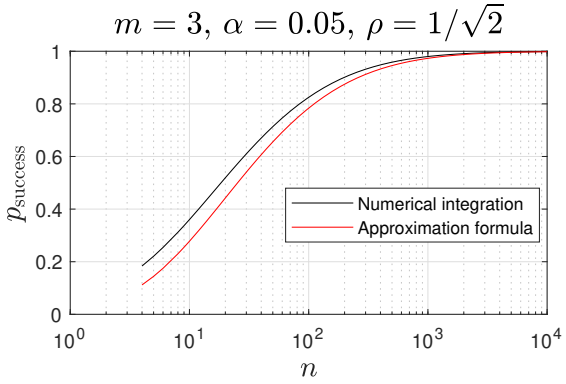
$$= 1 - \mathbf{\Phi}_{\boldsymbol{\mu}_{\widehat{\mathbf{X}}'}, C_{\widehat{\mathbf{X}}'}}\left(\Phi^{-1}(1-\alpha)\,\mathbf{1}\right). \tag{5.38}$$

Computing this approximation formula is of time complexity $O\left(m^2\right)$. This is due to the construction of the $m \times m$ covariance matrix $C_{\widehat{\mathbf{Z}}'}$, and marginalisation of Gaussians in (5.33) and (5.34) will take only $O\left(m^2\right)$ operations. Then, evaluation of the multivariate Gaussian CDF using the algorithms from [77] are at most $O\left(m^2\right)$. We proceed to illustrate that this formula yields a reasonable approximation.

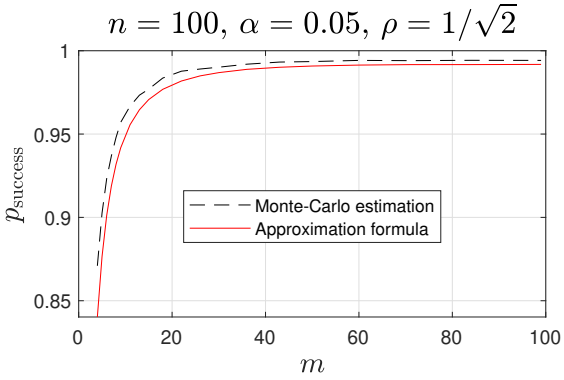
In Figures 5.2a-5.2d, we depict primarily the approximation formula, when (5.30) and (5.31) are used in (5.38). In each figure, the nominal values $n = 100$, $m = 3$, $\alpha = 0.05$, $\rho = 1/\sqrt{2}$ are used as a baseline, and one variable is varied at a time, keeping the others fixed. This is compared against a numerical integration of the success probability (and alternatively in the case m is varied, a Monte-Carlo estimate with 2×10^4 simulation replications at each point). The approximation formula is also exhibited to be reasonably close to the true probability, whilst still being conservative (in the sense of being an underestimate) of the success probability.

5.3.3 Analytic Lower Bound for Success Probability

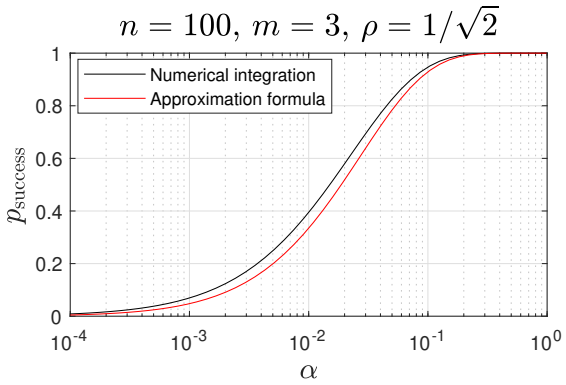
Motivated by our approximation formula, we rigorously study Gaussian approximations of the form (5.29), which allow us to analytically marginalise when approximating the



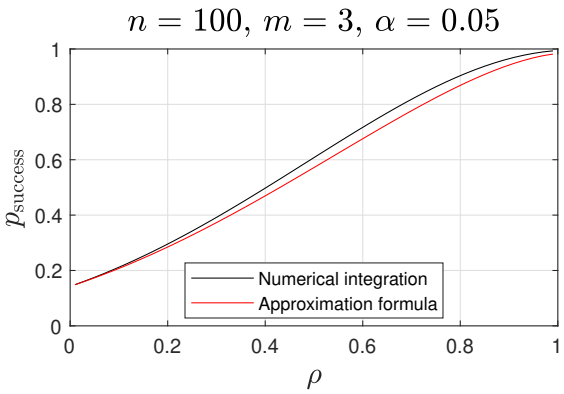
(a) Comparison of $p^{\mathcal{N}}_{\text{success}}$ when n is varied. In particular, this figure demonstrates monotonicity in n from Theorem 5.3(a) and convergence in n from (5.23).



(b) Comparison of $p^{\mathcal{N}}_{\text{success}}$ when m is varied. In particular, this figure demonstrates: 1) monotonicity in m from Theorem 5.3(b), and 2) tightness of the bounds when $m = n$ from Theorem 5.5(a).



(c) Comparison of $p^{\mathcal{N}}_{\text{success}}$ when α is varied. In particular, this figure demonstrates tightness of the bounds as $\alpha = 1$ and as $\alpha \to 0$ from Theorem 5.5(c). The success probability is increasing in α, which is intuitive (by goal softening, we can improve our odds of success) and demonstrates Theorem 5.3(c).



(d) Comparison of $p^{\mathcal{N}}_{\text{success}}$ when ρ is varied. The success probability is increasing in ρ, or alternatively decreasing in ξ, which is intuitive (there is a higher price to be paid for more noise) and demonstrates Theorem 5.6.

Figure 5.2: Numerical results for the approximation formula.

success probability. The following lemma provides a sufficient condition for such an approximation to yield a lower bound on $p^{\mathcal{N}}_{\text{success}}$. The result uses the following notion of stochastic dominance.

Definition 5.5 (Multivariate stochastic dominance [169]). *We say that random vector* $\mathbf{X}_1 \in \mathbb{R}^n$ *is stochastically dominated by random vector* $\mathbf{X}_2 \in \mathbb{R}^n$ *and denote* $\mathbf{X}_1 \underset{\text{st}}{\preceq} \mathbf{X}_2$ *if and only if*

$\mathbb{E}\left[u\left(\mathbf{X}_1\right)\right] \leq \mathbb{E}\left[u\left(\mathbf{X}_2\right)\right]$ *for all weakly increasing (i.e. non-decreasing) functions* $u : \mathbb{R}^n \to \mathbb{R}$. *Equivalently, we can say* $\mathbf{X}_1 \underset{\text{st}}{\preceq} \mathbf{X}_2$ *if and only if* $\Pr\left(\mathbf{X}_1 \in \mathcal{U}\right) \leq \Pr\left(\mathbf{X}_2 \in \mathcal{U}\right)$ *for all upper sets* $\mathcal{U}$ *(an upper set may be defined as a set which satisfies* $\mathbf{x}_2 \in \mathcal{U}$ *for all* $\mathbf{x}_2 \geq \mathbf{x}_1 \in \mathcal{U}$*).*

Lemma 5.1 (Sufficient condition for lower bound)**.** *Let* $\mathbf{Z}$ *be a random vector for the first* m *order statistics of* $Z_1, \ldots, Z_n$. *Let* $\widehat{\mathbf{Z}}$ *be an arbitrary Gaussian approximation to* $\mathbf{Z}$*, of the form* (5.29). *Suppose we have that* $\mathbf{Z} \underset{\text{st}}{\preceq} \widehat{\mathbf{Z}}$. *Then the approximation computed in the same way as* (5.38)*, using* $\widehat{\mathbf{Z}}'$ *constructed symmetrically to* $\widehat{\mathbf{Z}}$ *as per* (5.27)*, yields a lower bound*

$$\widehat{p}^{\mathcal{N}}_{\text{success}}\left(n, m, \alpha, \rho\right) \leq p^{\mathcal{N}}_{\text{success}}\left(n, m, \alpha, \rho\right). \tag{5.39}$$

Proof. Contained in Appendix B.3.8. □

It is an open problem whether the proposed approximation formula in Section 5.3.2 with (5.30) and (5.31) satisfies the stochastic dominance condition of Lemma 5.1. This is owing to the CDF of $\mathbf{Z}$ taking on a complicated form (see (B.3)), while the CDF of $\widehat{\mathbf{Z}}$ is a multidimensional integral of a multivariate Gaussian density which has no analytical form. This makes it challenging to directly verify stochastic dominance. However, in the case of $m = 1$, it is possible to construct a Gaussian approximation for the first order statistic that is stochastically dominating. The following result presents a class of such approximations.

Lemma 5.2 (Stochastic dominance of Gaussian first order statistic)**.** *Let* $Z_{1:n}$ *denote the first order statistic of an i.i.d. standard Gaussian sample of size* n. *For any* $\omega \in \left(0, \frac{\pi}{2}\right)$*, let*

$$\mathfrak{c}_1 = \frac{1}{2} - \frac{\omega}{\pi} \tag{5.40}$$

$$\mathfrak{c}_2 = \frac{\cot \omega}{\pi - 2\omega}. \tag{5.41}$$

Consider $\widehat{Z}_{1:n} \sim \mathcal{N}\left(\mu_n, \sigma_n^2\right)$ *where*

$$\mu_n = -\sqrt{\frac{\log\left(n\mathfrak{c}_1\right)}{\mathfrak{c}_2}} \tag{5.42}$$

$$\sigma_n^2 = \frac{-\log\log 2}{2\mathfrak{c}_2\left(\log\left(n\mathfrak{c}_1\right) - \log\log 2\right)}. \tag{5.43}$$

Then there exists some integer $n^*(\omega)$ *such that for all* $n \geq n^*(\omega)$*, we have* $Z_{1:n} \underset{\text{st}}{\preceq} \widehat{Z}_{1:n}$.

Proof. Contained in Appendix B.3.9. □

Combining Lemmas 5.1-5.2, we arrive at the following analytic lower bound on the success probability under the Gaussian copula model.

Theorem 5.7 (Success probability lower bound)**.** *Given* $\omega \in \left(0, \frac{\pi}{2}\right)$*, there exists some integer* $n^*(\omega)$ *such that for all* $n \geq n^*(\omega)$*,* $m \in [1, n]$*,* $\rho \in (0, 1]$*,* $\alpha \in (0, 1]$

$$p^{\mathcal{N}}_{\text{success}}(n, m, \alpha, \rho) \geq p^{\mathcal{N}}_{\text{success}}(n, 1, \alpha, \rho) \geq \Phi\left(\frac{\Phi^{-1}(\alpha) - \rho\mu_n(\omega)}{\sqrt{1 - \rho^2 + \rho^2\left[\sigma_n(\omega)\right]^2}}\right) \tag{5.44}$$

$$=: \underline{p}^{\mathcal{N}}_{\text{success},\omega}(n, 1, \alpha, \rho). \tag{5.45}$$

Morever, given any $n \in \mathbb{N}$*,* $m \in [1, n]$*,* $\rho \in (0, 1]$*,* $\alpha \in (0, 1]$*:*

$$p^{\mathcal{N}}_{\text{success}}(n, m, \alpha, \rho) \geq p^{\mathcal{N}}_{\text{success}}(n, 1, \alpha, \rho) \geq \sup_{\omega \in \Omega_n} \Phi\left(\frac{\Phi^{-1}(\alpha) - \rho\mu_n(\omega)}{\sqrt{1 - \rho^2 + \rho^2\left[\sigma_n(\omega)\right]^2}}\right) \tag{5.46}$$

$$=: \underline{p}^{\mathcal{N}}_{\text{success}}(n, 1, \alpha, \rho), \tag{5.47}$$

where $\Omega_n \subset \left(0, \frac{\pi}{2}\right)$ *is the set of all* ω *such that* $n \geq n^*(\omega)$*, while* $\mu_n(\omega)$*,* $\left[\sigma_n(\omega)\right]^2$ *are* (5.42)*,* (5.43) *respectively but with dependence on* ω *explicitly denoted.*

Proof. Contained in Appendix B.3.10. □

Remark 5.1. *It is worthwhile to consider the smallest integer* $n^*(\omega)$ *such that* (5.44) *is valid. It is clear that we must have* $n^*(\omega) > 1/\mathfrak{c}_1$*, otherwise it possibly allows for* $\log(n\mathfrak{c}_1) < 0$ *in* (5.42) *and* (5.43)*. Given* n *and* ω*, one can numerically certify whether* $n \geq n^*(\omega)$*, using conditions from the proof of Lemma 5.2. We have empirically observed that* $n^*(\omega)$ *can be quite small, i.e. we can usually accept* $n^*(\omega) = \lceil 1/\mathfrak{c}_1 \rceil$*. Using this numerical certification, the optimised lower bound* (5.46) *can also be implemented via a numerical optimisation algorithm, noting that we*

need only conduct search over a univariate bounded region. Further discussion and pseudocode for these implementations can be found in Appendix B.4.

Throughout Figures 5.3a-5.3d, we plot the optimised lower bound (5.46) from Theorem 5.7, with baseline values $n = 100$, $m = 1$, $\alpha = 0.05$, and $\rho = 0.4$. In Figure 5.3a, this is compared against a numerical evaluation of the success probability using (5.9). However, since the density in the integrand concentrates towards zero as n increases, numerical integration becomes inaccurate for large n. In Figure 5.3b, we instead plot the lower bound over a semi-log horizontal axis scale for n, to illustrate the convergence of the success probability to one. These plots demonstrate that the behaviour of the lower bound is reasonably close to the actual probability. As the lower bound has been derived with $m = 1$ while the bound itself does not change with m, this means the bound is least conservative for $m = 1$, and will generally become more conservative as m grows. For instance with $\alpha = 0.05$ and $m = 32$, the lower bound from Theorem 5.4 yields $p^{\mathcal{N}}_{\text{success}} \geq 0.806$ for any $n \geq 32$, already surpassing the lower bound with $n = 10^9$ from Figure 5.3a.

5.3.4 Inversion of Analytic Lower Bound

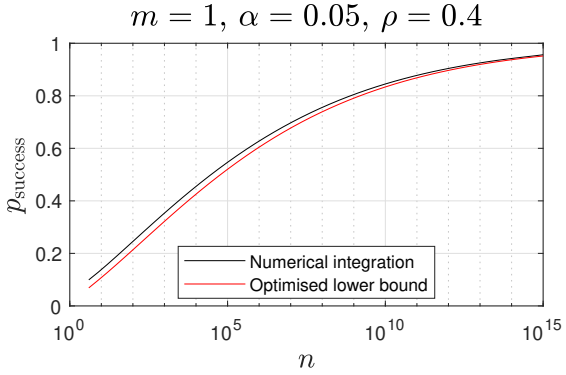
The purpose of this section is to numerically investigate the efficacy of inverting the analytic lower bound from Theorem 5.7, by considering the problem of guaranteeing a desired high probability.

Problem 5.1 (High probability guarantees of success)**.** *Given the triple* $(m, \alpha, \rho) \in \mathbb{N} \times (0, 1] \times (0, 1]$*, how large should* n *be, such that*

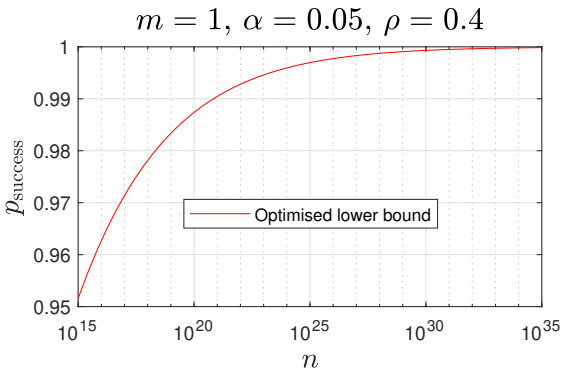
$$p^{\mathcal{N}}_{\text{success}}(n, m, \alpha, \rho) \geq 1 - \delta \tag{5.48}$$

for any given $\delta \in (0, 1]$*?*

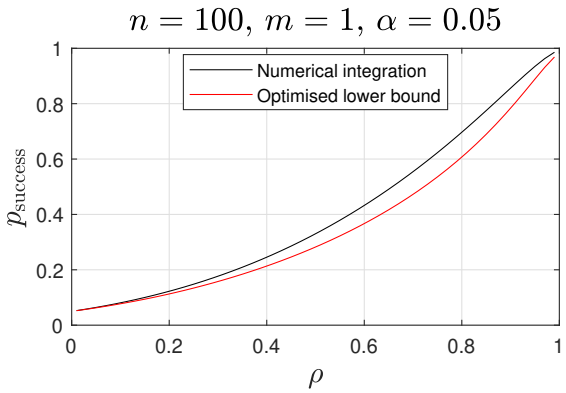
Having high probability guarantees would be useful in a situation where the sample size n can be increased relatively cheaply (hence large n would not be unreasonable), whereas increasing the selection size m in order to increase p_{success} may be prohibitively expensive and/or non-negotiable (as may be the case in controller tuning with limited



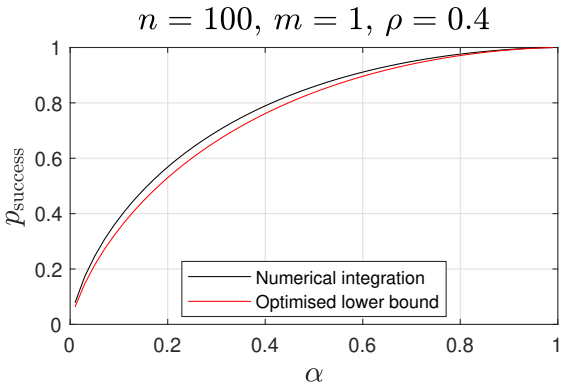
(a) Comparison of the optimised lower bound in (5.46) of Theorem 5.7, as n is varied. In addition, this figure demonstrates monotonicity in n from Theorem 5.3(a).



(b) Comparison of the lower bound in Theorem 5.7 when n is varied over a semi-log horizontal axis scale. In addition, this figure demonstrates convergence to one in n from (5.23).



(c) Comparison of the optimised lower bound in (5.46) of Theorem 5.7, as ρ is varied. In addition, this figure demonstrates monotonicity in ρ from Theorem 5.6.



(d) Comparison of the optimised lower bound in (5.46) of Theorem 5.7, as α is varied. In addition, this figure demonstrates monotonicity in α from Theorem 5.3(c).

Figure 5.3: Numerical results for the lower bound.

online testing). To address Problem 5.1 (while avoiding the trivial solution of $n = \infty$), take the lower bound in (5.44) of Theorem 5.7, which given α, ρ and δ, we aim to invert for n in terms of ω with the expression

$$\Phi\left(\frac{\Phi^{-1}(\alpha) - \rho\mu_n}{\sqrt{1-\rho^2+\rho^2\sigma_n^2}}\right) = 1 - \delta. \tag{5.49}$$

Putting the definitions of μ_n and σ_n from (5.42) and (5.43) respectively, this equation can

be rearranged into a quartic equation in $\sqrt{\log(n\mathfrak{c}_1)}$, of the form

$$\mathfrak{a}_4 \log(n\mathfrak{c}_1)^2 + \mathfrak{a}_3 \log(n\mathfrak{c}_1)^{3/2} + \mathfrak{a}_2 \log(n\mathfrak{c}_1) + \mathfrak{a}_1 \log(n\mathfrak{c}_1)^{1/2} + \mathfrak{a}_0 = 0, \tag{5.50}$$

where

$$\mathfrak{a}_4 = -\frac{2\rho^2}{\log\log 2} \tag{5.51}$$

$$\mathfrak{a}_3 = -\frac{4\Phi^{-1}(\alpha)\,\rho\sqrt{\mathfrak{c}_2}}{\log\log 2} \tag{5.52}$$

$$\mathfrak{a}_2 = 2\rho^2 - \frac{2\mathfrak{c}_2\left(\left[\Phi^{-1}(\alpha)\right]^2 - \left[\Phi^{-1}(1-\delta)\right]^2 + \rho^2\left[\Phi^{-1}(1-\delta)\right]^2\right)}{\log\log 2} \tag{5.53}$$

$$\mathfrak{a}_1 = 4\sqrt{\mathfrak{c}_2}\Phi^{-1}(\alpha)\,\rho \tag{5.54}$$

$$\mathfrak{a}_0 = 2\mathfrak{c}_2\left(\left[\Phi^{-1}(\alpha)\right]^2 - \left[\Phi^{-1}(1-\delta)\right]^2 + \rho^2\left[\Phi^{-1}(1-\delta)\right]^2\right) - \rho^2\left[\Phi^{-1}(1-\delta)\right]^2. \tag{5.55}$$

Therefore we take the solution for n corresponding to the greatest real root of the quartic equation. Let this solution for n in terms of ω be denoted $n^\sharp(\omega)$. According to the monotonicity and convergence properties from Theorem 5.3(a) and (5.23) respectively, then provided $n^\sharp(\omega) \geq n^*(\omega)$, we guarantee

$$p^{\mathcal{N}}_{\text{success}}\left(n^\sharp(\omega), m, \alpha, \rho\right) \geq 1 - \delta, \tag{5.56}$$

since $n^\sharp(\omega)$ upper bounds the smallest n needed such that $p_{\text{success}} \geq 1-\delta$. Moreover, one can numerically optimise with respect to ω to find the smallest $n^\sharp(\omega)$ that guarantees a high probability of success.

Table 5.1: Values for n which guarantees $p^{\mathcal{N}}_{\text{success}}(n, m, \alpha, \rho) \geq 1-\delta$, with $\alpha = 0.01$ fixed and valid for any $m \geq 1$.

	$\delta = 0.01$	$\delta = 0.05$	$\delta = 0.1$
$\rho = 0.01$	8.144×10^{47007}	5.427×10^{34246}	8.943×10^{28267}
$\rho = 0.3$	3.289×10^{51}	1.619×10^{38}	8.775×10^{31}
$\rho = 0.6$	8.703×10^{11}	1.988×10^{9}	1.078×10^{8}
$\rho = 0.9$	16744	4338	2188
$\rho = 0.99$	893	505	372

Table 5.1 lists computed values of $n^\sharp(\omega)$ numerically optimised with respect to ω,

taking into account the requirement $n^\sharp(\omega) \geq n^*(\omega)$, using the aforementioned approach. The table is valid for $m \geq 1$ (least conservative when $m = 1$), for $\alpha = 0.01$ and a variety of values for ρ and δ. The values for n trend downwards as ρ increases, which is intuitive (as fewer samples might be required if noisy observations are strongly correlated with the actual values). Of particular note, the case with extremely small correlation $\rho = 0.01$ (interpreted as a noise-to-signal ratio of $\xi^2 \approx 10^4$) requires n to be at an impractical order of magnitude, namely 10^{47007} when $\delta = 0.01$. This highlights the utility of the lower bound in Theorem 5.7, which allows for an $O(1)$ inversion to find a sufficiently high n. If Problem 5.1 were attempted to be solved by evaluating expression (5.9), then large n such as in the order of 10^{47007} would have rendered the evaluation of such probabilities to be intractable. Also, this example illustrates the value of having a strong positive dependence in ρ, as it reduces the sample size required to reach a prescribed high probability of success.

5.4 Summary

In this chapter, we established several properties on the success probability of OO on copula models. In particular, under a stochastic increasing positive dependence condition, the success probability was shown to be non-decreasing in n and m. Also, the limiting value of the success probability as $n \to \infty$ was shown to depend on the boundary conditional CDF of the copula. We studied in further detail the case of the Gaussian copula, and provided an analytic lower bound for the success probability. This analytic lower bound was then used to invert the success probability, to find the n which led to a prescribed high probability.

Chapter 6

Ordinal Optimisation for Controller Tuning

In this chapter, we focus on offline tuning of MPC for the diesel air-path, and test the tuned controllers online with transient drive-cycles. In Section 6.2, the results on ordinal optimisation from the previous chapter are specialised to offline controller tuning with a gain-scheduled architecture, where the monotonicity properties of the ordinal optimisation success probability hold under analogous regularity conditions. In Section 6.3, controllers are tuned offline by following an ordinal optimisation approach. Using performance indices obtained via preference learning in Chapter 3, online experimental results demonstrate that some offline tuned controllers yielded decent performance, outperforming a baseline manually tuned MPC. Lastly in Section 6.4, the offline tuning procedure is complemented by additional online manual tuning experiments, to further improve the performance.

6.1 Problem Setup

As discussed in Chapter 1, there is a constrained availability of experiment time within the automotive industry for bench testing of controllers with the physical engine. Fortunately, there is a relatively much larger time budget for offline tuning via simulation. Hence the rationale behind offline tuning is to find controllers through simulation, prior to physical experimentation, such that these controllers perform well when tested online (or at least provide good initial points for further online tuning). Moreover, offline tuned controllers can potentially complement online tuning approaches. Suppose that a non-negotiable budget of m is given as the number of different offline tuned controllers that can be tested online. Then once these m controllers are tested, the best performing one can be selected for further refinement, using a remaining testing budget allocated to

online tuning.

This motivates the question of **how** controllers should be tuned offline. If using the same strategy traditionally endorsed by ordinal optimisation (OO) and randomised algorithms (introduced in Sections 2.4 and 2.5.1 respectively), this stipulates we should generate many controllers offline, and "select the best to test". The focus herein is to investigate the merit of this strategy. The remainder of this chapter is split into two main parts. In the first part beginning in Section 6.2, the offline controller tuning problem is framed in the language of OO from Chapter 5, specialised to the particular controller architecture under consideration. Informed by these developments, OO offline tuning simulations and resulting online experiments are documented in the second part beginning in Section 6.3. Some manual online tuning experiments are also performed in Section 6.4.

6.2 Offline Controller Tuning with OO

Consider a measurable *controller performance function*

$$\overline{J}(\theta) : \Theta \to \mathbb{R} \tag{6.1}$$

with controller tuning variable $\theta \in \Theta$ from an **uncountable** topological space Θ, that allows us to compare any two controllers offline. Since Θ is uncountable, this accommodates the space of positive-definite cost matrices Q, P, R in quadratic-cost MPC. The $\overline{J}(\theta)$ could be, for example, a closed-loop simulation of controller θ on a designated task. Also consider a measurable *test performance function*

$$J_{\psi^*}(\theta, W) : \Theta \times \mathbb{W} \to \mathbb{R} \tag{6.2}$$

with respect to a realised plant ψ^* and noise $W \in \mathbb{W}$. This function represents the performance evaluation of controller θ on plant ψ^*, and W is stochastic. That is, if we test the same controller θ on the same plant ψ^* multiple occasions, we will not always observe the same performances, due to different realisations of W. The goal here is to find con-

trollers via offline tuning which perform well in online tests. To that end, suppose there is a mechanism $\mathcal{P}_\theta$ to randomly sample a candidate controller $\theta_i \in \Theta$, e.g. this mechanism could be from running the machine learning controller tuning framework (MLCTF) from Section 2.1.2.4. Along with the distribution $\mathcal{P}_W$ for W, this mechanism $\mathcal{P}_\theta$ induces a joint distribution $\left(\overline{J}\left(\theta_i\right), J_{\psi^*}\left(\theta_i, W\right)\right)$ of offline-online performance pairs. Naturally, the strength of positive dependence in this distribution impacts whether well-performing offline controllers will also perform well in online tests. In order to invoke the OO results developed in the previous chapter, we require this distribution to be continuous, and impose regularity conditions analogous to Assumption 5.1.

Assumption 6.1. *The induced distribution* $\left(\overline{J}\left(\theta_i\right), J_{\psi^*}\left(\theta_i, W\right)\right)$ *is a continuous distribution, and* $J_{\psi^*}\left(\theta_i, W\right)$ *is stochastically increasing (SI) positive dependent in* $\overline{J}\left(\theta_i\right)$.

Note that a necessary condition for $\left(\overline{J}\left(\theta_i\right), J_{\psi^*}\left(\theta_i, W\right)\right)$ to have a continuous distribution is that Θ be an uncountable set, which we have already specified the latter to be the case, via (6.1). Also, the positive dependence assumption can be intuitively reasonable if, for instance, an accurate system identification of the plant ψ^* is used in offline simulation, and the control task in the online test well-represents the task in offline simulation.

We can apply the results in OO from the previous chapter in the following way. An OO approach for offline controller tuning is to first randomly generate n controllers $\theta_1, \ldots, \theta_n \in \Theta$ offline using the mechanism $\mathcal{P}_\theta$, and evaluate their performances using the controller performance function $\overline{J}\left(\theta\right)$. We thus observe values $\overline{J}\left(\theta_1\right), \ldots, \overline{J}\left(\theta_n\right)$. Let m be a fixed budget for the number of offline tuned controllers that can be devoted to testing these online. We denote the best m observed controllers by $\theta^*_{(1:n)}, \ldots \theta^*_{(m:n)}$, and then physically test each online with the same realised plant ψ^* with i.i.d realisations $W_1, \ldots, W_m$ of the noise, yielding test performances

$$J_{\psi^*}\left(\theta^*_{(1:n)}, W_1\right), \ldots, J_{\psi^*}\left(\theta^*_{(m:n)}, W_m\right). \tag{6.3}$$

Under regularity Assumption 6.1, we can assign the pair (Z, X) to be equal in distribution to $\left(\overline{J}\left(\theta_i\right), J_{\psi^*}\left(\theta_i, W_i\right)\right)$, and treat this as the same pair (Z, X) from Chapter 5, so the OO

success probability is expressed as

$$p_{\text{success}}(n, m, \alpha) = \Pr\left(\min_{j\in\{1,\dots,m\}} J_{\psi^*}\left(\theta^*_{(j:n)}, W_j\right) \leq J^*_\alpha\right), \tag{6.4}$$

where α is given by

$$\alpha = \Pr_{\theta_i}\left(J_{\psi^*}(\theta_i, W_i) \leq J^*_\alpha\right). \tag{6.5}$$

One possible interpretation for J^*_α is a benchmark performance threshold aiming to be met, which determines (or is determined by) α. Furthermore, from the stochastically increasing positive dependence condition satisfied by regularity Assumption 6.1, then all the established properties of p_{success} from Section 5.2 are enjoyed. Since the copula of the distribution of $\left(\overline{J}(\theta_i), J_{\psi^*}(\theta_i, W)\right)$ is typically not not exactly known in however, we are unable to compute the true success probability in practice. Despite this, desirable properties still hold in the context of offline controller tuning without requiring knowledge of the copula, particularly monotonicity in n for fixed m and any choice of $\alpha \in (0, 1]$. More formally,

$$\max_{i\in\{m,\dots,n\}} \Pr\left(\min_{j\in\{1,\dots,m\}} J_{\psi^*}\left(\theta^*_{(j:i)}, W_j\right) \leq J^*_\alpha\right) = \Pr\left(\min_{j\in\{1,\dots,m\}} J_{\psi^*}\left(\theta^*_{(j:n)}, W_j\right) \leq J^*_\alpha\right). \tag{6.6}$$

Or in other words, to maximise the success probability, the number of controllers generated offline should be made as large as possible, regardless of m or J^*_α.

6.2.1 Offline Tuning of Gain-Scheduled Controllers

As discussed in Section 2.2.2, a standard way of controlling the diesel air-path over the operating space is by using gain-scheduling. In particular, the previous work [103] implemented a tuned gain-scheduled MPC architecture for the diesel air-path. The gain-scheduled MPC partitioned the operating space into a grid consisting of 12 local controllers (shown here in Figure 6.1), and each local controller was tuned separately and independently in simulation prior to the whole controller being tested on transient drive-cycles. Here, we transfer the setup in the previous section to the just-described case where the diesel air-path has a gain-scheduled architecture composed of d local controllers with

tuning variables

$$\boldsymbol{\theta} = ([\boldsymbol{\theta}]_1, \ldots, [\boldsymbol{\theta}]_d) \in \Theta_1 \times \cdots \times \Theta_d, \tag{6.7}$$

where each $[\boldsymbol{\theta}]_j$ for $j = 1, \ldots, d$ are the local controller tuning variables responsible for different regions of the operating space. Since these local controllers are 'modular' (i.e. responsible for non-overlapping regions of the operating space), we can consider tuning them independently as is done in [103]. Suppose for the j^{th} local controller currently being tuned, that we have a *local controller performance function*

$$\overline{J}_j\left([\boldsymbol{\theta}]_j\right) : \Theta_j \to \mathbb{R}, \tag{6.8}$$

which allows us to compare any two of the same local controller offline. In [103], these local controller performance functions involved performances of local step response trajectories. Since an online test over a transient drive-cycle will traverse the operating space and activate multiple local controllers, we introduce the *conditional test performance function*

$$J_{\psi^*, j|\boldsymbol{\theta}_{\setminus j}}\left([\boldsymbol{\theta}]_j, W\right) : \Theta_j \times \mathbb{W} \to \mathbb{R}, \tag{6.9}$$

given realised plant ψ^*, and the tunings

$$\boldsymbol{\theta}_{\setminus j} := \left([\boldsymbol{\theta}]_1, \ldots, [\boldsymbol{\theta}]_{j-1}, [\boldsymbol{\theta}]_{j+1}, \ldots [\boldsymbol{\theta}]_d\right) \tag{6.10}$$

$$\in \Theta_1 \times \cdots \times \Theta_{j-1} \times \Theta_{j+1} \times \cdots \times \Theta_d =: \Theta_{\setminus j}, \tag{6.11}$$

for all the other local controllers. Now letting m still be the available budget for online testing, the OO approach is to independently generate offline n_j candidates for the current local controller, using the mechanism $\mathcal{P}_{[\boldsymbol{\theta}]_j}$ (e.g. this mechanism could still be via a MLCTF setup for the local controller), and evaluate them on the local controller performance function $\overline{J}_j$, leading to the best m:

$$\overline{J}_j\left([\boldsymbol{\theta}]^*_{j,(1:n_j)}\right) \leq \cdots \leq \overline{J}_j\left([\boldsymbol{\theta}]^*_{j,(m:n_j)}\right). \tag{6.12}$$

Completely analogous to (6.4) in the previous section, we may now consider the *conditional success probability* for the local controller

$$p^{\text{G.S.}}_{\text{success},j|\boldsymbol{\theta}_{\setminus j}}(n_j, m, \alpha_j) = \Pr\left(\min_{i\in\{1,\ldots,m\}} J_{\psi^*,j|\boldsymbol{\theta}_{\setminus j}}\left(\theta^*_{(i:n_j)}, W_i\right) \leq J^*_{\alpha_j}\middle|\boldsymbol{\theta}_{\setminus j}\right) \tag{6.13}$$

given the other tunings $\boldsymbol{\theta}_{\setminus j}$, where the relation between $\alpha_j > 0$ and $J^*_{\alpha_j} \in \mathbb{R}$ is

$$\alpha_j = \Pr_{[\boldsymbol{\theta}]_j}\left(J_{\psi^*,j|\boldsymbol{\theta}_{\setminus j}}\left([\boldsymbol{\theta}]_j, W\right) \leq J^*_{\alpha_j}\middle|\boldsymbol{\theta}_{\setminus j}\right). \tag{6.14}$$

By conditioning on the other controller tunings $\boldsymbol{\theta}_{\setminus j}$, this decouples the drive-cycle performance of the j^{th} controller from the other controllers. The role of regularity Assumption 6.1 when applied here is immediate. If the induced conditional joint distribution of

$$\left(\overline{J}_j\left([\boldsymbol{\theta}]_j\right), J_{\psi^*,j|\boldsymbol{\theta}_{\setminus j}}\left([\boldsymbol{\theta}]_j, W\right)\right) \tag{6.15}$$

is continuous (given $\boldsymbol{\theta}_{\setminus j}$), and moreover the second variate is SI positive dependent in the first variate for all possible tunings $\boldsymbol{\theta}_{\setminus j} \in \Theta_{\setminus j}$, then the conditional success probability (6.13) is non-decreasing in n_j. Note in particular, that this result holds for all possible tunings $\boldsymbol{\theta}_{\setminus j} \in \Theta_{\setminus j}$. Thus, the same principle in (6.6) applies; we should generate as many local candidates for $[\boldsymbol{\theta}]_j$ offline as possible, no matter what the tunings of the other local controllers are.

Furthermore, if the analogous regularity conditions apply for each of the other local controllers in $[\boldsymbol{\theta}]_1, \ldots, [\boldsymbol{\theta}]_d$, then the conclusion is that we should make the numbers $n_1, \ldots, n_d$ of offline candidates for each of local controllers $j = 1, \ldots, d$ as high as possible. Considering that the local controllers in a gain-scheduled architecture are modular, and if a test performance function that promotes little coupling between the controllers is used, e.g. an integral of absolute error (IAE) over the drive-cycle, then these regularity conditions should be reasonable. The strength of positive dependence in each of the $\left(\overline{J}_j\left([\boldsymbol{\theta}]_j\right), J_{\psi^*,j|\boldsymbol{\theta}_{\setminus j}}\left([\boldsymbol{\theta}]_j, W\right)\right)$ may also rely on the drive-cycle being tested on. If the drive-cycle spends a longer time in the operating region local to the j^{th} controller, then

we should reasonably anticipate the positive dependence to be stronger.

6.3 Diesel Air-Path Experiments

In this section, a gain-scheduled MPC architecture will be tuned offline under advice of the OO "select the best to test" methodology in (6.6), where as many candidate controllers are generated offline as possible.

6.3.1 Controller Architecture

We describe the gain-scheduled controller architecture to be tuned, which is similar to that used in [103], although our architecture additionally incorporates an integrator online. The operating space is divided into a grid of $d = 12$ rectangular regions with grid points $\left(\overline{\mathrm{N}}_{\mathrm{e}}^{\{j\}}, \overline{\mathrm{w}}_{\mathrm{fuel}}^{\{j\}}\right)$ for $j = 1, \ldots, 12$ as shown in Figure 6.1, where a local linear MPC is responsible for each region.

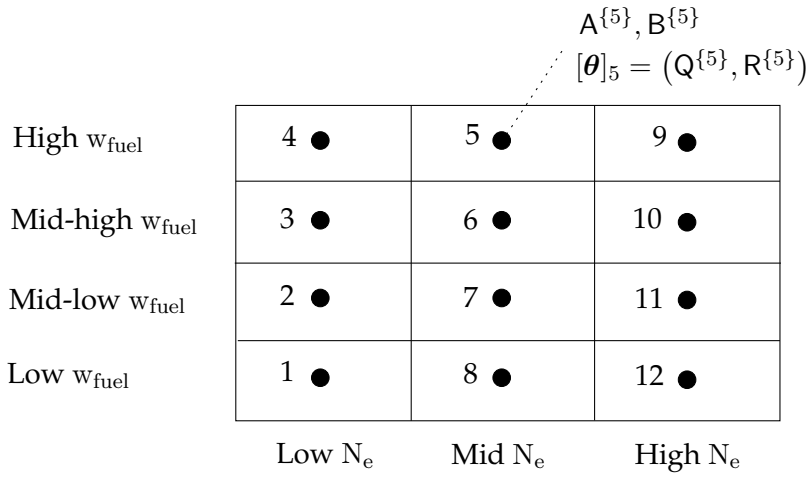


Figure 6.1: Partition of the operating space into operating regions and local grid points for the gain-scheduled controller architecture.

As the current operating point $\mathfrak{p}$ switches over to a different region, the controller also switches. Each local linear controller has its own prediction model and cost function weights; for the j^{th} controller these are denoted by $\mathsf{A}^{\{j\}}$, $\mathsf{B}^{\{j\}}$, $\mathsf{Q}^{\{j\}}$, $\mathsf{R}^{\{j\}}$. The controller

tuning variables at each grid point are $[\boldsymbol{\theta}]_j = \left(\mathsf{Q}^{\{j\}}, \mathsf{R}^{\{j\}}\right)$.

6.3.1.1 Steady-State Maps

During transient drive-cycle testing, the vehicle speed reference trajectory is translated by *vehicle models* (see Figure 6.3) into a trajectory for the operating point $\mathfrak{p} = (\mathrm{N_e}, \mathrm{w_{fuel}})$. In transient drive-cycle experiments, the controller takes $\mathfrak{p}$ as given. As is standard practice (originally discussed in Section 2.2.2), the controller is embedded with pre-calibrated steady-state maps for the inputs (implemented as lookup tables with linear interpolation) over the operating space, denoted

$$\mathsf{u}_{\mathrm{ss}}(\mathfrak{p}) : \mathbb{R}^2 \to \mathbb{R}^3, \quad \mathsf{u}_{\mathrm{ss}}(\mathfrak{p}) = \begin{bmatrix} \mathrm{u_{thr,ss}}(\mathfrak{p}) & \mathrm{u_{EGR,ss}}(\mathfrak{p}) & \mathrm{u_{VGT,ss}}(\mathfrak{p}) \end{bmatrix}^\top. \tag{6.16}$$

Additionally, steady-state maps are used for the states:

$$\mathsf{x}_{\mathrm{ss}}(\mathfrak{p}) : \mathbb{R}^2 \to \mathbb{R}^4, \quad \mathsf{x}_{\mathrm{ss}}(\mathfrak{p}) = \begin{bmatrix} \mathrm{p_{im,ss}}(\mathfrak{p}) & \mathrm{p_{em,ss}}(\mathfrak{p}) & \mathrm{W_{comp,ss}}(\mathfrak{p}) & \mathrm{y_{EGR,ss}}(\mathfrak{p}) \end{bmatrix}^\top. \tag{6.17}$$

These maps have been previously obtained from a static calibration procedure as detailed in [164], where at at given operating point $\mathfrak{p}$, the set-point $\mathsf{u}_{\mathrm{ss}}(\mathfrak{p})$ is applied, and the steady-state $\mathsf{x}_{\mathrm{ss}}(\mathfrak{p})$ is recorded. The steady-state boost pressure contains our reference boost pressure $\mathrm{p}_{\mathrm{im}}^{\mathrm{ref}}(\mathfrak{p}) = \mathrm{p_{im,ss}}(\mathfrak{p})$ for the given operating point, while the steady-state EGR rate contains the reference EGR rate $\mathrm{y}_{\mathrm{EGR}}^{\mathrm{ref}}(\mathfrak{p})$ for the given operating point. Thus, a drive-cycle test induces an operating point trajectory $\mathfrak{p}_k$, which is transformed into a time-varying reference $\left(\mathrm{p}_{\mathrm{im}}^{\mathrm{ref}}(\mathfrak{p}_k), \mathrm{y}_{\mathrm{EGR}}^{\mathrm{ref}}(\mathfrak{p}_k)\right)$ trajectory (an example shown in Figure 6.2) through the steady-state maps. This whole process is depicted in Figure 6.3.

In the controller architecture, the maps $\mathsf{u}_{\mathrm{ss}}(\mathfrak{p})$ and $\mathsf{x}_{\mathrm{ss}}(\mathfrak{p})$ accommodate the reference tracking problem by acting as *trimming conditions*. That is, at time k the trimmed input and state are given by

$$\widetilde{\mathsf{u}}_k = \mathsf{u}_k - \mathsf{u}_{\mathrm{ss}}(\mathfrak{p}_k) \tag{6.18}$$

$$\widetilde{\mathsf{x}}_k = \mathsf{x}_k - \mathsf{x}_{\mathrm{ss}}(\mathfrak{p}_k) \tag{6.19}$$

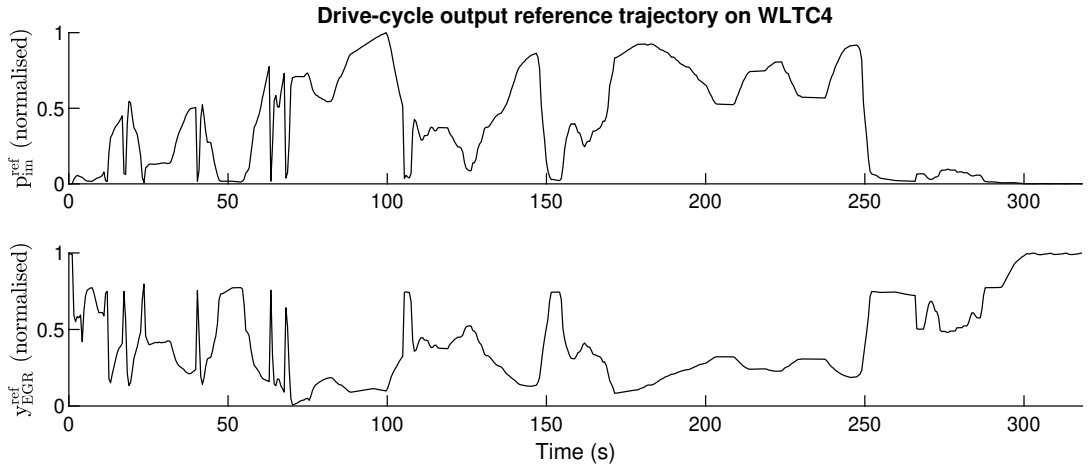


Figure 6.2: A typical time-varying output reference trajectory on the fourth section of the WLTC.

respectively. Thus, regulation in the trimmed state becomes reference tracking in the untrimmed state, and the input $\mathsf{u}_{\mathrm{ss}}(\mathfrak{p}_k)$ is treated as a feed-forward component.

6.3.1.2 Local Linear Models

The local $\left(\mathsf{A}^{\{j\}}, \mathsf{B}^{\{j\}}\right)$ matrices at each grid point of the operating space are obtained from previously conducted system identification experiments. A short (40 second) multisine perturbation signal was applied to the input, centered about the input feed-forward $\mathsf{u}_{\mathrm{ss}}\left(\overline{\mathfrak{p}}^{\{j\}}\right)$, where $\overline{\mathfrak{p}}^{\{j\}}$ is the grid point central for region j. The state signals were measured throughout the input perturbation signal, and trimmed about the steady-state values. Via least squares estimation, the local linear model was obtained as

$$\widetilde{\mathsf{x}}_{k+1} = \mathsf{A}^{\{j\}}\widetilde{\mathsf{x}}_k + \mathsf{B}^{\{j\}}\widetilde{\mathsf{u}}_k \tag{6.20}$$

for each $j \in \{1, \ldots, 12\}$.

6.3.1.3 Integrator Architecture

To eliminate offset in reference tracking caused by constant disturbances, each local linear MPC is fitted with an integrator, of the structure described in Section 2.1.1.2. To formulate

the integrator, the trimmed state is augmented with two local error states

$$\widetilde{\mathsf{x}}'_k = \begin{bmatrix} \widetilde{\mathsf{x}}_k\left(\mathfrak{p}_k\right) \\ \mathsf{e}_k^{\{j_k\}} \end{bmatrix} \tag{6.21}$$

where j_k denotes the identity of the active controller at time k. This necessitates an augmentation to the Q matrix (which we choose be block-diagonal, for simplicity):

$$\widetilde{\mathsf{Q}}^{\{j\}} = \begin{bmatrix} \mathsf{Q}^{\{j\}} & \mathbf{0} \\ \mathbf{0} & \mathsf{Q}_{\mathsf{e}}^{\{j\}} \end{bmatrix}, \tag{6.22}$$

where

$$\mathsf{Q}_{\mathsf{e}}^{\{j\}} = \begin{bmatrix} \mathsf{K}_{\mathsf{e},\mathrm{p_{im}}}^{\{j\}} & 0 \\ 0 & \mathsf{K}_{\mathsf{e},\mathrm{y_{EGR}}}^{\{j\}} \end{bmatrix} \tag{6.23}$$

and $\mathsf{K}_{\mathsf{e},\mathrm{p_{im}}}^{\{j\}}$, $\mathsf{K}_{\mathsf{e},\mathrm{y_{EGR}}}^{\{j\}}$ are the integrator weights for the boost pressure and EGR rate respectively. This also leads to the augmented matrices

$$\widetilde{\mathsf{A}}^{\{j\}} = \begin{bmatrix} \mathsf{A}^{\{j\}} & \mathbf{0} \\ -\mathsf{C} & I \end{bmatrix} \tag{6.24}$$

$$\widetilde{\mathsf{B}}^{\{j\}} = \begin{bmatrix} \mathsf{B}^{\{j\}} \\ \mathbf{0} \end{bmatrix}. \tag{6.25}$$

At each time k, the integrator states of the active region are updated by

$$\mathsf{e}_k^{\{j_k\}} = -\mathsf{C}\widetilde{\mathsf{x}}_{k-1} + \mathsf{e}_{k-1}^{\{j_k\}}, \tag{6.26}$$

while the integrator states of all the other non-active regions are left unchanged.

6.3.1.4 Gain-Scheduled MPC Law

At time k, given operating point $\mathfrak{p}_k$, the controller determines the active grid point j_k, and the corresponding matrices $\mathsf{A}^{\{j_k\}}, \mathsf{B}^{\{j_k\}}, \mathsf{Q}^{\{j_k\}}, \mathsf{R}^{\{j_k\}}$. The augmented Q matrix is formed

by

$$\widetilde{Q}^{\{j_k\}} = \begin{bmatrix} Q^{\{j_k\}} & \mathbf{0} \\ \mathbf{0} & Q_e^{\{j_k\}} \end{bmatrix} \tag{6.27}$$

and the matrix $\widetilde{P}^{\{j\}}$ is found by solving the discrete-time algebraic Riccati equation as in (2.11), except with the substitutions $A = \widetilde{A}^{\{j_k\}}$, $B = \widetilde{B}^{\{j_k\}}$, $Q = \widetilde{Q}^{\{j_k\}}$, $R = R^{\{j_k\}}$. Then the open-loop cost function is expressed as

$$\widetilde{V}_k = \sum_{i=0}^{N-1} \left(\widetilde{x}_{k|i}'^{\top} \widetilde{Q}^{\{j_k\}} \widetilde{x}'_{k|i} + \widetilde{u}_{k|i}{}^{\top} R^{\{j_k\}} \widetilde{u}_{k|i} \right) + \widetilde{x}_{k|N}'^{\top} \widetilde{P}^{\{j_k\}} \widetilde{x}'_{k|N} \tag{6.28}$$

in terms of the open loop sequence of inputs $\left(\widetilde{u}_{k|0}, \ldots, \widetilde{u}_{k|N-1}\right)$ and the predicted sequence of states

$$\widetilde{x}'_{k|i+1} = \widetilde{A}^{\{j\}} \widetilde{x}'_{k|i} + \widetilde{B}^{\{j\}} \widetilde{u}_{k|i}. \tag{6.29}$$

The following optimisation problem is posed:

$$\begin{aligned} &\min_{\widetilde{u}_{k|0}, \ldots, \widetilde{u}_{k|N-1}} \widetilde{V}_k \\ &\text{subject to } \widetilde{x}'_{k|0} = \widetilde{x}'_k \\ &\qquad \widetilde{x}'_{k|i+1} = \widetilde{A}^{\{j_k\}} \widetilde{x}'_{k|i} + \widetilde{B}^{\{j_k\}} \widetilde{u}_{k|i},\ i = 0, \ldots, N-1 \\ &\qquad M \widetilde{x}'_{k|i+1} \leq f,\ i = 1, \ldots, N \\ &\qquad E \widetilde{u}_{k|i} \leq h,\ i = 0, \ldots, N-1 \\ &\qquad \left| \widetilde{u}_{k|0} - \widetilde{u}_{k-1} \right| \leq u_{\text{slew}} \\ &\qquad \left| \widetilde{u}_{k|i} - \widetilde{u}_{k|i-1} \right| \leq u_{\text{slew}},\ i = 1, \ldots, N-1, \end{aligned} \tag{6.30}$$

where the formulation of the constraint matrices M, f, E, h is further detailed in Appendix B.5.1. At time k, the problem (6.30) is cast as a quadratic program and solved, leading to the optimal solution $\left(\widetilde{u}^*_{k|0}, \ldots, \widetilde{u}^*_{k|N-1}\right)$, and the controller plus feed-forward input $\widetilde{u}^*_{k|0} + u_{ss}(p_k)$ is commanded to the actuators.

6.3.1.5 Hardware and Software Setup

The gain-scheduled MPC architecture is implemented in MATLAB/Simulink. The quadratic program is formulated using the *condensed* formulation [106] and solved using the QP-KWIK solver [167]. The code is compiled to the dSPACE DS1006 processor board, which interfaces directly with the ECU during testing, as shown in Figure 6.3. The test diesel engine is a Toyota GD engine as introduced in Section 2.2. Further technical specifications may be found in [2]. The engine speed is controlled by the *dynamometer controller*, via readings from the *transient dynanometer* which is connected to the diesel engine via a shaft.

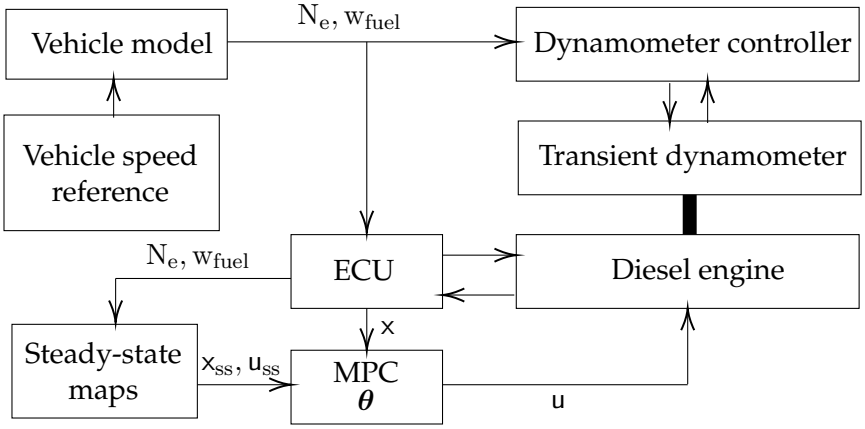


Figure 6.3: A block diagram of the experimental testing setup.

6.3.2 Simulation Setup

The offline tuning procedure is depicted in Figure 6.4, which produces m gain-scheduled MPCs $\boldsymbol{\theta}^*_{(1)}, \ldots, \boldsymbol{\theta}^*_{(m)}$ which have been tuned offline and selected via OO. Each aspect of this procedure is further explained below.

Offline Plant Model The local linear models $\left(\mathsf{A}^{\{j\}}, \mathsf{B}^{\{j\}}\right)$ described in Section 6.3.1.2 were used in offline simulation, for both the MPC prediction and state updates in closed-loop.

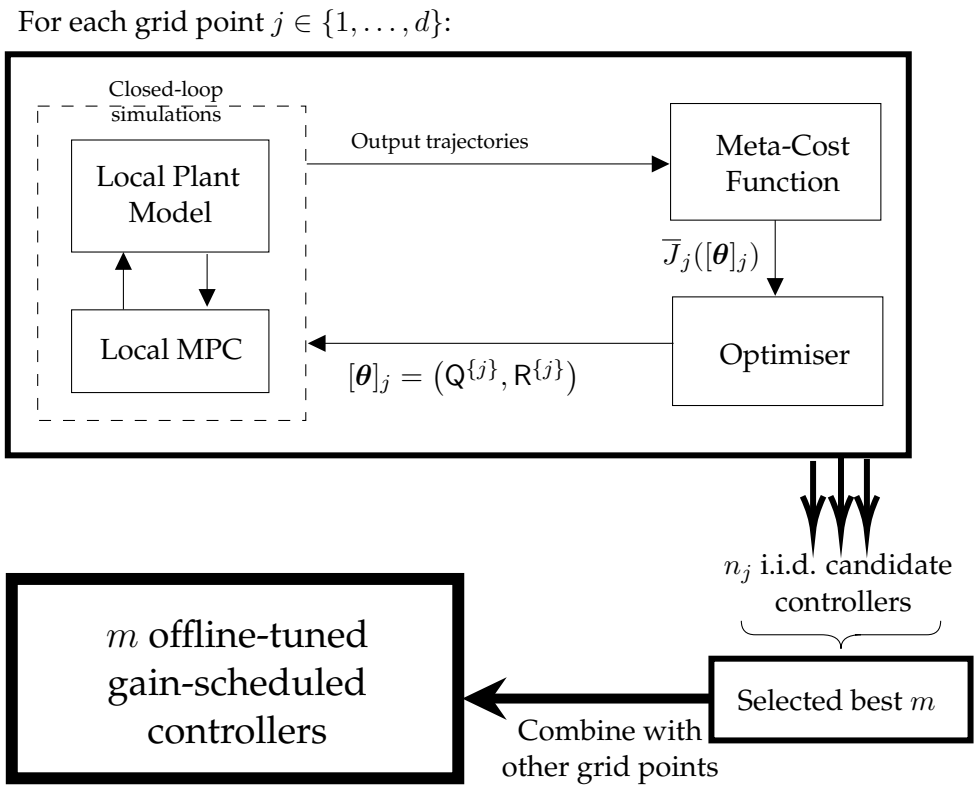


Figure 6.4: A diagram illustrating the simulation setup offline using the controller tuning framework.

Meta-Cost Function (Offline Performance Index) The performance of each local controller $\overline{J}_j\left([\boldsymbol{\theta}]_j\right)$ is evaluated offline by a weighted average of the step response performances from adjacent grid points, using the function (3.31) in 8 time-domain features obtained by preference learning. Further details are provided in Appendix B.5.2.

Optimiser Each local $\left(\mathsf{Q}^{\{j\}}, \mathsf{R}^{\{j\}}\right)$ pair is parametrised using Given rotations and hyperspherical coordinates, as described in Section 2.1.2.5. A candidate local controller at each grid point is generated from the MLCTF using a full i.i.d. run of the CMA-ES (Covariance Matrix Adaptation Evolution Strategies) optimiser [87] with box-constrained decision variables. The decision to use CMA-ES is based on its competitive empirical performance as a derivative-free black-box optimisation algorithm [14, 86, 88], however an alternative randomised optimiser could have easily as well been used (in so far as it forms the mechanism by which to generate candidate local controllers to satisfy the

regularity conditions). Although often compared to other metaheuristic algorithms, the behaviour of the CMA-ES algorithm has been studied from a information geometric perspective [5, 149], due to its connection with natural gradient descent on a Riemannian manifold [102, §12.1.2].

Ordinal Optimisation Selection Offline simulations were performed using the University of Birmingham's BEAR [1] and The University of Melbourne's Spartan [118] high performance computing services. Due to the advice of (6.6), as many candidate local controllers as possible were generated for each of the grid points (the exact number of samples for each grid point can be found in Table 6.1). We invoke the OO selection methodology, and select the best $m = 20$ of each local controller. Then, 20 ordered controllers are composed, whereby the i^{th} ordered controller consists of the i^{th} best of each local controllers across all the grid points.

Table 6.1: Number of candidate controllers n_j generated offline for OO at each of the grid points $j = 1, \ldots, 12$.

	Low $\mathrm{N_e}$	Mid $\mathrm{N_e}$	High $\mathrm{N_e}$
Low $\mathrm{w_{fuel}}$	1000	1000	1150
Mid-low $\mathrm{w_{fuel}}$	1000	3000	1000
Mid-high $\mathrm{w_{fuel}}$	1000	1000	1240
High $\mathrm{w_{fuel}}$	1000	1000	1000

6.3.3 Experimental Results

Engine bench tests involving drive-cycles were performed at the Toyota Motor Corporation Higashi-Fuji Technical Centre in Susono, Japan. Rather than testing each controller on a full drive-cycle, the third section of the UDC (abbreviated UDC3) from Figure 2.3a and fourth section of the WLTC (abbreviated WLTC4) in Figure 2.4d have been selected as the test cycles, as while being relatively short, these together contain sufficient coverage of all the different operating regions. Each of the $m = 20$ controllers are tested online with the realised plant ψ^* on both the UDC3 and WLTC4 drive-cycles. To calculate the test performance index, we use a linear cost function in the IAE over the respective drive-

cycle

$$J_{\psi^*} = 0.0019\text{IAE}_{\text{p}_{\text{im}}} + 0.4659\text{IAE}_{\text{y}_{\text{EGR}}} \tag{6.31}$$

where the coefficients have been obtained via preference learning, and directly found in (3.25). The IAE for p_{im} and y_{im} for each of the $m = 20$ controllers are scatter-plotted in Figures 6.5 and 6.6.

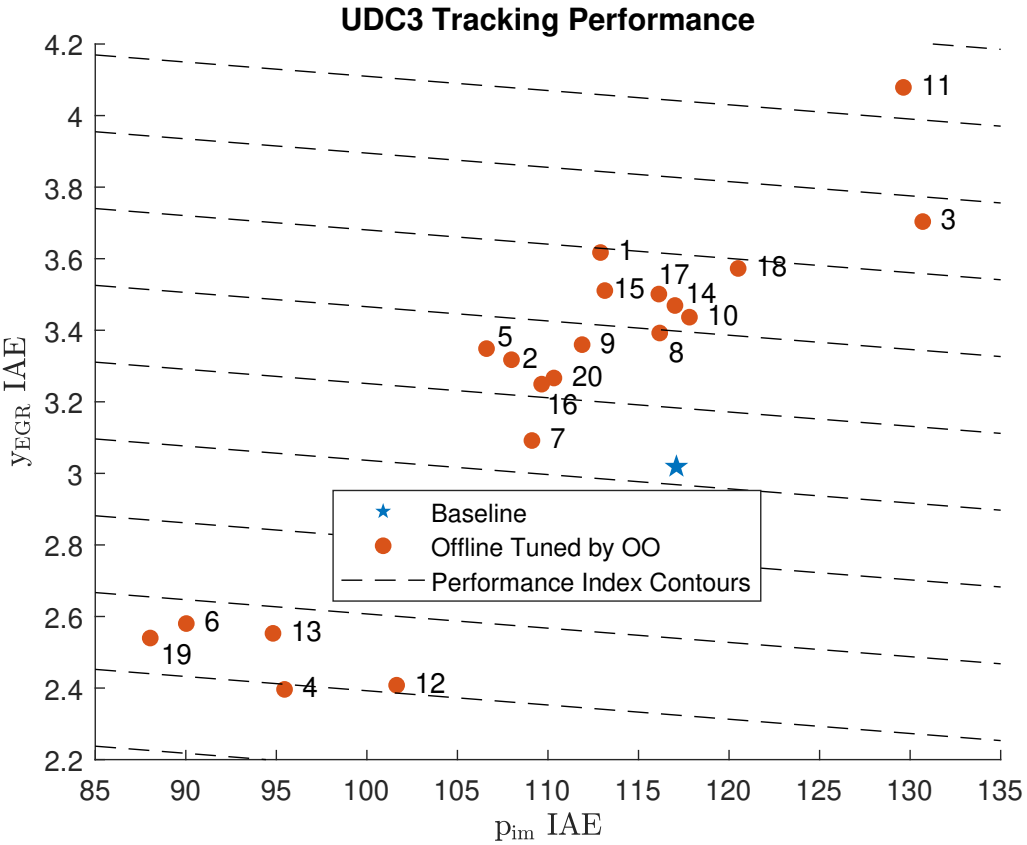


Figure 6.5: A comparison of IAE of tuned controllers for the UDC3. The numbered points indicate the offline observed rankings of the controllers. The contours are for the test performance function which is linear in the variables $(\text{IAE}_{\text{p}_{\text{im}}}, \text{IAE}_{\text{y}_{\text{EGR}}})$.

6.3.3.1 Comparison with Baseline

The offline tuned controllers are compared against a baseline MPC, which is of the same gain-scheduled architecture (including integrator), and has been manually tuned online from previous experiments. The baseline MPC also shares qualitatively compara-

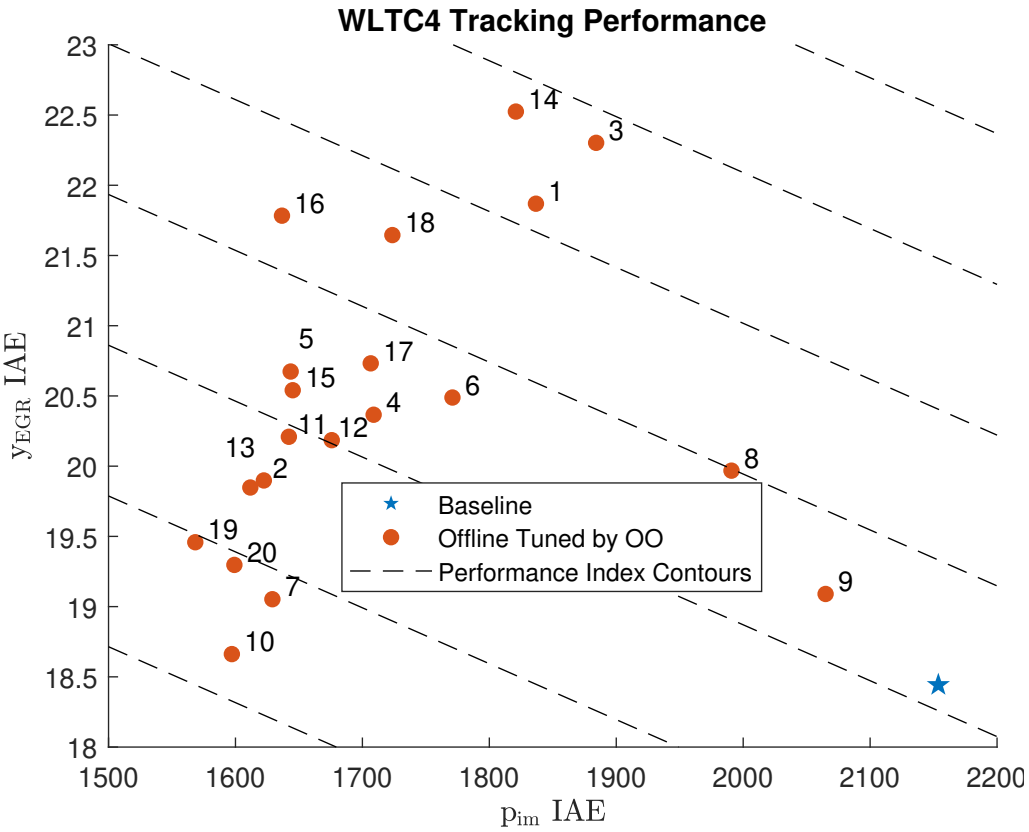


Figure 6.6: A comparison of IAE of tuned controllers for the WLTC4. The numbered points indicate the offline observed rankings of the controllers. The contours are for the test performance function which is linear in the variables $(\mathrm{IAE}_{p_{im}}, \mathrm{IAE}_{y_{EGR}})$.

ble tracking performance to the Toyota production controller. One caveat however, is that the production controller uses proprietary logic to generate the reference trajectories (whereas our reference trajectories are based on steady-state maps). Hence direct quantitative comparisons against the production controller are difficult to make; we instead focus on comparisons with our baseline controller.

The contour lines in Figures 6.5 and 6.6 represent the linear function being used to evaluate the test performance, which allow us to straightforwardly inspect the relative performance of contours (as better controllers will lie on lower contours). In can be seen that on the UDC3, five out 20 offline tuned controllers outperformed the baseline MPC. Moreover, four of these five **dominated** the baseline MPC (i.e. had smaller IAE for both p_{im} and y_{EGR}, so that it would have outperformed the baseline MPC using any monotonic

cost/utility function). While on the WLTC4, eight out 20 offline tuned controllers outperformed the baseline MPC. If the baseline MPC performance were used as a benchmark performance, then the fact that several (not just one or two) of the controllers outperformed the baseline suggests that the success probability would have been quite high on each of the respective drive-cycles. Overall, these results suggest that the OO approach to offline tuning can provide decent performance when tested online.

6.3.3.2 Analysis of Selection Size

Even though $m = 20$ was the fixed number of controllers tested online in these experiments, we can also perform a *counterfactual* analysis of the selection size m, which answers the question: "what if only $m = 1, 5, 10$, etc. (for some m less than 20) were selected instead?" As gathered from Figures 6.5 and 6.6, the following are evident.

- If only $m = 1$, then the single controller tested would have performed worse than the baseline MPC on both the UDC3 and WLTC4.

- With $m = 2$, the second-ranked offline controller would have outperformed the baseline on the WLTC4, but not the UDC3.

- We would have required to go up to $m = 12$, before finding an offline tuned controller which outperformed the baseline MPC on both the UDC3 and WLTC4.

- In addition, the 19^{th} ranked controller performed relatively well on both drive-cycles (ranked third in online performance on the UDC3, and fourth on the WLTC4).

One interpretation stemming from this analysis is that by increasing m (if allowable), this helps to increase the 'diversity' of controllers, so that at least one performs acceptably well when tested. This draws connections with the strategy of maintaining a *diversity* of candidate solutions in population-based search metaheuristic algorithms [156, §2.3.8].

6.4 Online Manual Tuning

To investigate whether the performance of the offline tuned controllers could be further improved, manual online tuning experiments were performed. The manual tuning procedure parallels that of [165], in which online tuning rules were used to calibrate the tuning variables for another variant of a gain-scheduled MPC architecture also applied to the diesel air-path over transient drive-cycles. In their specific formulation, there was an intuitive relationship between the tuning variables and qualitative characteristics of the closed-loop response (e.g. certain variables could be tuned to address excessive overshoot). In much the same way, we adapt the procedure to our architecture and apply the following simple rules for online manual tuning.

1. For the current controller, test it over a drive-cycle and obtain the closed-loop response.

2. Identify the local controllers responsible for regions over the transient response where better tracking is desired.

3. For each of the identified local controllers $j \in \{1, \ldots, d\}$, apply the following rule-based tuning guidelines.

    - If there is slow convergence of the integrator in eliminating offset for any of the outputs, increase the integrator weights $\mathsf{K}_{\mathsf{e},\mathrm{p_{im}}}^{\{j\}}$, $\mathsf{K}_{\mathsf{e},\mathrm{y_{EGR}}}^{\{j\}}$ accordingly for the respective outputs.
    - If there are excessive oscillations, add a scaled identity matrix to $\mathsf{R}^{\{j\}}$, to regularise and suppress the controller towards the feed-forward input.

4. Repeat the above steps until no further improvements are required, or until an predetermined online tuning budget has been reached.

Through inspection of the trajectories for the drive-cycle tests on both the UDC3 and WLTC4, a single tuned controller was composed from well-performing local controllers across all of the 20 controllers. Then the aforementioned online manual tuning procedure was applied using both the UDC3 and WLTC4. The tracking performance of this final

tuned and refined controller is plotted in Figures 6.7 and 6.8.

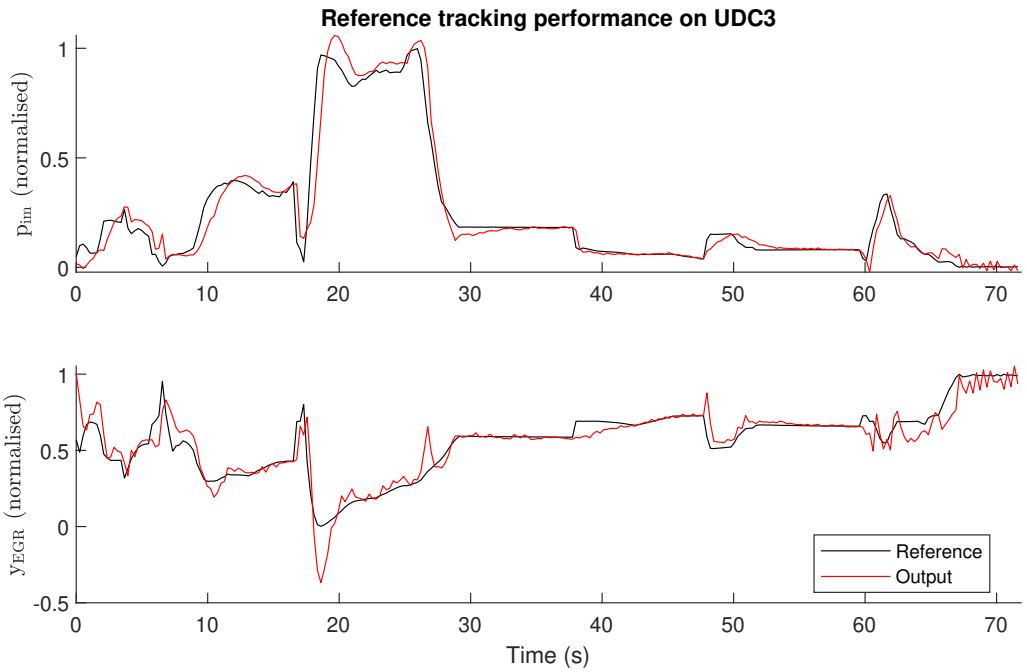


Figure 6.7: Output reference tracking trajectories of the final tuned and refined controller on the UDC3.

The performance of this final tuned and refined controller is shown in Table 6.2 to be better than the best offline tuned controllers on the UDC3 and WLTC4. To assess how the fully tuned and refined controller 'generalises' beyond the UDC3 and WLTC4, it was tested on other drive-cycles (including the full UDC and other sections of the WLTC) and compared against the baseline MPC. Table 6.2 additionally lists the IAE for the tests, and as highlighted in the table, the fully tuned and refined MPC outperformed the baseline MPC in both output tracking metrics. Plots comparing the tracking performance against the baseline on all the drive-cycles tested are included in Figures 6.9-6.14. These results also demonstrate that the performance of the best controllers is replicable to other drive-cycles, and that noise is not dominating the observed test performances.

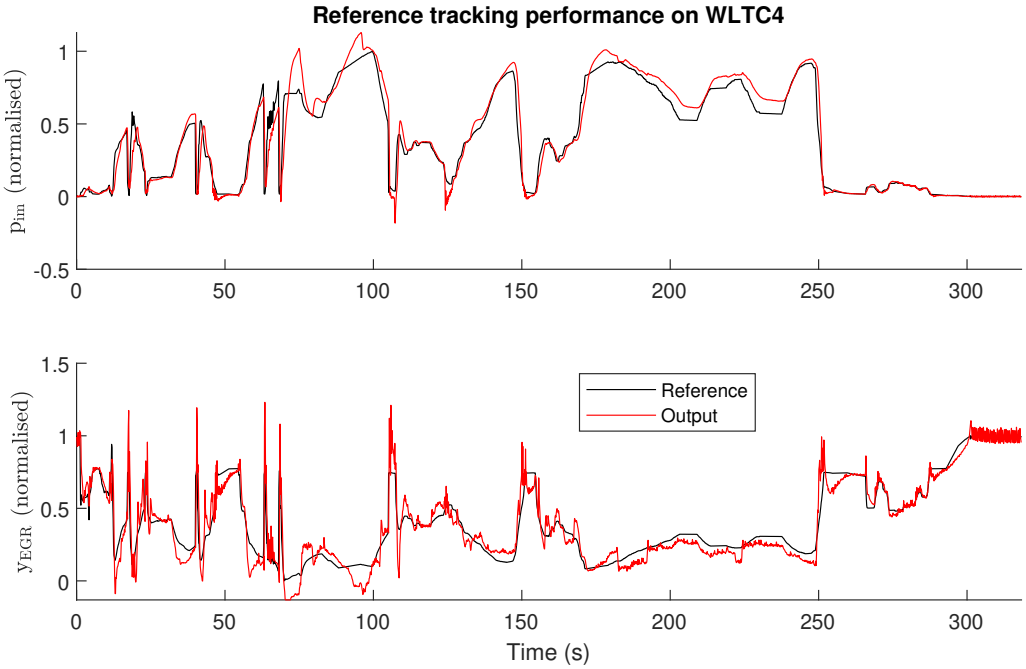


Figure 6.8: Output reference tracking trajectories of the final tuned and refined controller on the WLTC4.

Table 6.2: IAE values for baseline controller and fully tuned and refined controller over different drive-cycles.

Drive-cycle	MPC	p_{im} IAE	y_{EGR} IAE
UDC3	Baseline	117.1	3.019
	Best Offline Tuned (#4)	95.45	2.396
	Fully Tuned Refined	**84.63**	**2.186**
UDC	Baseline	267.5	8.752
	Fully Tuned Refined	**184.1**	**6.362**
EUDC	Baseline	515.0	8.397
	Fully Tuned Refined	**492.4**	**6.340**
WLTC1	Baseline	1249	32.32
	Fully Tuned Refined	**1001**	**24.64**
WLTC2	Baseline	1879	24.11
	Fully Tuned Refined	**1572**	**18.42**
WLTC3	Baseline	1899	23.83
	Fully Tuned Refined	**1820**	**20.33**
WLTC4	Baseline	2154	18.44
	Bast Offline Tuned (#10)	1597	18.66
	Fully Tuned Refined	**1818**	**14.08**

6.5 Summary

In this chapter, we specialised the OO framework to gain-scheduled MPC tuning, which was applied and experimented on the diesel air-path. It was demonstrated that by following an offline tuning approach of selecting the best m to test (out of as many candidate controllers generated offline as possible), an acceptable level of performance was achieved on the best of the controllers in online tests. This suggests that OO is suitable as a viable offline tuning approach. Additionally, it was seen that via a combination of offline and online tuning, the controller reached a higher performance than that of either alone (i.e. better than both the manually tuned baseline and purely offline tuned controller). Hence, this demonstrates that OO offline tuning is complementary, rather than a substitute, to online tuning.

This chapter ends on an epistemological note, which is that although (6.6) suggests that the success probability is maximised by selecting the largest possible sample sizes, computation of the actual success probability itself is usually not possible in practice, since the distribution (6.15) is typically not known. This also makes verifying the regularity Assumption 6.1 difficult. Information about the distribution could be gained by sampling from the joint controller and test performances, but this is itself contradictory at the outset, since access to samples from the physical plant are initially not possible within the context of offline controller tuning. We partially address some of these limitations in the next chapter, by considering a framework that allows for sampling from the induced distribution.

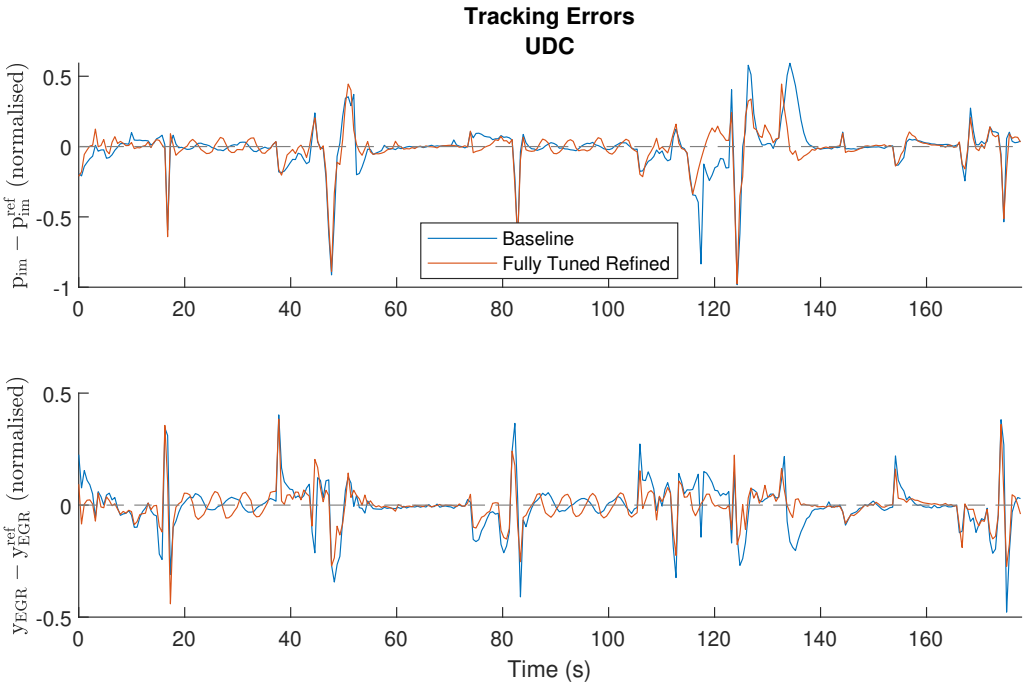


Figure 6.9: A comparison of tracking errors of the final tuned controller against the baseline for the UDC.

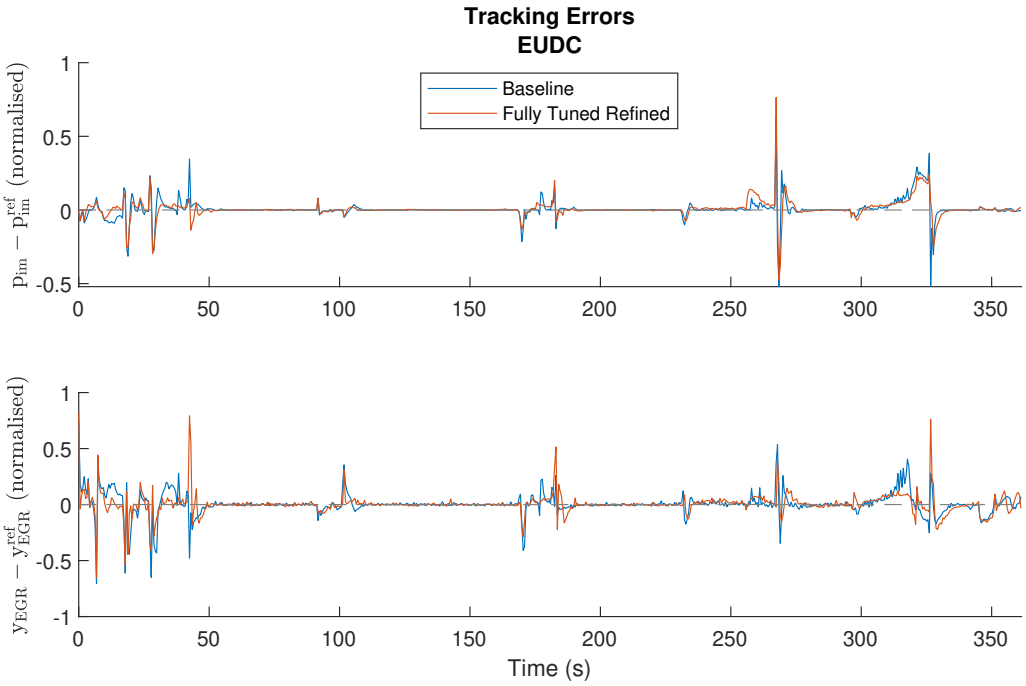


Figure 6.10: A comparison of tracking errors of the final tuned controller against the baseline for the EUDC.

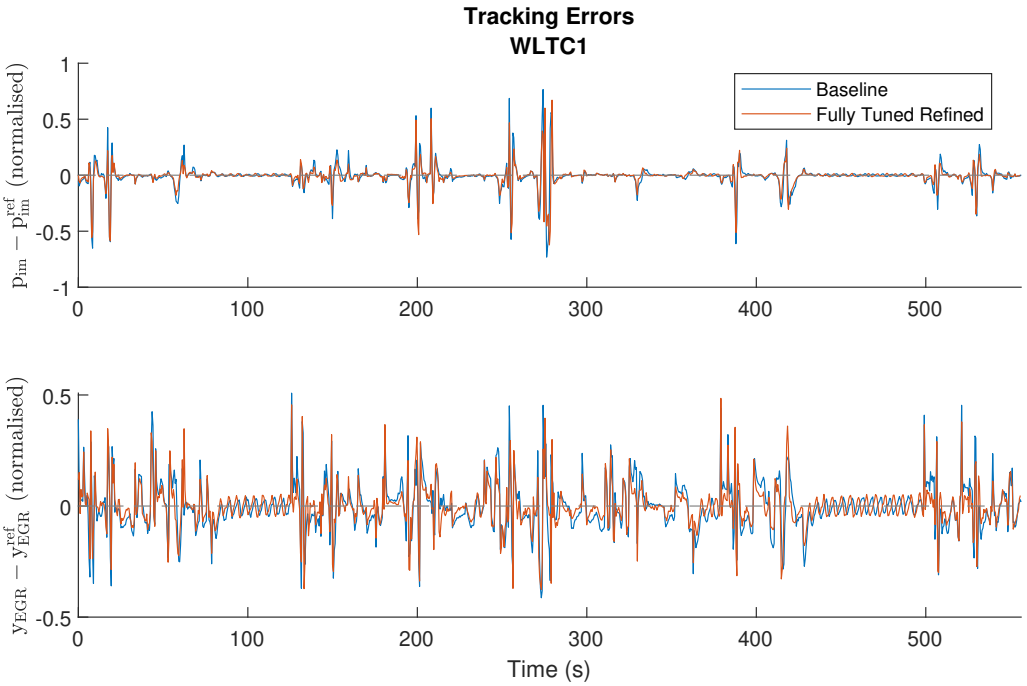


Figure 6.11: A comparison of tracking errors of the final tuned controller against the baseline for the first section of the WLTC.

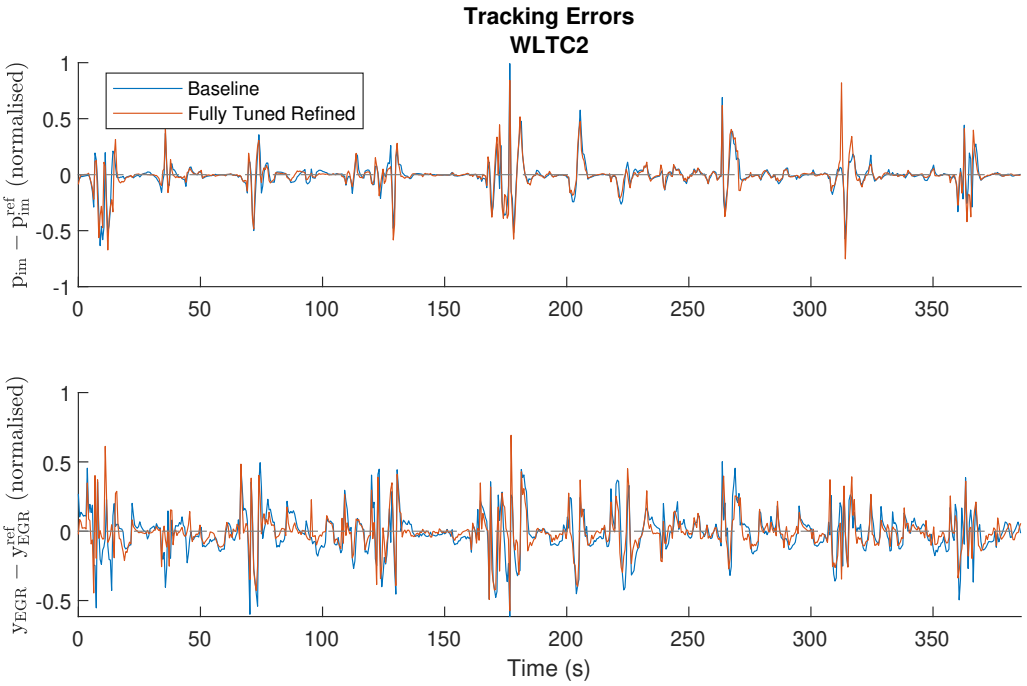


Figure 6.12: A comparison of tracking errors of the final tuned controller against the baseline for the second section of the WLTC.

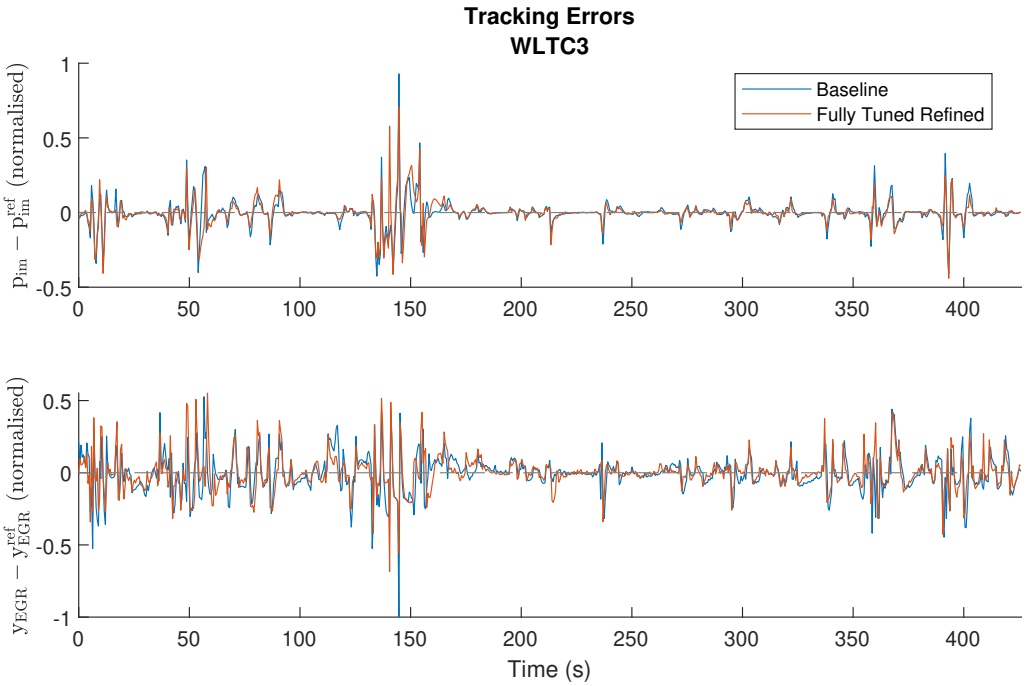


Figure 6.13: A comparison of tracking errors of the final tuned controller against the baseline for the third section of the WLTC.

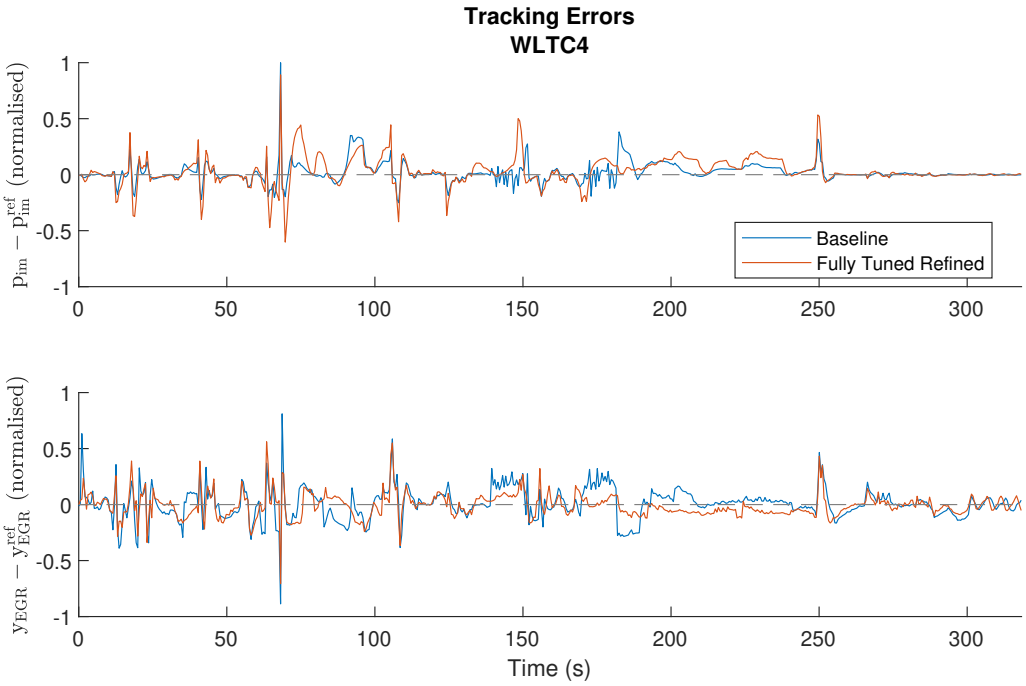


Figure 6.14: A comparison of tracking errors of the final tuned controller against the baseline for the fourth section of the WLTC.

Chapter 7

A Sequential Learning Algorithm for Probabilistically Robust Controller Tuning

In this chapter, we introduce a sequential learning algorithm to address a probabilistically robust offline controller tuning problem. The algorithm leverages ideas from the areas of randomised algorithms and ordinal optimisation. In Section 7.3, we formally prove that our algorithm yields a controller which meets a specified probabilistic performance specification, assuming a Gaussian or near-Gaussian copula model for the controller performances. Additionally, we are able to characterise the computational requirement of the algorithm by using a lower bound on the distribution function of the algorithm's stopping time. To validate our work, the algorithm is then demonstrated for the purpose of tuning model predictive controllers on a diesel engine air-path in Section 7.4. It is shown in a simulation study that the algorithm is able to successfully tune a single controller offline to meet a desired performance threshold, even in the presence of uncertainty in the diesel engine model (that is inherent when a single representation is used across a fleet of vehicles). It is also demonstrated in the simulation study that the offline tuned controller serves as an advantageous initial condition that 'hot-starts' online tuning approaches.

Contributions from this chapter also appear in the preprint publication [44].

7.1 Problem Setup

Consider a measurable *system performance function*

$$J(\psi, \theta) : \Psi \times \Theta \to \mathbb{R}, \tag{7.1}$$

with controller tuning variable $\theta \in \Theta$ from an uncountable topological space Θ, and uncertain plant parameter $\psi \in \Psi$ from a topological space Ψ. This function gives the performance of controller θ on plant ψ, for some designated control task. The uncertainty over ψ is represented by some probability distribution $\mathcal{P}_\psi$ over Ψ.

Unlike the previous chapter, where success probabilities cannot be computed unless the induced distributions of the performance functions are fully known, our motivation here is to prescribe a high probability and find a single controller θ^* so that tested the system will perform 'well' with at least the prescribed probability. To this end, suppose again that there is a mechanism $\mathcal{P}_\theta$ to randomly sample a candidate controller $\theta_i \in \Theta$. Then to evaluate candidate controllers, re-introduce the controller performance function

$$\overline{J}(\theta) : \Theta \to \mathbb{R}, \tag{7.2}$$

whose role is such that in order to find θ^*, we first sample n i.i.d. $\theta_i \sim \mathcal{P}_\theta$ and let

$$\theta^* = \underset{\theta_i \in \{\theta_1, \ldots, \theta_n\}}{\operatorname{argmin}} \overline{J}(\theta_i). \tag{7.3}$$

As a concrete example for $\overline{J}(\theta)$, we could take $\overline{J}(\theta) = J\left(\overline{\psi}, \theta\right)$ for some nominal value $\overline{\psi}$, such as $\overline{\psi} = \mathbb{E}_{\mathcal{P}_\psi}[\psi]$. An alternative example is to take $\overline{J}(\theta) = \mathbb{E}_{\mathcal{P}_\psi}[J(\psi, \theta)]$, supposing this expectation can be evaluated.

Once θ^* has been obtained, a single 'test' of the system yields the performance $J(\psi^*, \theta^*)$, from an independently realised plant $\psi^* \sim \mathcal{P}_\psi$. Unlike the previous chapter, exogenous noise is not explicitly considered in this test performance, however a noise term can be absorbed into the variable ψ^* if desired, since only a single test performance is being conducted. This test performance naturally predicates on how well the two random variables $\overline{J}(\theta_i)$ and $J(\psi^*, \theta_i)$ are correlated, via their dependence on θ_i. A strong correlation should suggest that well-performing $\overline{J}(\theta_i)$ is highly indicative of well-performing $J(\psi^*, \theta_i)$, thus we would reasonably anticipate the test $J(\psi^*, \theta^*)$ to also perform well.

7.1.1 Copula Modelling

To formalise the concept of dependence between $\overline{J}(\theta_i)$ and $J(\psi^*, \theta_i)$, we use again use copulae, like in Chapter 5. A well-defined notion of correlation valid for any bivariate distribution is the Kendall correlation.

Definition 7.1 (Population Kendall correlation)**.** *For a bivariate distribution (Z, X), the population Kendall correlation is defined as*

$$\kappa = \mathbb{E}\left[\operatorname{sign}\left(Z - \grave{Z}\right)\operatorname{sign}\left(X - \grave{X}\right)\right], \tag{7.4}$$

where $\left(\grave{Z}, \grave{X}\right)$ is an independent copy of (Z, X).

In this chapter, it will be convenient to associate every bivariate distribution with a bivariate Gaussian copula, which we do so through the Kendall correlation.

Definition 7.2 (Associated Gaussian copula)**.** *For any bivariate distribution (Z, X) with population Kendall correlation κ, the Gaussian copula associated with this distribution is defined as the bivariate Gaussian copula with correlation $\rho = \sin(\pi\kappa/2)$.*

The formula $\rho = \sin(\pi\kappa/2)$ is from the relation between κ and ρ for a Gaussian copula [111, Equation (9.11)]. As such, any bivariate distribution with a Gaussian copula has its own copula as the associated Gaussian copula.

7.1.2 Problem Statement

We are ready to list the standing assumptions of the chapter, for which the main results rely on.

Assumption 7.1. *The bivariate distribution for the performances $\left(\overline{J}(\theta_i), J(\psi^*, \theta_i)\right)$ is continuous, and has population Kendall correlation $\kappa > 0$, however the value of κ itself is unknown.*

Remark 7.1. *The positive likelihood ratio dependence condition from Definition 5.3 is strong enough to imply positive Kendall correlation (see [145, (5.2.17) and Theorem 5.2.20]). Also by*

Sklar's theorem [109, Theorem 1.1], the continuity property in Assumption 7.1 ensures that the distribution of $\left(\overline{J}\left(\theta_i\right), J\left(\psi^*, \theta_i\right)\right)$ *has a unique copula.*

We also assume the following bound between the copula of $\left(\overline{J}\left(\theta_i\right), J\left(\psi^*, \theta_i\right)\right)$, and its associated Gaussian copula.

Assumption 7.2. *Let* $\left(\widetilde{Z}, \widetilde{X}\right)$ *denote the copula of* $\left(\overline{J}\left(\theta_i\right), J\left(\psi^*, \theta_i\right)\right)$ *and let* $\left(\widetilde{Z}, \widetilde{X}'\right)$ *denote the Gaussian copula associated with* $\left(\overline{J}\left(\theta_i\right), J\left(\psi^*, \theta_i\right)\right)$. *For a given* $\nu \in [0, 1)$, *then for all* $z \in (0, 1)$ *we have*

$$\sup_{x \in (0,1)} \left\{ \Pr\left(\widetilde{X}' \leq x \middle| \widetilde{Z} = z\right) - \Pr\left(\widetilde{X} \leq x \middle| \widetilde{Z} = z\right) \right\} \leq \nu. \tag{7.5}$$

Remark 7.2. *The condition* (7.5) *in Assumption 7.2 is saying that the copula of* $\left(\overline{J}\left(\theta_i\right), J\left(\psi^*, \theta_i\right)\right)$ *is not too unfavourably 'far' from that of a Gaussian copula, which is slighter weaker than saying that the copula is 'near' to a Gaussian copula. To elaborate further, a given bound on the Kolmogorov-Smirnov distance (i.e. supremum norm) or total variation distance [85, §5.9] between* $\Pr\left(\widetilde{X}' \leq x \middle| \widetilde{Z} = z\right)$ *and* $\Pr\left(\widetilde{X} \leq x \middle| \widetilde{Z} = z\right)$ *will imply* (7.5). *Moreover, if* $\left(\overline{J}\left(\theta_i\right), J\left(\psi^*, \theta_i\right)\right)$ *is assumed to have a Gaussian copula, then* (7.5) *is satisfied with* $\nu = 0$.

Now let $J^* \in \mathbb{R}$ denote a nominal performance threshold, which is used to benchmark the test performance $J\left(\psi^*, \theta^*\right)$. We require this nominal performance threshold to be feasible, in the following sense.

Assumption 7.3. *The nominal performance threshold* J^* *satisfies*

$$\Pr_{\psi^*, \theta_i}\left(J\left(\psi^*, \theta_i\right) \leq J^*\right) > 0. \tag{7.6}$$

Lastly, we can forego exact knowledge about the distributions of $\mathcal{P}_\psi$, $\mathcal{P}_\theta$, but the standing assumption is that they can at the very least be sampled from (e.g. via a computer simulation).

Assumption 7.4. *Samples can be drawn i.i.d. from the distributions* $\mathcal{P}_\psi$ *and* $\mathcal{P}_\theta$.

As a consequence, we can produce an i.i.d. sample from the distribution of $\left(\overline{J}\left(\theta_i\right), J\left(\psi^*, \theta_i\right)\right)$, which we denote

$$\left(\overline{J}\left(\theta_1\right), J\left(\psi_1, \theta_1\right)\right), \ldots, \left(\overline{J}\left(\theta_n\right), J\left(\psi_n, \theta_n\right)\right). \tag{7.7}$$

We may now state the main problem addressed in this chapter.

Problem 7.1. *Suppose Assumptions 7.1, 7.2, 7.3 and 7.4 hold. Given $\gamma \in (\nu, 1]$ and nominal performance threshold $J^* \in \mathbb{R}$, find a controller θ^* such that*

$$\Pr_{\psi^*, \theta^*}\left(J\left(\psi^*, \theta^*\right) \leq J^*\right) \geq 1 - \gamma. \tag{7.8}$$

In Section 7.3, we propose Algorithm 7.2 to address this problem, with the formal statement contained in Theorem 7.3.

7.2 Success Probability Lower Confidence Bound

Problem 7.1 can be framed in the context of ordinal optimisation (OO), by taking

$$\left(Z_i, X_i\right) \underset{\text{st}}{=} \left(\overline{J}\left(\theta_i\right), J\left(\psi^*, \theta_i\right)\right) \tag{7.9}$$

in Definition 5.3 with $m = 1$. However, a value for $\alpha = \Pr_{\psi^*, \theta_i}\left(J\left(\psi^*, \theta_i\right) \leq J^*\right)$ is not explicitly given in Problem 7.1, nor can the analytical lower bound in Theorem 5.7 be readily applied since (Z, X) may generally not have a Gaussian copula. In this section, we overcome these obstacles by developing a lower confidence bound (LCB) for the OO success probability. This is to be derived from LCBs for α, and for ρ, the latter being the correlation of the associated Gaussian copula.

To facilitate this, we will work more abstractly with a continuous bivariate distribution for (Z, X) as in Chapter 5, with Kendall correlation $\kappa > 0$, and its associated Gaussian copula correlation ρ. It is to be kept in mind that we can take (7.9) to bring the context back into controller tuning. Also, the standing assumptions can be stated in an

analogous way for the distribution (Z, X). In particular, the analogy to Assumption 7.4 is to let an i.i.d. sample of size n be denoted by

$$(Z_1, X_1), \ldots, (Z_n, X_n) \underset{\text{st}}{=} \left(\overline{J}(\theta_1), J(\psi_1, \theta_1)\right), \ldots, \left(\overline{J}(\theta_n), J(\psi_n, \theta_n)\right). \tag{7.10}$$

First, we consider the following point estimators for α and ρ.

Definition 7.3 (Point estimator for α)**.** *From the sample* (7.10)*, a point estimate of* $\alpha = \Pr(X \leq x^*)$ *for performance threshold* x^* *is*

$$\widehat{\alpha}_n := \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}_{\{X_i \leq x^*\}}. \tag{7.11}$$

Definition 7.4 (Point estimator for ρ)**.** *From the sample* (7.10)*, a point estimate of the correlation* ρ *for the associated Gaussian copula is*

$$\widehat{\rho}_n := \sin\left(\frac{\pi}{2} \max\{0, \widehat{\kappa}_n\}\right), \tag{7.12}$$

where $\widehat{\kappa}_n$ *is the sample Kendall correlation*

$$\widehat{\kappa}_n := \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j=1}^{n} \operatorname{sign}\left((X_i - X_j)(Z_i - Z_j)\right). \tag{7.13}$$

Confidence bounds for α and ρ can be obtained from the following concentration inequalities.

Lemma 7.1 (Concentration inequalities for α)**.** *For* $a > 0$*, we have*

$$\Pr(\widehat{\alpha}_n - \alpha < -a) \leq \exp\left(-2na^2\right) \tag{7.14}$$

$$\Pr(\widehat{\alpha}_n - \alpha > a) \leq \exp\left(-2na^2\right). \tag{7.15}$$

Proof. Recognising that $n\widehat{\alpha}_n$ is a sum of independent Bernoulli random variables (each bounded between 0 and 1) with mean α, use Hoeffding's inequality [63, Theorem 1.1] to

obtain

$$\Pr\left(\widehat{\alpha}_n - \alpha > a\right) = \Pr\left(n\widehat{\alpha}_n - n\alpha > na\right) \tag{7.16}$$

$$\leq \exp\left(-2na^2\right), \tag{7.17}$$

and analogously for the lower tail bound.

□

Lemma 7.2 (Concentration inequalities for ρ)**.** *Under Assumption 7.1 with the substitution* (7.9)*, for $r > 0$, we have*

$$\Pr\left(\widehat{\rho}_n - \rho < -r\right) \leq \exp\left(-\left\lfloor\frac{n}{2}\right\rfloor\frac{2r^2}{\pi^2}\right) \tag{7.18}$$

$$\Pr\left(\widehat{\rho}_n - \rho > r\right) \leq \exp\left(-\left\lfloor\frac{n}{2}\right\rfloor\frac{2r^2}{\pi^2}\right). \tag{7.19}$$

Proof. Contained in Appendix C.1.

□

Remark 7.3. *A two-tailed bound similar to Lemma 7.2 with a slightly different exponent can be found in [124, Theorem 4.2]. Applying the fact that $n/4 \leq \lfloor n/2 \rfloor$ for all $n > 1$, one can eliminate the floor operator in* (7.18)*,* (7.19) *and recover the same exponent as found in [124].*

From the upper tailed concentration inequalities for α and ρ, we may then derive LCBs. To derive a LCB for α with confidence level at least $1-\beta_1$, equate $\exp\left(-2na^2\right) = \beta_1$ and rearrange in the upper-tailed bound (7.15) to obtain

$$\Pr\left(\widehat{\alpha}_n - \alpha > \sqrt{\frac{\log\left(1/\beta_1\right)}{2n}}\right) \leq \beta_1. \tag{7.20}$$

Let

$$b_1 := \sqrt{\frac{\log\left(1/\beta_1\right)}{2n}}, \tag{7.21}$$

so that

$$\Pr\left(\alpha > \widehat{\alpha}_n - b_1\right) \geq 1 - \beta_1. \tag{7.22}$$

Thus the LCB for α with confidence at least $1-\beta_1$ is obtained as

$$\underline{\widehat{\alpha}}_n := \widehat{\alpha}_n - b_1. \tag{7.23}$$

To derive a LCB for ρ with confidence level at least $1-\beta_2$, equate $\exp\left(-\lfloor n/2 \rfloor\, 2r^2/\pi^2\right) = \beta_2$ and rearrange in the upper-tailed bound (7.19) to obtain

$$\Pr\left(\widehat{\rho}_n - \rho > \pi\sqrt{\log\left(\frac{1}{\beta_2}\right)\cdot\frac{1}{2\left\lfloor\frac{n}{2}\right\rfloor}}\right) \leq \beta_2. \tag{7.24}$$

Let

$$b_2 := \pi\sqrt{\log\left(\frac{1}{\beta_2}\right)\cdot\frac{1}{2\left\lfloor\frac{n}{2}\right\rfloor}}, \tag{7.25}$$

so that

$$\Pr\left(\rho > \widehat{\rho}_n - b_2\right) \geq 1-\beta_2. \tag{7.26}$$

Thus the LCB for ρ with confidence at least $1-\beta_2$ is obtained as

$$\underline{\widehat{\rho}}_n := \widehat{\rho}_n - b_2. \tag{7.27}$$

We may also bound the difference in the success probability from that of its associated Gaussian copula.

Lemma 7.3 (Difference in OO success probability)**.** *Consider the OO success probability* (5.3) *from Definition 5.2, and let ρ be the correlation of the associated Gaussian copula. If Assumption 7.2 holds under the substitution* (7.9)*, then for all $n \in \mathbb{N}$ and $\alpha \in (0,1]$ we have*

$$p^{\mathcal{N}}_{\text{success}}(n,1,\alpha,\rho) - p_{\text{success}}(n,1,\alpha) \leq \nu. \tag{7.28}$$

Proof. Let $\left(\widetilde{Z},\widetilde{X}\right)$ denote the copula of (Z,X) and let $\left(\widetilde{Z},\widetilde{X}'\right)$ denote the associated Gaussian copula, where the marginal $\widetilde{Z}$ can be shared since it is a $\mathrm{Uniform}(0,1)$ random variable. Using the fact that the first order statistic of an i.i.d. $\mathrm{Uniform}(0,1)$ sample is $\mathrm{Beta}(1,n)$ distributed [12, §1.1], and recognising that the OO success probability only

depends on the underlying copula of the distribution, we have in the case $m = 1$ that

$$p_{\text{success}}(n, 1, \alpha) = \int_0^1 \Pr\left(\widetilde{X} \leq \alpha \middle| \widetilde{Z} = z\right) f_{U_{1:n}}(z)\,\mathrm{d}z, \tag{7.29}$$

where $f_{U_{1:n}}(\cdot)$ is the density of the $\mathrm{Beta}(1, n)$ distribution. Likewise

$$p^{\mathcal{N}}_{\text{success}}(n, 1, \alpha, \rho) = \int_0^1 \Pr\left(\widetilde{X}' \leq \alpha \middle| \widetilde{Z} = z\right) f_{U_{1:n}}(z)\,\mathrm{d}z. \tag{7.30}$$

The difference between these is

$$\begin{aligned}
&p^{\mathcal{N}}_{\text{success}}(n, 1, \alpha, \rho) - p_{\text{success}}(n, 1, \alpha) && (7.31)\\
&= \int_0^1 \left(\Pr\left(\widetilde{X}' \leq \alpha \middle| \widetilde{Z} = z\right) - \Pr\left(\widetilde{X} \leq \alpha \middle| \widetilde{Z} = z\right)\right) f_{U_{1:n}}(z)\,\mathrm{d}z && (7.32)\\
&\leq \int_0^1 \sup_{\alpha \in (0,1]} \left\{\Pr\left(\widetilde{X}' \leq \alpha \middle| \widetilde{Z} = z\right) - \Pr\left(\widetilde{X} \leq \alpha \middle| \widetilde{Z} = z\right)\right\} f_{U_{1:n}}(z)\,\mathrm{d}z && (7.33)\\
&\leq \nu \int_0^1 f_{U_{1:n}}(z)\,\mathrm{d}z && (7.34)\\
&= \nu, && (7.35)
\end{aligned}$$

where the second inequality is from (7.5) in Assumption 7.2. □

Using Lemma 7.3, the aforementioned properties on α and ρ, as well as the lower bound for p_{success} in (5.46), we are ready to establish a LCB on the OO success probability.

Theorem 7.1 (Lower confidence bound for p_{success})**.** *Consider the OO success probability from Definition 5.2. If Assumption 7.2 holds under the substitution* (7.9)*, then from the sample* (7.10)*, with confidence at least* $1 - \beta_1 - \beta_2$*, we have*

$$\underline{p}^{\mathcal{N}}_{\text{success}}\left(n, 1, \widehat{\underline{\alpha}}_n, \widehat{\underline{\rho}}_n\right) - \nu \leq p_{\text{success}}(n, m, \alpha). \tag{7.36}$$

Proof. As $\underline{p}^{\mathcal{N}}_{\text{success}}$ from (5.46) is a lower bound, then

$$\underline{p}^{\mathcal{N}}_{\text{success}}\left(n, 1, \widehat{\underline{\alpha}}_n, \widehat{\underline{\rho}}_n\right) \leq p^{\mathcal{N}}_{\text{success}}\left(n, 1, \widehat{\underline{\alpha}}_n, \widehat{\underline{\rho}}_n\right). \tag{7.37}$$

Applying Lemma 7.3 (which requires Assumption 7.2), this implies

$$\underline{p}_{\text{success}}\left(n, 1, \widehat{\underline{\alpha}}_n, \widehat{\underline{\rho}}_n\right) - \nu \leq p'_{\text{success}}\left(n, 1, \widehat{\underline{\alpha}}_n, \widehat{\underline{\rho}}_n\right). \tag{7.38}$$

Using the property of monotonicity in m from Theorem 5.3(b), we have

$$p_{\text{success}}(n, 1, \alpha) \leq p_{\text{success}}(n, m, \alpha). \tag{7.39}$$

Therefore

$$\begin{aligned}
&\Pr\left(\underline{p}^{\mathcal{N}}_{\text{success}}\left(n, 1, \widehat{\underline{\alpha}}_n, \widehat{\underline{\rho}}_n\right) - \nu \leq p_{\text{success}}(n, m, \alpha)\right) && (7.40)\\
&\geq \Pr\left(\underline{p}^{\mathcal{N}}_{\text{success}}\left(n, 1, \widehat{\underline{\alpha}}_n, \widehat{\underline{\rho}}_n\right) - \nu \leq p_{\text{success}}(n, 1, \alpha)\right) && (7.41)\\
&\geq \Pr\left(\underline{p}^{\mathcal{N}}_{\text{success}}\left(n, 1, \widehat{\underline{\alpha}}_n, \widehat{\underline{\rho}}_n\right) \leq p^{\mathcal{N}}_{\text{success}}(n, 1, \alpha, \rho)\right) && (7.42)\\
&\geq \Pr\left(p^{\mathcal{N}}_{\text{success}}\left(n, 1, \widehat{\underline{\alpha}}_n, \widehat{\underline{\rho}}_n\right) \leq p^{\mathcal{N}}_{\text{success}}(n, 1, \alpha, \rho)\right) && (7.43)\\
&\geq \Pr\left(\widehat{\underline{\alpha}}_n \leq \alpha, \widehat{\underline{\rho}}_n \leq \rho\right) && (7.44)\\
&= 1 - \Pr\left(\widehat{\underline{\alpha}}_n > \alpha \text{ or } \widehat{\underline{\rho}}_n > \rho\right) && (7.45)\\
&\geq 1 - \Pr\left(\widehat{\underline{\alpha}}_n > \alpha\right) - \Pr\left(\widehat{\underline{\rho}}_n > \rho\right) && (7.46)\\
&\geq 1 - \beta_1 - \beta_2, && (7.47)
\end{aligned}$$

where the first inequality is from applying (7.39), the second inequality is due to the implication (7.38), the third inequality is from (7.37), the fourth inequality is by applying the monotonicity properties from Theorems 5.3(c) and 5.6, the fifth inequality is by the union bound (Boole's inequality), and the last inequality is from the LCB properties (7.22), (7.23), (7.26), (7.27).

□

Remark 7.4. *A $1 - \beta_1 - \beta_2$ LCB for the OO success probability under the associated Gaussian copula is $\underline{p}^{\mathcal{N}}_{\text{success}}\left(n, 1, \widehat{\underline{\alpha}}_n, \widehat{\underline{\rho}}_n\right)$, i.e.*

$$\Pr\left(\underline{p}^{\mathcal{N}}_{\text{success}}\left(n, 1, \widehat{\underline{\alpha}}_n, \widehat{\underline{\rho}}_n\right) \leq p^{\mathcal{N}}_{\text{success}}(n, 1, \alpha, \rho)\right) \geq 1 - \beta_1 - \beta_2. \tag{7.48}$$

7.3 Sequential Learning Algorithm

In view of Remark 7.4, we present Algorithm 7.1, which sequentially draws samples from (Z, X) and stops after a random τ samples until an associated Gaussian copula OO success probability of at least $1 - \delta$ is reached, with confidence of at least $1 - \beta_1 - \beta_2$. Note that this algorithm works irrespective of the value of ν in Assumption 7.2, because the algorithm considers only the associated Gaussian copula.

Our algorithm uses a stopping rule, so that the sample complexity is not known in advance, but rather is a random variable induced by the randomness over each run of the algorithm. As the decision of whether to stop is learned from the algorithm by drawing sequential samples, we refer to our algorithm as a sequential learning algorithm. Another sequential learning algorithm also appeared in [115], which built upon the work of [195] with less conservative sample complexities. Their algorithm is based on the Rademacher bootstrap technique. Stopping rules in RA were also studied in [70] for designing linear quadratic regulators, while [7] investigated another class of sequential algorithms. A stopping rule is also considered by [21] for solving stochastic programs, in which the algorithm stops when the computed confidence widths of estimated quantities become sufficiently small; this is similar to the nature of our algorithm.

Algorithm 7.1 Sequential learning for Gaussian copula OO success probability

Require: $\delta \in (0,1]$, $\beta_1 \in (0,1]$, $\beta_2 \in (0,1]$, performance threshold x^*, initial sample (7.10) of size n

1: $n \leftarrow n+1$
2: Independently sample (Z, X) and add to existing samples
3: Compute $\widehat{\alpha}_n$, $\widehat{\rho}_n$ via (7.11), (7.12), (7.13)
4: Compute $\underline{\widehat{\alpha}}_n$, $\underline{\widehat{\rho}}_n$ via (7.23), (7.27) using β_1, β_2 respectively
5: $p \leftarrow \underline{p}^{\mathcal{N}}_{\text{success}}\left(n, 1, \underline{\widehat{\alpha}}_n, \underline{\widehat{\rho}}_n\right)$
6: If $p \geq 1-\delta$, continue, otherwise go to step 1
7: $\tau \leftarrow n$

Qualitatively, as n increases, the confidence widths b_1 and b_2 decrease to zero. The lower bound from Theorem 5.7 also stipulates that $\underline{p}^{\mathcal{N}}_{\text{success}}$ is increasing in n. Thus, we intuitively reason that Algorithm 7.1 eventually terminates with sufficiently large n. This intuition can be made precise with the following theorem and subsequent corollary, which uses the concentration inequalities for α and ρ to bound the distribution of the time at which Algorithm 7.1 stops.

Theorem 7.2 (Bound on stopping time). *Fix δ, β_1, β_2 in Algorithm 7.1. Given some $n \in \mathbb{N}$, suppose the pair (α^*, ρ^*) satisfies $\underline{p}^{\mathcal{N}}_{\text{success}}(n, 1, \alpha^*, \rho^*) \geq 1-\delta$. Also let*

$$a := \alpha_0 - \alpha^* \tag{7.49}$$

$$r := \rho_0 - \rho^*, \tag{7.50}$$

where α_0, ρ_0 are the actual values of α, ρ respectively. Then, for all n greater than the initial sample size, we have

$$\Pr(\tau \leq n) \geq 1 - \exp\left(-2n\left(\alpha_0 - \alpha^* - b_1\right)^2\right) - \exp\left(-\left\lfloor \frac{n}{2} \right\rfloor \frac{2\left(\rho_0 - \rho^* - b_2\right)^2}{\pi^2}\right), \tag{7.51}$$

provided $\alpha_0 - \alpha^* - b_1 > 0$ and $\rho_0 - \rho^* - b_2 > 0$.

Proof. We may bound

$$\Pr(\tau \leq n) \geq \Pr\left(\underline{p}^{\mathcal{N}}_{\text{success}}\left(n, 1, \widehat{\underline{\alpha}}_n, \widehat{\underline{\rho}}_n\right) \geq 1 - \delta\right) \tag{7.52}$$

$$\geq \Pr\left(\widehat{\underline{\alpha}}_n \geq \alpha^*, \widehat{\underline{\rho}}_n \geq \rho^*\right) \tag{7.53}$$

$$= 1 - \Pr\left(\widehat{\underline{\alpha}}_n < \alpha^* \text{ or } \widehat{\underline{\rho}}_n < \rho^*\right) \tag{7.54}$$

$$\geq 1 - \Pr\left(\widehat{\underline{\alpha}}_n < \alpha^*\right) - \Pr\left(\widehat{\underline{\rho}}_n < \rho^*\right) \tag{7.55}$$

$$= 1 - \Pr\left(\widehat{\underline{\alpha}}_n - \alpha_0 < -a\right) - \Pr\left(\widehat{\underline{\rho}}_n - \rho_0 < -r\right) \tag{7.56}$$

$$= 1 - \Pr\left(\widehat{\alpha}_n - \alpha < -a + b_1\right) - \Pr\left(\widehat{\rho}_n - \rho < -r + b_2\right) \tag{7.57}$$

$$\geq 1 - \exp\left(-2n\left(a - b_1\right)^2\right) - \exp\left(-2\left\lfloor\frac{n}{2}\right\rfloor \frac{2\left(r - b_2\right)^2}{\pi^2}\right), \tag{7.58}$$

where the first inequality holds because of the stopping condition, the second inequality is by definition of α^* and ρ^* along with monotonicity properties from Theorems 5.3(c) and 5.6, the third inequality is from the union bound (Boole's inequality), and the fourth equality is by application of the lower tailed concentration inequalities (7.14), (7.18) from Lemmas 7.1 and 7.2 respectively. Substituting (7.49), (7.50) completes the proof.

□

Corollary 7.1 (Finite stopping time)**.** *Under Assumptions 7.1 and 7.3 with the substitution* (7.9)*, the stopping time τ from Algorithm 7.1 satisfies*

$$\Pr(\tau < \infty) = 1. \tag{7.59}$$

Proof. Assumptions 7.1 and 7.3 ensure that $\alpha_0 > 0$ and $\rho_0 > 0$. By Theorem 5.3(a) and (5.23), for any $\delta > 0$ there exists a pair (α^*, ρ^*) such that $\alpha_0 - \alpha^* - b_1 > 0$ and $\rho_0 - \rho^* - b_2 > 0$ for all n greater than some sufficiently large number. Hence from the monotone convergence theorem [38, Theorem 4.8], we have

$$\Pr(\tau < \infty) = \lim_{n \to \infty} \Pr(\tau < n + 1) \tag{7.60}$$

$$= \lim_{n\to\infty} \Pr(\tau \leq n) \tag{7.61}$$

$$\geq \lim_{n\to\infty} \left[1 - \exp\left(-2n\left(\alpha_0 - \alpha^* - b_1\right)^2\right) - \exp\left(-\left\lfloor\frac{n}{2}\right\rfloor \frac{2\left(\rho_0 - \rho^* - b_2\right)^2}{\pi^2}\right)\right] \tag{7.62}$$

$$= 1, \tag{7.63}$$

where the inequality is by applying Theorem 7.2. □

Remark 7.5 (Optimised bound on stopping time)**.** *We can also numerically optimise the bound* (7.51) *with respect to* (α^*, ρ^*)*. This can be useful for characterising the computational requirement (i.e. number of samples needing to be simulated) of the algorithm. Further details on optimising the bound are provided in Appendix C.2.*

7.3.1 Controller Tuning Algorithm

Next, we specialise Algorithm 7.1 to the context of controller tuning, in order to explicitly address Problem 7.1. This is presented in Algorithm 7.2, which now also outputs the tuned controller θ_τ^*. A simplified flowchart is presented in Figure 7.1.

Algorithm 7.2 Probabilistically robust controller tuning

Require: $\delta \in (0,1]$, $\beta_1 \in (0,1]$, $\beta_2 \in (0,1]$, performance threshold J^*, initial sample (7.7) of size n

1: $n \leftarrow n+1$
2: Independently sample θ_i and ψ_i
3: Form $(Z_i, X_i) \leftarrow \left(\overline{J}(\theta_i), J(\psi_i, \theta_i)\right)$ and add to existing samples
4: Compute $\widehat{\alpha}_n$ via (7.11) with performance threshold J^*
5: Compute $\widehat{\rho}_n$ via (7.12), (7.13)
6: Compute $\underline{\widehat{\alpha}}_n$, $\underline{\widehat{\rho}}_n$ via (7.23), (7.27) using β_1, β_2 respectively
7: $p \leftarrow \underline{p}^{\mathcal{N}}_{\text{success}}\left(n, 1, \underline{\widehat{\alpha}}_n, \underline{\widehat{\rho}}_n\right)$
8: If $p \geq 1-\delta$, continue, otherwise go to step 1
9: $\tau \leftarrow n$
10: $\theta_\tau^* \leftarrow \operatorname{argmin}_{\theta_i \in \{\theta_1,\ldots,\theta_\tau\}} \overline{J}(\theta_i)$

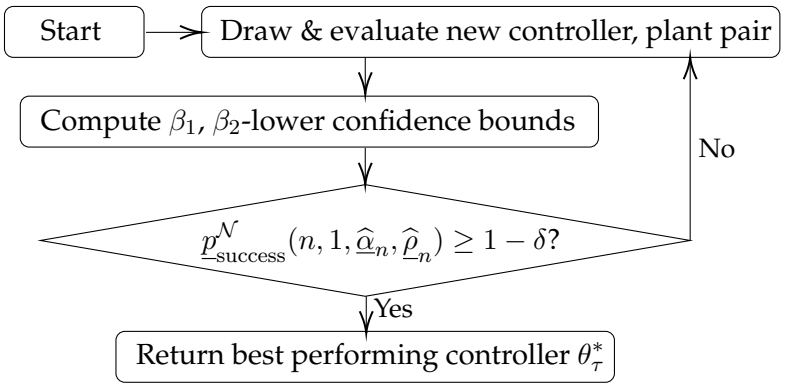


Figure 7.1: A simplified flowchart depiction of Algorithm 7.2.

By chaining the confidence level with the OO success probability, we demonstrate how Algorithm 7.2 addresses Problem 7.1, via the following theorem.

Theorem 7.3. *Suppose Algorithm 7.2 is applied to tuning controllers of a system with a performance function $J(\psi, \theta)$. Let θ^*_τ denote the candidate solution output by the algorithm. Under Assumptions 7.1, 7.2, 7.3 and 7.4, then given any $\gamma \in (\nu, 1]$ and $\delta > 0$, $\beta_1 > 0$, $\beta_2 > 0$ with $\delta + \beta_1 + \beta_2 = \gamma - \nu$, then*

$$\Pr_{\psi^*, \theta^*_\tau}\left(J(\psi^*, \theta^*_\tau) \leq J^*\right) \geq 1 - \gamma. \tag{7.64}$$

Proof. Let $\widetilde{n}_\delta(\alpha, \rho)$ denote the smallest integer n such that $p^{\mathcal{N}}_{\text{success}}(n, 1, \alpha, \rho) \geq 1 - \delta$. Combining this with Lemma 7.3 (requiring Assumption 7.2), we have

$$\Pr_{\psi^*, \theta^*_\tau}\left(J(\psi^*, \theta^*_\tau) \leq J^* | \tau \geq \widetilde{n}_\delta(\alpha_0, \rho_0)\right) \geq 1 - \delta - \nu. \tag{7.65}$$

Recognise that for any τ such that $p^{\mathcal{N}}_{\text{success}}(\tau, 1, \alpha, \rho) \geq 1 - \delta$, this implies

$$\tau \geq \widetilde{n}_\delta(\alpha, \rho), \tag{7.66}$$

by definition of $\widetilde{n}_\delta(\alpha, \rho)$ and due to monotonicity in n (Theorem 5.3(a)) for the Gaussian copula. As noted in Corollary 7.1 (requiring Assumptions 7.1 and 7.3), the algorithm

stops at time τ with probability one such that

$$\underline{p}^{\mathcal{N}}_{\text{success}}\left(\tau, 1, \widehat{\underline{\alpha}}_\tau, \widehat{\underline{\rho}}_\tau\right) \geq 1 - \delta. \tag{7.67}$$

Then by letting $\beta = \beta_1 + \beta_2$, we have

$$\begin{aligned}
1 - \beta_1 - \beta_2 = 1 - \beta & \tag{7.68}\\
&\leq \Pr\left(\underline{p}^{\mathcal{N}}_{\text{success}}\left(\tau, 1, \widehat{\underline{\alpha}}_\tau, \widehat{\underline{\rho}}_\tau\right) \leq p^{\mathcal{N}}_{\text{success}}\left(\tau, 1, \alpha_0, \rho_0\right)\right) & \tag{7.69}\\
&\leq \Pr\left(1 - \delta \leq p^{\mathcal{N}}_{\text{success}}\left(\tau, 1, \alpha_0, \rho_0\right)\right) & \tag{7.70}\\
&\leq \Pr\left(\tau \geq \widetilde{n}_\delta\left(\alpha_0, \rho_0\right)\right), & \tag{7.71}
\end{aligned}$$

where the first inequality is via Remark 7.4, the second inequality is due to the stopping condition (7.67), and the third inequality is due to the implication (7.66). Thus

$$\begin{aligned}
\Pr\left(J\left(\psi^*, \theta^*_\tau\right) \leq J^*\right) &\geq \Pr\left(J\left(\psi^*, \theta^*_\tau\right) \leq J^*, \tau \geq \widetilde{n}_\delta\left(\alpha_0, \rho_0\right)\right) & \tag{7.72}\\
&= \Pr\left(J\left(\psi^*, \theta^*_\tau\right) \leq J^* | \tau \geq \widetilde{n}_\delta\left(\alpha_0, \rho_0\right)\right) \Pr\left(\tau \geq \widetilde{n}_\delta\left(\alpha_0, \rho_0\right)\right) & \tag{7.73}\\
&\geq \left(1 - \delta - \nu\right)\left(1 - \beta\right) & \tag{7.74}\\
&> 1 - \delta - \nu - \beta & \tag{7.75}\\
&= 1 - \gamma, & \tag{7.76}
\end{aligned}$$

because δ, β_1, β_2 were chosen such that

$$\gamma = \delta + \beta_1 + \beta_2 + \nu. \tag{7.77}$$

□

Remark 7.6. *An explicit choice of algorithm settings may, for instance, be* $\delta = \beta_1 = \beta_2 = (\gamma - \nu)/3$, *or* $\delta = (\gamma - \nu)/2$, $\beta_1 = \beta_2 = (\gamma - \nu)/4$. *The lower bound on the stopping time in Theorem 7.2 will generally depend on* δ, β_1, β_2, *so by choosing an appropriate combination of* δ, β_1, β_2 *in a practice known as risk allocation [150], the performance of the algorithm can potentially be improved. However, doing so would not be reasonable in practice since it also requires the actual*

values of α and ρ to be known.

Remark 7.7. *If $\left(\overline{J}\left(\theta_i\right), J\left(\psi^*, \theta_i\right)\right)$ is assumed to have a Gaussian copula, we may take $\nu = 0$ as per Remark 7.2, so $\Pr_{\psi^*,\theta^*_\tau}\left(J\left(\psi^*, \theta^*_\tau\right) \leq J^*\right)$ can be made arbitrarily close to one. This is because the lower tail boundary conditional CDF for the bivariate Gaussian copula is degenerate at zero for $\rho > 0$ (as discussed in Section 5.3), so in the expression* (7.30) *for the OO success probability, $\lim_{n\to\infty} p^{\mathcal{N}}_{\text{success}}\left(n, 1, \alpha, \rho\right) = 1$. However, there exist families of bivariate copulae where the lower tail boundary conditional CDF is non-degenerate (e.g. the bivariate Frank copula mentioned in Section 5.2), so in the expression* (7.29)*, generally $\lim_{n\to\infty} p_{\text{success}}\left(n, 1, \alpha\right) \neq 1$. Therefore in the case $\nu > 0$, we generally cannot make $\Pr_{\psi^*,\theta^*_\tau}\left(J\left(\psi^*, \theta^*_\tau\right) \leq J^*\right)$ arbitrarily close to one.*

Remark 7.8. *Although the value of ν is treated as given in Theorem 7.3, this might not be explicitly known a priori in practice, and instead must be assumed. However, once the algorithm has finished running, an a posteriori value for ν can be estimated from the collected sample. This is demonstrated in the numerical example.*

7.4 Numerical Example

We demonstrate our proposed approach on a numerical example, of tuning MPC offline for the diesel air-path. We shall consider two particular applications for the tuned controller.

1. As mentioned in Section 2.1.2.3, offline tuning and online tuning methods may complement each other. The experimental tuning results in Section 6.4 and Figures 6.5, 6.5 demonstrated that a combination of offline and online tuning could outperform either along. In this application, Algorithm 7.2 will be used to tune an MPC offline, to investigate the extent to which the tuned controller yields a good initial point for online tuning.

2. Typically in production, a single controller will be tuned for a fleet of vehicles. However, the exact model representing any individual engine dynamics may differ

slightly. We apply Algorithm 7.2 to tuning engine controllers so that the performance will be robust to these variations.

Here, the individual engine is modelled as a nominal linear process model

$$x_{k+1} = \overline{A}x_k + \overline{B}u_k, \tag{7.78}$$

with the nominal matrices $\overline{A} \in \mathbb{R}^{n\times n}$, $\overline{B} \in \mathbb{R}^{n\times m}$ subjected to a Gaussian disturbance

$$A = \overline{A} + S_A \odot Z_{n\times n} \tag{7.79}$$

$$B = \overline{B} + S_B \odot Z_{n\times m}, \tag{7.80}$$

where $Z_{n\times m}$ denotes an $n \times m$ matrix of independent standard Gaussian entries, while S_A, S_B contain the standard deviations of the disturbances. These disturbances model the variations of different vehicles in the fleet.

7.4.1 System Description

The air-path of an automotive diesel engine can be locally represented by a reduced order linear model with 4 states, and 3 inputs [172]. The state vector is denoted by

$$x = \begin{bmatrix} p_{\mathrm{im}} & p_{\mathrm{em}} & W_{\mathrm{comp}} & y_{\mathrm{EGR}} \end{bmatrix}^\top, \tag{7.81}$$

where p_{im} is the engine intake manifold pressure, p_{em} is the exhaust manifold pressure, W_{comp} is the compressor mass flow rate and y_{EGR} is the known as the exhaust gas recirculation (EGR) rate. The inputs are composed of the actuation signals

$$u = \begin{bmatrix} u_{\mathrm{thr}} & u_{\mathrm{EGR}} & u_{\mathrm{VGT}} \end{bmatrix}^\top, \tag{7.82}$$

where u_{thr} is for the throttle valve, u_{EGR} is for the EGR valve and u_{VGT} is for the variable geometry turbine (VGT) vane. At a particular operating condition, the nominal model (7.78) with normalised units and variables trimmed from the equilibrium point has been

obtained as

$$\overline{\mathsf{A}} = \begin{bmatrix} 0.9846 & -0.0003 & 0.0010 & 0.0088 \\ 0.0066 & 0.9847 & 0.0058 & 0.0021 \\ -0.0108 & -0.0276 & 1.0186 & 0.1280 \\ 0.2116 & 0.1107 & -0.2596 & 0.1946 \end{bmatrix} \tag{7.83}$$

$$\overline{\mathsf{B}} = \begin{bmatrix} -0.0006 & -0.0002 & 0.0276 \\ -0.0001 & -0.0001 & 0.0284 \\ 0.0025 & -0.0012 & -0.1508 \\ 0.0123 & 0.0003 & 0.2997 \end{bmatrix}. \tag{7.84}$$

We consider the task of regulating the states to the origin from the initial condition

$$\mathsf{x}_0 = \begin{bmatrix} -0.7298 & -0.6931 & -1.2875 & 0.1051 \end{bmatrix}^\top, \tag{7.85}$$

which constitutes a step change from a 'medium' engine operating condition to a 'high' engine operating condition. Let the system performance function for our regulation problem be defined as

$$J(\psi, \theta) = \mathtt{ST}\left(\mathbf{Y}_{\psi,\theta}\right), \tag{7.86}$$

where $\mathbf{Y}_{\psi,\theta}$ is the discrete-time closed-loop trajectory of $\mathrm{y_{EGR}}$ under controller θ on plant ψ, and $\mathtt{ST}(\cdot)$ is the 2% settling time. This settling time function is implemented in MATLAB, which linearly interpolates in between discrete-time trajectories, and is thus continuous-valued. Here, the MPC law is obtained by solving the receding horizon quadratic-cost problem

$$\begin{aligned} \min_{\mathsf{u}_{k|0},\ldots,\mathsf{u}_{k|9}} & \left\{ \sum_{i=0}^{9} \left(\mathsf{x}_{k|i}^\top \mathsf{Q} \mathsf{x}_{k|i} + \mathsf{u}_{k|i}^\top \mathsf{R} \mathsf{u}_{k|i} \right) + \mathsf{x}_{k|10}^\top \mathsf{P} \mathsf{x}_{k|10} \right\} \\ \text{subject to } & \mathsf{x}_{k|i+1} = \overline{\mathsf{A}} \mathsf{x}_{k|i} + \overline{\mathsf{B}} \mathsf{u}_{k|i},\ i = 0, \ldots, 9 \\ & \mathsf{M} \mathsf{x}_{k|i} \leq \mathsf{f},\ i = 1, \ldots, 10 \\ & \mathsf{E} \mathsf{u}_{k|i} \leq \mathsf{h},\ i = 0, \ldots, 9 \\ & \left| \mathsf{u}_{k|i} - \mathsf{u}_{k|i-1} \right| \leq \mathsf{u}_{\mathrm{slew}},\ i = 0, \ldots, 9 \end{aligned} \tag{7.87}$$

where $\mathsf{Q} \succ \mathbf{0}$, $\mathsf{P} \succ \mathbf{0}$, $\mathsf{R} \succ \mathbf{0}$, and with state constraint values (representing physical constraints on the signals):

$$\mathsf{M} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 \end{bmatrix} \tag{7.88}$$

$$\mathsf{f} = \begin{bmatrix} 1.2568 & 0.8296 & 0.3704 \end{bmatrix}^{\top}, \tag{7.89}$$

input constraint values

$$\mathsf{E} = \begin{bmatrix} I_{3\times 3} \\ -I_{3\times 3} \end{bmatrix} \tag{7.90}$$

$$\mathsf{h} = \begin{bmatrix} 0.15 & 0.15 & 0.15 & 0 & 0.15 & 0.15 \end{bmatrix}, \tag{7.91}$$

and slew rate

$$\mathsf{u}_{\text{slew}} = 0.05, \tag{7.92}$$

while the initial input begins at $\mathsf{u}_0 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^{\top}$. At state x_k for $k \geq 1$, the optimal solution to (7.87) with $\mathsf{x}_{k|0} = \mathsf{x}_k$ is obtained as $\left(\mathsf{u}^*_{k|0}, \ldots, \mathsf{u}^*_{k|9}\right)$, and the control law applied at time k is $\mathsf{u}_k = \mathsf{u}^*_{k|0}$.

The controller tuning variables are the positive definite cost matrices

$$\theta = (\mathsf{Q}, \mathsf{P}, \mathsf{R}). \tag{7.93}$$

The mechanism we use for randomly generating the Q, R matrices, using a similar approach to [103, 128], is given by

$$\mathsf{Q} = \mathsf{W}_{\mathsf{Q}} \mathsf{D}_{\mathsf{Q}} \mathsf{W}_{\mathsf{Q}}^{\top} \tag{7.94}$$

$$\mathsf{R} = \mathsf{W}_{\mathsf{R}} \mathsf{D}_{\mathsf{R}} \mathsf{W}_{\mathsf{R}}^{\top}, \tag{7.95}$$

where

- $\mathsf{W_Q}$ is a uniformly random orthogonal matrix of dimension 4×4,
- $\mathsf{W_R}$ is a uniformly random orthogonal matrix of dimension 3×3,
- $\mathsf{D_Q}$ is a diagonal matrix whose diagonal elements are independently $\mathrm{Exp}\,(1)$ distributed,
- $\mathsf{D_R}$ is a diagonal matrix whose diagonal elements are independently $\mathrm{Exp}\,(1/1000)$ distributed.

Then P is fixed with respect to $\overline{\mathsf{A}}, \overline{\mathsf{B}}, \mathsf{Q}, \mathsf{R}$ by solving the discrete-time algebraic Riccati equation.

Uncertainty quantification for the diesel engine air-path has been performed using the methodology detailed in Chapter 4, which we assume for the purpose of this example represents the uncertainty over a fleet of vehicles. Each random plant is characterised by the pair

$$\psi = (\mathsf{A}, \mathsf{B})\,, \tag{7.96}$$

and ψ can be sampled by perturbing the nominal model (7.78) with small noise. The standard deviations from (7.79), (7.80) are

$$\mathsf{S_A} = \begin{bmatrix} 6.095 & 4.775 & 3.610 & 9.502 \\ 0.6014 & 0.4692 & 0.3555 & 0.9324 \\ 20.85 & 16.56 & 12.85 & 31.097 \\ 44.38 & 34.46 & 26.26 & 64.57 \end{bmatrix} \times 10^{-4} \tag{7.97}$$

$$\mathsf{S_B} = \begin{bmatrix} 5.287 & 3.753 & 4.9861 \\ 0.5261 & 0.3705 & 0.4910 \\ 17.70 & 12.31 & 16.62 \\ 37.98 & 26.26 & 34.82 \end{bmatrix} \times 10^{-4}. \tag{7.98}$$

Let the performance comparison function $\overline{J}\,(\theta)$ be the settling time from using the nomi-

nal model $\overline{\psi} = \left(\overline{\mathsf{A}}, \overline{\mathsf{B}}\right)$ as the plant dynamics in closed-loop under controller θ, i.e.

$$\overline{J}(\theta) = \mathtt{ST}\left(\mathbf{Y}_{\overline{\psi},\theta}\right). \tag{7.99}$$

We also set a desired nominal performance threshold of $J^* = 6.75$ seconds for the settling time.

7.4.2 Single Tuned Controller

Assume a value for $\nu = 0.1$. By running Algorithm 7.2 with settings $\delta = 0.1, \beta_1 = 0.05, \beta_2 = 0.05$, then from Theorem 7.3 the prescribed probability of the nominal performance threshold being met in a single test is at least $1 - \gamma = 0.7$. We ran the algorithm once, which stopped after $\tau = 1544$ samples. A histogram for the performances $\overline{J}(\theta_i)$ and $J(\psi_i, \theta_i)$ are plotted in Figure 7.2.

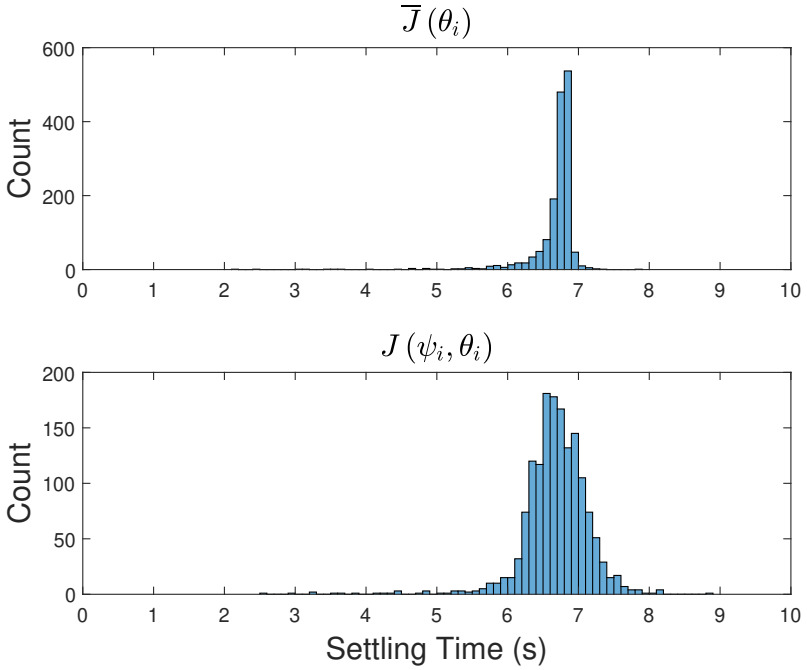


Figure 7.2: Histograms of $\overline{J}(\theta_i)$ and $J(\psi_i, \theta_i)$ for the 1544 samples in a single run of Algorithm 7.2.

The best controller θ^*_τ when evaluated on the performance comparison function $\overline{J}(\theta)$

was found to have a settling time of $\overline{J}(\theta_\tau^*) = 2.1482$ seconds. Upon simulating a test of this tuned controller using another randomly generated plant ψ^*, we obtained a trajectory for y_{EGR} with a settling time of 2.7148 seconds, which far outperforms the nominal threshold $J^* = 6.75$.

7.4.2.1 Application to Online Tuning

We conduct a simulation of a mock online tuning example, using the offline tuned controller θ_τ^* as an initial point. The online optimiser used was a variant of Nesterov's gradient free algorithm [146] tailored towards positive definite matrices, which has previously been applied in online tuning of MPC [128]. Evaluations were performed using the objective $J(\psi^*, \cdot)$, with the realised plant ψ^* held constant. Each iteration of the algorithm requires two function evaluations (i.e. two experiments per function evaluation in the context of online tuning).

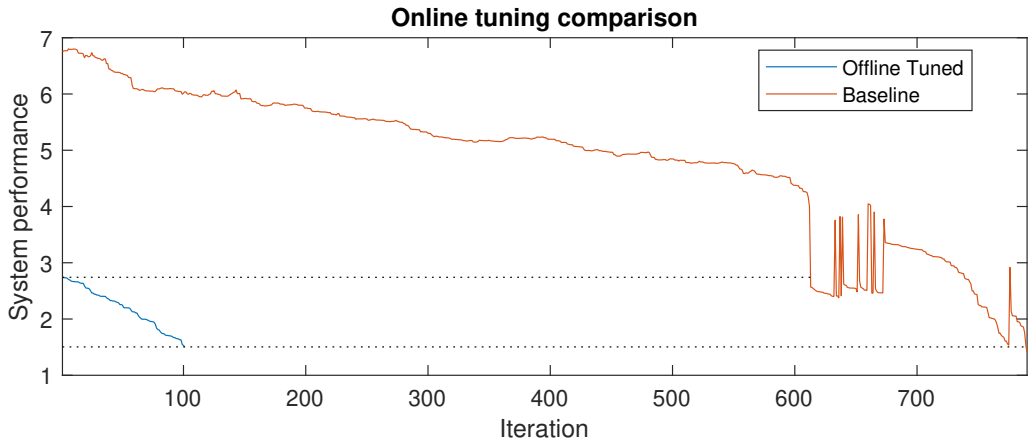


Figure 7.3: Iterations of the system performance in the online tuning example.

Figure 7.3 shows the iterations of the online tuning, compared against that of another baseline controller with initial performance roughly equal to the threshold $J^* = 6.75$. Since the behaviour of the algorithm is sensitive to the online optimiser settings (e.g. step size sequence), the step size sequence in each example has been chosen favourably in order to provide a fairer comparison of optimistic online tuning performance. As the comparison shows, the improvement in performance approximately follows a linear

trend with similar slope. It takes roughly 600 iterations for the baseline controller to reach the initial level of the offline tuned controller, while it takes almost 800 iterations for it to reach the same level that it took the offline tuned controller 100 iterations to reach. This demonstrates how a well-performing controller obtained by offline tuning can complement online tuning methods, by providing good initial controllers to 'hot-start' the online tuning method, and reduce the amount of online tuning to reach a given performance.

7.4.2.2 Application to Tuning for a Fleet of Vehicles

To investigate the tuned controller performance on a fleet of vehicles, we simulated 10000 tests on another set of independently generated plants, with the same tuned controller. By the linearity of expectation, Theorem 7.3 prescribes that the expected proportion of tests which outperform $J^* = 6.75$ to be at least $1 - \gamma = 0.7$. We found that all 10000 of the tests outperformed the nominal threshold, which far exceeds 0.7. Moreover, the minimum, mean and maximum performance times were of 0.9359, 2.4826 and 4.8024 seconds respectively.

7.4.3 Multiple Tuned Controllers

Aggregate results were also obtained for 1000 independent runs of Algorithm 7.2 with identical tuning procedure and settings as described above, producing 1000 tuned controllers. Each controller was then tested on its own randomly generated plant. It was found that all 1000 tests succeeded in outperforming the nominal threshold of $J^* = 6.75$ seconds, with minimum, mean and maximum performance times of 0.5381, 2.7407 and 4.8867 seconds respectively. Note that the distribution of these 1000 tests is different from that of the 10000 tests in previous section, as each test here consists of a different controller.

7.4.4 Discussion

For this example, we may validate Theorem 7.2 by plotting in Figure 7.4 the numerically optimised lower bound for the CDF of the stopping time τ (using the point estimates $\widehat{\alpha}_\tau = 0.5654$, $\widehat{\rho}_\tau = 0.4907$ from the sample in Figure 7.2, in place of the actual α_0, ρ_0), against the empirical CDF of the stopping time for the 1000 runs.

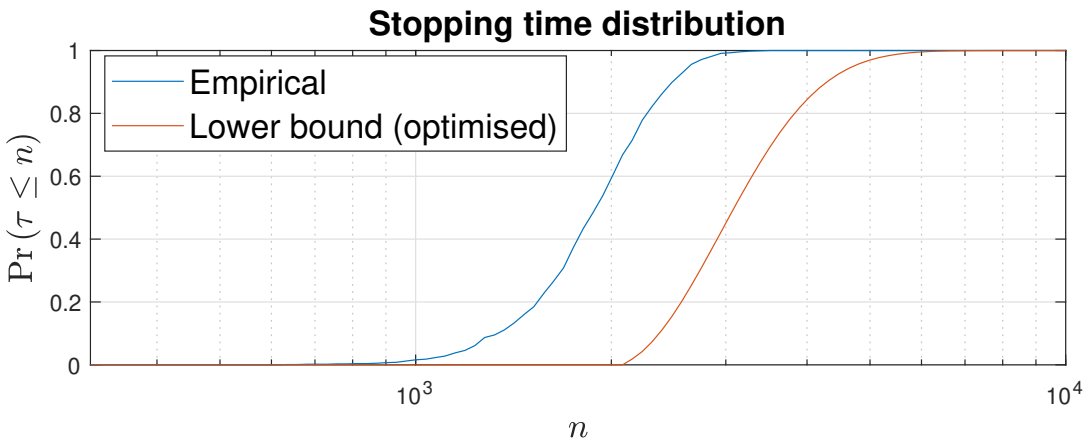


Figure 7.4: The empirical algorithm run times compared against those as suggested by Theorem 7.2.

As the curves in Figure 7.4 are within less than order of magnitude on the horizontal scale, this hints that Theorem 7.2 is not overly conservative. However, our empirical results also suggest that $1 - \gamma$ can be conservative for the actual probability of outperforming J^* (which appears to be much closer to 1 than 0.7). This is partly attributed to the 'price' of needing to estimate α and ρ, since the LCBs (7.23), (7.27) are non-asymptotic, which tend to be more conservative than their asymptotic counterparts. This conservativeness can somewhat be reduced in absolute terms by choosing a lower γ. For instance, one could select $\delta = 0.025$, $\beta_1 = 0.0125$ and $\beta_2 = 0.0125$ such that $\gamma = 0.15$. However, this comes at a trade-off of longer stopping times, hence more computation. Using point estimates in place of the actual α and ρ, applying Theorem 7.2 suggests that Algorithm 7.2 will have stopped by $n = 40000$, with probability at least 0.999. In a similar fashion, choosing $\delta = 0.005$, $\beta_1 = 0.0025$, $\beta_2 = 0.0025$ so that $\gamma = 0.11$, this value for n changes to 480000.

Decreasing J^* rather than γ also comes at a trade-off of more computation. We performed additional simulations identical to Section 7.4.2 with a single tuned controller, except with $J^* = 6.5$. After 10000 independent tests, the minimum, mean and maximum settling times were 0.7185, 2.5036 and 5.1081, which are arguably not improved over the results from setting $J^* = 6.75$. However, this run of Algorithm 7.2 took 37144 samples. Using point estimates in place of the actual α and ρ, applying Theorem 7.2 suggests that Algorithm 7.2 will have stopped by $n = 6800$ with probability at least 0.999 for $J^* = 6.75$ (evident from Figure 7.4). On the other hand with $J^* = 6.5$, Theorem 7.2 suggests that Algorithm 7.2 will have stopped by $n = 93000$, with probability at least 0.999. Therefore while the indicative value for J^* does seem to be conservative of the performance in this example, decreasing J^* by a small amount does not appear to improve the test performance, but results in an order of magnitude increase in the computation time.

From the relatively large sample of size 37144 obtained in the previous paragraph, we can also numerically validate the assumption that $\nu = 0.1$. A nonparametric copula kernel density estimate was fitted from this sample, using the `kdecopula` package in R [144], which implements the transformation local likelihood estimation method [76]. A Gaussian copula was also fitted with the point estimate for ρ. Performing a sweep over z and x values in (7.5), we computed

$$\sup_{z,x\in(0,1)} \left\{ \Pr\left(\widetilde{X}' \leq x \middle| \widetilde{Z} = z\right) - \Pr\left(\widetilde{X} \leq x \middle| \widetilde{Z} = z\right) \right\} \approx 0.097, \tag{7.100}$$

which is consistent with the assumption of $\nu = 0.1$.

7.5 Summary

In this chapter, we addressed a probabilistically robust control design problem using a sequential learning algorithm, based on OO. Our results were enabled by relating the general OO success probability to work from Section 5.3 on the Gaussian copula OO success probability, via the associated Gaussian copula. The algorithm was illustrated on

a numerical example involving the tuning of MPC for automotive diesel engines, and showed the advantage of using the offline tuned controller as an initial condition for online tuning. Additionally, the performance of offline tuned controllers was robust to plant uncertainty in both a multi-plant setting over a fleet of vehicles with a single algorithm run, and a multi-controller setting over many algorithm runs.

Chapter 8
Conclusion

8.1 Summary of Contributions

This thesis has explored several contributions to methods of offline tuning for MPC, using techniques from machine learning and ordinal optimisation (OO). These contributions and their relation to the research aims from Section 2.6 are reiterated below.

8.1.1 Contributions to Research Aim 1

To augment the machine learning controller tuning framework with isotonic pairwise preference learning and plant uncertainty quantification.

In Chapter 3, two methods were presented for performing isotonic pairwise preference learning. The first method, detailed in Algorithm 3.1, found a weighted average between the MAP and MLE latent utility vectors $\vec{\mathtt{f}}_{\mathrm{MAP}}$ and $\vec{\mathtt{f}}_{\mathrm{lin}}$ respectively for Gaussian process regression, based on derived conditions for monotonicity. This was successfully demonstrated for learning preferences over diesel air-path trajectories in Section 3.2.5, with a bivariate feature space. In Section 3.3, another method was presented, which performed Bayesian estimation of a linear utility function $\mathtt{f}(\mathtt{x}) = \mathtt{b}^{\top}\mathtt{x}$ with a Dirichlet prior. This was also applied to diesel air-path trajectories, with an 8-dimensional feature space. Both methods augmented the machine learning controller tuning framework by supplying the performance index used to evaluate trajectories.

In Chapter 4, an active learning methodology was proposed for system identification

of linear parameter-varying systems. The resultant model, called the GPR-LPV (Gaussian Process Regression Linear Parameter-Varying) model, fits a Gaussian process over operating point $\mathfrak{p}$ for each of the elements in the $\mathsf{A}(\mathfrak{p})$ and $\mathsf{B}(\mathfrak{p})$ matrices. In addition to selecting subsequent operating points for experimentation, the resulting identified model naturally quantifies the plant uncertainty for the machine learning controller tuning framework, since the estimation procedure is based on Gaussian process regression (which inherently quantifies the variance). The method was successfully demonstrated on identification of a diesel-engine air path (with operating point $\mathfrak{p}$ being the engine speed $\mathrm{N_e}$ and fuel rate $\mathrm{w_{fuel}}$).

8.1.2 Contributions to Research Aim 2

To develop an ordinal optimisation approach valid for offline tuning of MPC, which is to be tested online.

Chapter 5 considered a copula model for the pair (Z, X), and developed theoretical results for the OO success probability p_{success}. The results are compatible with offline tuning of controllers over an uncountable space, as addressed in Section 6.2. Later within the same section, the OO methodology was specialised to offline tuning of a gain-scheduled controller architecture.

Moreover, Chapter 7 presented a sequential learning algorithm for probabilistically robust offline controller tuning, using the analytic lower bound for the Gaussian copula success probability $p^{\mathcal{N}}_{\mathrm{success}}$ derived in Section 5.3.3. The algorithm operates by drawing samples of candidate controllers and plants, to sequentially estimate properties of the underlying copula. The algorithm stops when a lower confidence bound on the success probability is estimated to be high (at least $1-\delta$), with high confidence (at least $1-\beta_1-\beta_2$). The algorithm was demonstrated in simulation on offline tuning a single MPC for the diesel air-path when: 1) the initial controller is used to hot-start an online tuning procedure; and 2) there is uncertainty across a fleet of vehicles.

8.1.3 Contributions to Research Aim 3

To implement and experimentally validate the efficacy of quadratic-cost MPC tuned offline from ordinal optimisation on a diesel engine test rig.

A gain-scheduled MPC architecture was implemented and tested at the Toyota Higashi-Fuji Center in Japan. The experiments in Chapter 6 show that by following the OO offline tuning methodolgoy, some of the offline tuned controllers were able to perform better than a baseline tuned MPC, over the UDC3 and WLTC4 drive-cycles. After further manual online tuning, the controller was then demonstrated to outperform the baseline MPC over the full UDC, EUDC, and WLTC. These experimental results provide evidence that MPC tuned offline by OO can provide an acceptable level of tuning. Moreover, offline tuning was found to complement the online tuning experiments.

8.2 Future Research

Based on the work in this thesis, the following directions are highlighted for future research.

8.2.1 Preference Learning

- Further theoretical analysis could be conducted for the pairwise isotonic learning Algorithm 3.1 in Chapter 3 to show, for example, that by enforcing monotonicity, generalisation error is reduced (compared to not enforcing monotonicity).
- Algorithm 3.1 could also be modified to scale better with the dimension $\mathtt{d}$ of the feature space, since there is currently no efficient way to find $\mathtt{a}^*$, over a brute force approach.

8.2.2 Active Learning

- The ideas from the active learning framework from Chapter 4 could be extended to other classes of parametric non-linear systems, whereby a Gaussian process is

fitted to each parameter. In the same way, the resultant identified model would be able to quantify plant uncertainty.

- Additionally, the Assumption 4.3 of full state measurement could be relaxed, so that identification must be performed from only input-output (u, y) data, rather than input-state (u, x) data as is currently done. This would perhaps involve combining subspace identification techniques [126, §10.6] with the current approach.

8.2.3 Ordinal Optimisation

- Further study could be conducted into how properties of copula dependence, and lower tail dependence affect p_{success}. A monotonicity property in the correlation ρ for a Gaussian copula was established Theorem 5.6. But since several other classes of single-parameter bivariate copulae have a parameter that influences dependence (e.g. ϖ for the Frank copula), it may also be possible to obtain an analogous property applicable to a wider class of copulae.

- Although the approximation formula in Section 5.3.2 appeared to lower bound the actual success probability, confirmation of this is currently elusive. If a lower bound is established, then this would constitute a tractable and principled way to approximate success probabilities that scale with m, and computed with $O\left(m^2\right)$ time complexity, rather than with m-dimensional integration.

- Analytic lower bounds in the style of Section 5.3.3 could potentially be developed for classes of copulae other than the Gaussian copula, which would then allow us to invert the bound as in Section 5.3.4.

- It is also worthwhile investigating an extension of the OO problem from Definition 5.2, where we relax the requirement that (Z, X) have a continuous distribution. This may allow for the OO tuning methodology to be applied to variables from a countable set, such as the prediction horizon N. One complication arising from this direction however is that the underlying copula of (Z, X) will no longer be unique, hence this will need to be carefully considered in the treatment.

8.2.4 Offline Controller Tuning Experiments

- With offline OO tuning, other MPC architectures could be experimented on, such as a robust control architecture with stability and recursive feasibility guarantees like that considered in [166].

- Other practical applications for the OO offline tuning approach could be considered, other than for control of the diesel air-path. Key application areas would share similar logistical characteristics (i.e. online experimentation is relatively scarce compared to offline simulation).

8.2.5 Sequential Learning Algorithm

- The examples in Section 7.4 illustrate Algorithm 7.2 with only simulation studies, but the algorithm could be further validated through further physical experiments.

- Currently, the results in Chapter 7 are reliant on the underlying copula being not too unfavourably far from a Gaussian copula (with the bound ν). As discussed in Section 8.2.3, development of alternative lower bounds for the OO success probability may facilitate sequential learning algorithms which relax this restriction.

- Another avenue is to investigate the role that the distribution $\mathcal{P}_\theta$ (for sampling candidate controllers) plays in the tuned controller performance. In our formulation, $\mathcal{P}_\theta$ is an arbitrary choice left to the practitioner. In Section 7.4, $\mathcal{P}_\theta$ was chosen by explicitly constructing a distribution, however another option would have been to let θ_i be the solution output by running a randomised optimisation algorithm with objective $\overline{J}(\theta)$ (similar to Chapter 6). Modifying $\mathcal{P}_\theta$ affects the distribution of $\left(\overline{J}(\theta_i), J(\psi^*, \theta_i)\right)$, and consequently the value of ν. Hence it is perhaps possible to find guiding principles in designing $\mathcal{P}_\theta$ which will lead to more favourable probabilistic performance specifications, or alternatively, reduced computational requirements for a fixed performance specification.

- The setup in Chapter 7 effectively considers a form of OO with selection size $m = 1$, i.e. only the single best offline controller is tested online. An extension to $m > 1$ is

also worth investigating, which would foremost require developing lower bounds for the success probability that favourably scale with m (as mentioned above in Section 8.2.3).

Appendix A

Gaussian Process Regression

A Gaussian process on d-variate feature variable $x \in \mathbb{R}^d$ may be defined by:

$$f(x) \sim \mathcal{GP}\left(m(x), k\left(x, x'\right)\right), \tag{A.1}$$

where $m(x) : \mathbb{R}^d \to \mathbb{R}$ is called the mean function and positive definite kernel $k(x, x') : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is also known as the covariance function; these two functions completely specify the Gaussian process. For two collections of points $X = (x_1, \ldots, x_n)$ and $X' = (x'_1, \ldots, x'_m)$, denote the *Gram matrix* as

$$K\left(X, X'\right) := \begin{bmatrix} k(x_1, x'_1) & \ldots & k(x_1, x'_m) \\ \vdots & \ddots & \vdots \\ k(x_n, x'_1) & \ldots & k(x_n, x'_m) \end{bmatrix} \tag{A.2}$$

and let

$$m(X) := \begin{bmatrix} m(x_1) & \ldots & m(x_m) \end{bmatrix}. \tag{A.3}$$

Then for a Gaussian process specified by $m(\cdot)$ and $k(\cdot, \cdot)$, the joint distribution of the process at the points in X is

$$\vec{f}(X) := \begin{bmatrix} f(x_1) & \ldots & f(x_n) \end{bmatrix}^\top \sim \mathcal{N}(m(X), K(X, X)). \tag{A.4}$$

Suppose we have some input-output training data $\mathcal{D} = \left(X, \vec{y}\right)$, subject to some Gaussian observation noise with covariance C on the outputs (if the noise is i.i.d., we would take C to be a scaled identity). Then given pre-specified prior mean and covariance functions

$\mathtt{m}(\cdot)$ and $\mathtt{k}(\cdot,\cdot)$, we may obtain the posterior predictive distribution of the process at test points $\mathtt{X}_*$, by conditioning on the data. As the prior and likelihood are both Gaussian, it follows that the posterior of the process at points $\mathtt{X}_*$ is also Gaussian, given by [155, §2.7]:

$$\left[\overrightarrow{\mathtt{f}}_* \middle| \mathtt{X}_*, \mathcal{D}\right] \sim \mathcal{N}\left(\mathbf{m}(\mathtt{X}_*) + \mathtt{K}(\mathtt{X}_*, \mathtt{X})\,\mathbf{K}^{-1}\left(\overrightarrow{\mathtt{f}} - \mathbf{m}(\mathtt{X})\right), \mathtt{K}(\mathtt{X}_*, \mathtt{X}_*) - \mathtt{K}(\mathtt{X}_*, \mathtt{X})\,\mathbf{K}^{-1}\mathtt{K}(\mathtt{X}, \mathtt{X}_*)\right), \tag{A.5}$$

where

$$\mathbf{K} := \mathtt{K}(\mathtt{X}, \mathtt{X}) + \mathtt{C}. \tag{A.6}$$

The primary computational cost incurred by GPR is the inversion of the $\mathtt{n} \times \mathtt{n}$ matrix $\mathbf{K}$, which can be tempered using methods such as the Cholesky decomposition [155, Algorithm 2.1].

Appendix B
Ordinal Optimisation

B.1 Joint Distribution of Order Statistics

Consider an i.i.d. sample of size n from a 'parent' random variable X. Then the ordered values of the sample, denoted

$$X_{1:n} \leq \cdots \leq X_{n:n} \tag{B.1}$$

are called the order statistics.

Lemma B.1 (Joint PDF of order statistics)**.** *Denote ranks $1 \leq n_1 < \cdots < n_k \leq n$. Then the joint PDF of the order statistics $X_{n_1:n}, \ldots, X_{n_k:n}$ for continuous X with parent PDF denoted $f(\cdot)$, and CDF denoted $F(\cdot)$, is*

$$f_{n_1,\ldots,n_k}(x_1,\ldots,x_k) = n!\left(\prod_{j=1}^{k} f(x_j)\right)\prod_{j=0}^{k}\left[\frac{(F(x_{j+1})-F(x_j))^{n_{j+1}-n_j-1}}{(n_{j+1}-n_j-1)!}\right]\mathbb{I}_{\{x_1\leq\cdots\leq x_k\}}, \tag{B.2}$$

where we take $x_0 := -\infty$, $x_{k+1} := \infty$, $n_0 := 0$ and $n_{k+1} := n+1$.

Proof. See [51, §2.2]. □

Lemma B.2 (Joint CDF of order statistics)**.** *Denote ranks $1 \leq n_1 < \cdots < n_k \leq n$. Then the joint CDF of the order statistics $X_{n_1:n}, \ldots, X_{n_k:n}$ for continuous X with parent CDF denoted $F(\cdot)$ is*

$$F_{n_1,\ldots,n_k}(x_1,\ldots,x_k) = \sum_{i_k=n_k}^{n}\sum_{i_{k-1}=n_{k-1}}^{i_k}\cdots\sum_{i_1=n_1}^{i_2}\frac{n!}{i_1!\,(i_2-i_1)!\times\cdots\times(n-i_k)!}\left[F(x_1)\right]^{i_1}$$

$$\times \left[F\left(x_2\right) - F\left(x_1\right)\right]^{i_2 - i_1} \times \cdots \times \left[1 - F\left(x_k\right)\right]^{n - i_k} \quad \text{(B.3)}$$

for the case $x_1 \leq \cdots \leq x_k$*. For the case we do not have* $x_1 \leq \cdots \leq x_k$*, then it holds that*

$$F_{n_1,\ldots,n_k}\left(x_1, \ldots, x_k\right) = F_{n_1,\ldots,n_k}\left(x_1^*, \ldots, x_k^*\right), \quad \text{(B.4)}$$

where

$$x_k^* := x_k \quad \text{(B.5)}$$

$$x_{k-1}^* := \min\left\{x_{k-1}, x_k^*\right\} \quad \text{(B.6)}$$

$$\vdots$$

$$x_1^* := \min\left\{x_1, x_2^*\right\}. \quad \text{(B.7)}$$

Proof. The expression (B.3) generalises naturally based on arguments provided in [51, Section 2.2]. The result (B.4) is obtained by noting that by construction $x_1^* \leq \cdots \leq x_k^*$, and the joint density in the region bounded between $(x_1^*, \ldots, x_k^*)$ and $(x_1, \ldots, x_k)$ is zero. □

B.2 Equivalent Characterisation of Multivariate Stochastic Dominance

Stochastic orderings are notions of partial orders over random elements, such as univariate random variables or random vectors. The necessary and sufficient conditions for two random vectors to satisfy the 'usual' stochastic order $\underset{\text{st}}{\preceq}$, often referred to as (multivariate) stochastic dominance, is stated in Definition 5.5, which we repeat here for convenience.

- For two random vectors $\mathbf{X}_1, \mathbf{X}_2 \in \mathbb{R}^n$, we have $\mathbf{X}_1 \underset{\text{st}}{\preceq} \mathbf{X}_2$ if and only if $\mathbb{E}\left[u\left(\mathbf{X}_1\right)\right] \leq \mathbb{E}\left[u\left(\mathbf{X}_2\right)\right]$ for all weakly increasing (i.e. non-decreasing) functions $u : \mathbb{R}^n \to \mathbb{R}$.

- Equivalently, $\mathbf{X}_1 \underset{\text{st}}{\preceq} \mathbf{X}_2$ if and only if $\Pr\left(\mathbf{X}_1 \in \mathcal{U}\right) \leq \Pr\left(\mathbf{X}_2 \in \mathcal{U}\right)$ for all upper sets $\mathcal{U}$ (an upper set may be defined as a set which satisfies $\mathbf{x}_2 \in \mathcal{U}$ for all $\mathbf{x}_2 \geq \mathbf{x}_1 \in \mathcal{U}$).

Another equivalent characterisation of multivariate stochastic dominance, found in [169, Theorem 6.B.1], involves a construction on the same probability space as follows.

Theorem B.1. *For two random vectors $\mathbf{X}_1, \mathbf{X}_2 \in \mathbb{R}^n$, we have $\mathbf{X}_1 \underset{\text{st}}{\preceq} \mathbf{X}_2$ if and only if there exist two random vectors $\mathbf{X}_1', \mathbf{X}_2'$ defined on the same probability space such that $\mathbf{X}_1' \underset{\text{st}}{=} \mathbf{X}_1$, $\mathbf{X}_2' \underset{\text{st}}{=} \mathbf{X}_2$ and $\Pr(\mathbf{X}_1' \leq \mathbf{X}_2') = 1$.*

B.3 Proofs for Chapter 5

B.3.1 Proof of Proposition 5.1

For the first causal additive representation, as Z and Y are independent, we write

$$[X|Z=z] \underset{\text{st}}{=} z + Y. \tag{B.8}$$

Thus

$$\begin{aligned}
\Pr(X > x|Z=z) &= \Pr(Z+Y > x|Z=z) && \text{(B.9)}\\
&= \Pr(Y > x-z|Z=z) && \text{(B.10)}\\
&\leq \Pr\left(Y > x-z'\middle|Z=z'\right) && \text{(B.11)}\\
&= \Pr\left(X > x\middle|Z=z'\right). && \text{(B.12)}
\end{aligned}$$

For the second causal additive representation, first denote

$$\mathcal{C}(z,x) = \Pr(Z \leq z, X \leq x). \tag{B.13}$$

By analogous arguments to the first causal additive representation, we have

$$\Pr(Z > z|X=x) \leq \Pr\left(Z > z\middle|X=x'\right) \tag{B.14}$$

for all $z, x, x' \in (0,1)$ such that $x \leq x'$. Due to exchangeability, $\mathcal{C}(z,x) = \mathcal{C}(x,z)$ and we have the copula conditional CDF (computed by [109, §2.1.3]) as

$$\Pr(Z > z | X = x) = \frac{\partial \mathcal{C}(z,x)}{\partial x} \tag{B.15}$$

$$= \frac{\partial \mathcal{C}(x,z)}{\partial x} \tag{B.16}$$

$$= \Pr(X > z | Z = x). \tag{B.17}$$

Performing the same for the right-hand side of (B.14), we get

$$\Pr(X > z | Z = x) \leq \Pr\left(X > z \middle| Z = x'\right) \tag{B.18}$$

for all $z, x, x' \in (0,1)$ such that $x \leq x'$, which is the required condition for (5.5).

B.3.2 Proof of Theorem 5.1

Let $F_X(\cdot)$ denote the CDF of X, and observe that

$$\min\left\{X_{\langle 1\rangle}, \ldots, X_{\langle m\rangle}\right\} \leq x^*_\alpha \tag{B.19}$$

if and only if

$$\min\left\{F_X\left(X_{\langle 1\rangle}\right), \ldots, F_X\left(X_{\langle m\rangle}\right)\right\} \leq \alpha. \tag{B.20}$$

Moreover, letting $F_Z(\cdot)$ denote the CDF of Z, the ordering of $F_Z(Z_1), \ldots, F_Z(Z_n)$ is unchanged, compared to $Z_1, \ldots, Z_n$. Therefore using the probability integral transform, we can assume without loss of generality that (Z, X) is a copula distribution, so that Z and X are each $\mathrm{Uniform}(0,1)$ distributed. With the law of total probability, write

$$p_{\text{success}}(n, m, \alpha)$$

$$= \int_0^1 \cdots \int_0^1 \Pr\left(\min\left\{X_{\langle 1\rangle}, \ldots, X_{\langle m\rangle}\right\} \leq \alpha \middle| Z_{1:n} = z_1, \ldots, Z_{m:n} = z_m\right)$$

$$\times f_{Z_{1:n},\ldots,Z_{m:n}}(z_1, \ldots, z_m)\, \mathrm{d}z_1 \ldots \mathrm{d}z_m, \tag{B.21}$$

where

$$f_{Z_{1:n},\ldots,Z_{m:n}}(z_1,\ldots,z_m) = \frac{n!}{(n-m)!}(1-z_m)^{n-m}\,\mathbb{I}_{\{z_1\leq\cdots\leq z_m\}} \tag{B.22}$$

is the joint PDF of the first m order statistics of $(Z_1,\ldots,Z_n)$, from Lemma B.1. Further note that since each pair (Z_i, X_i) is sampled independently, then each $X_{\langle j\rangle}$ is conditionally independent of all the variables

$$\left(X_{\langle 1\rangle},\ldots,X_{\langle j-1\rangle},X_{\langle j+1\rangle},\ldots,X_{\langle m\rangle},Z_{1:n},\ldots,Z_{(j-1):n},Z_{(j+1):n},\ldots,Z_{m:n}\right), \tag{B.23}$$

given $Z_{j:m}$. Denote the event $\mathcal{Z} = \{Z_{1:n} = z_1,\ldots,Z_{m:n} = z_m\}$ for brevity. Then

$$\begin{aligned}
\Pr\left(\min\left\{X_{\langle 1\rangle},\ldots,X_{\langle m\rangle}\right\}\leq\alpha\middle|\mathcal{Z}\right) &= 1-\Pr\left(X_{\langle 1\rangle}>\alpha,\ldots,X_{\langle m\rangle}>\alpha\middle|\mathcal{Z}\right) && \text{(B.24)}\\
&= 1-\prod_{j=1}^{m}\Pr\left(X_{\langle j\rangle}>\alpha\middle|Z_{j:n}=z_j\right) && \text{(B.25)}\\
&= 1-\prod_{j=1}^{m}\Pr\left(X>\alpha|Z=z_j\right) && \text{(B.26)}\\
&= 1-\prod_{j=1}^{m}\left(1-\Pr\left(X\leq\alpha|Z=z_j\right)\right) && \text{(B.27)}\\
&= 1-\prod_{j=1}^{m}\left(1-\mathcal{C}_{X|Z}\left(\alpha|z_j\right)\right). && \text{(B.28)}
\end{aligned}$$

Plugging (B.22) and (B.28) into (B.21), we have the result claimed.

B.3.3 Proof of Theorem 5.2

Starting from (B.21) and applying (B.28), we have

$$\begin{aligned}
p_{\text{success}}(n,m,\alpha) = \int_0^1\cdots\int_0^1 &\Pr\left(\min\left\{X_{\langle 1\rangle},\ldots,X_{\langle m\rangle}\right\}\leq\alpha\middle|Z_{1:n}=z_1,\ldots,Z_{m:n}=z_m\right)\\
&\times f_{Z_{1:n},\ldots,Z_{m:n}}(z_1,\ldots,z_m)\,\mathrm{d}z_1\ldots\mathrm{d}z_m
\end{aligned} \tag{B.29}$$

$$= 1-\mathbb{E}_{Z_{1:n},\ldots,Z_{m:n}}\left[\prod_{j=1}^{m}\Pr\left(X>\alpha|Z=Z_{j:n}\right)\right]. \tag{B.30}$$

Consider some arbitrary ordered selection of size m from $(Z_1, \ldots, Z_n)$, denoted $(Z'_1, \ldots, Z'_m)$. Then we have multivariate stochastic dominance

$$(Z_{1:n}, \ldots, Z_{m:n}) \underset{\text{st}}{\preceq} \left(Z'_1, \ldots, Z'_m\right), \tag{B.31}$$

since

$$Z_{1:n} \leq Z'_1, \quad Z_{2:n} \leq Z'_2, \quad \ldots, \quad Z_{m:n} \leq Z'_m \tag{B.32}$$

for every realisation of $(Z_1, \ldots, Z_n)$, so Theorem B.1 holds. Applying the stochastically increasing positive dependence assumption, $\prod_{j=1}^{m} \Pr(X > \alpha | Z = z_j)$ is a non-decreasing function in $(z_1, \ldots, z_m)$, so by the definition of stochastic dominance (Section B.2), we have

$$\mathbb{E}_{Z_{1:n},\ldots,Z_{m:n}} \left[\prod_{j=1}^{m} \Pr(X > \alpha | Z = Z_{j:n})\right] \leq \mathbb{E}_{Z'_1,\ldots,Z'_m} \left[\prod_{j=1}^{m} \Pr\left(X > \alpha \middle| Z = Z'_j\right)\right]. \tag{B.33}$$

Hence the success probability is maximised when $(Z'_1, \ldots, Z'_m)$ is chosen as the first m order statistics.

B.3.4 Proof of Theorem 5.3

For (a), we require the following lemma.

Lemma B.3. *Let $\mathbf{Z}_{[m]:n} := (Z_{1:n}, \ldots, Z_{m:n})$ denote the joint first m order statistics of an i.i.d. sample of size n from parent distribution Z. Then*

$$\mathbf{Z}_{[m]:(n+1)} \underset{\text{st}}{\preceq} \mathbf{Z}_{[m]:n}. \tag{B.34}$$

Proof. Consider the following construction on the same probability space. Form an i.i.d. sample of size $n+1$ from Z and take the first m order statistics. This will be equal in law to $\mathbf{Z}_{[m]:(n+1)}$. Now delete one element uniformly at random, and re-compute the first m order statistics. This will be equal in law to $\mathbf{Z}_{[m]:n}$. Moreover, for every realisation

(denoted in lowercase), we have

$$\left(z_{1:(n+1)}, \ldots, z_{m:(n+1)}\right) \leq \left(z_{1:n}, \ldots, z_{m:n}\right). \tag{B.35}$$

Therefore from the characterisation of stochastic dominance in Theorem B.1, (B.34) holds. □

Then (a) follows, via the same technique and analogous arguments as in the proof of Theorem 5.2 found in Section B.3.3.

For (b), since

$$\min\left\{X_{\langle 1\rangle}, \ldots, X_{\langle m\rangle}\right\} \leq \min\left\{X_{\langle 1\rangle}, \ldots, X_{\langle m+1\rangle}\right\}, \tag{B.36}$$

then

$$\begin{aligned} p_{\text{success}}\left(n, m, \alpha\right) &= \Pr\left(\min\left\{X_{\langle 1\rangle}, \ldots, X_{\langle m\rangle}\right\} \leq \alpha\right) && \text{(B.37)} \\ &\leq \Pr\left(\min\left\{X_{\langle 1\rangle}, \ldots, X_{\langle m+1\rangle}\right\} \leq \alpha\right) && \text{(B.38)} \\ &= p_{\text{success}}\left(n, m+1, \alpha\right). && \text{(B.39)} \end{aligned}$$

For (c), by De Morgan's laws (i.e. complement of the union is the intersection of the complements), put the definition of $p_{\text{success}}\left(n, m, \alpha\right)$ in terms of

$$p_{\text{success}}\left(n, m, \alpha\right) = 1 - \Pr\left(\bigcap_{i=1}^{m}\left\{X_{\langle i\rangle} > x_{\alpha}^{*}\right\}\right). \tag{B.40}$$

Then apply the properties that x_{α}^{*} is non-decreasing in α and $\Pr\left(\bigcap_{i=1}^{m}\left\{X_{\langle i\rangle} > x_{\alpha}^{*}\right\}\right)$ is non-increasing in x_{α}^{*}.

B.3.5 Proof of Theorem 5.4

For the lower bound, consider the success probability where the selection of size m is uniformly random without replacement from $(Z_1, \ldots, Z_n)$. This selection is identical in

law to $(Z_1, \ldots, Z_m)$, which are i.i.d. Then the success probability can be computed by

$$\Pr\left(\min\left\{X_1, \ldots, X_m\right\} \leq x_{\alpha^*}\right) = 1 - \prod_{j=1}^{m} \Pr\left(X_j > x_\alpha^*\right) \tag{B.41}$$

$$= 1 - (1-\alpha)^m . \tag{B.42}$$

Based on the arguments provided from the proof of Theorem 5.2 in Appendix B.3.3, it follows that this lower bounds p_{success}.

For the upper bound, we can use the fact

$$\min_{i \in \{1,\ldots,n\}} X_i \leq \min\left\{X_{\langle 1 \rangle}, \ldots, X_{\langle m \rangle}\right\} \tag{B.43}$$

to bound

$$p_{\text{success}}\left(n, m, \alpha\right) = \Pr\left(\min\left\{X_{\langle 1 \rangle}, \ldots, X_{\langle m \rangle}\right\} \leq x_\alpha^*\right) \tag{B.44}$$

$$\leq \Pr\left(\min_{i \in \{1,\ldots,n\}} X_i \leq x_\alpha^*\right) \tag{B.45}$$

$$= 1 - \prod_{i=1}^{n} \Pr\left(X_i > \alpha\right) \tag{B.46}$$

$$= 1 - (1-\alpha)^n . \tag{B.47}$$

B.3.6 Proof of Theorem 5.5

For cases (a), (b), (c), the results are immediate from applying the general upper and lower bounds in Theorem 5.4. For (d), assuming without loss of generality that (Z, X) is a copula and applying the property of comotonicity, then

$$p_{\text{success}}\left(n, m, \alpha\right) = \Pr\left(\min\left\{X_{\langle 1 \rangle}, \ldots, X_{\langle m \rangle}\right\} \leq \alpha\right) \tag{B.48}$$

$$= \Pr\left(\min\left\{Z_{1:n}, \ldots, Z_{m:n}\right\} \leq \alpha\right) \tag{B.49}$$

$$= \Pr\left(Z_{1:n} \leq \alpha\right) \tag{B.50}$$

$$= 1 - \prod_{i=1}^{n} \Pr\left(Z_i > \alpha\right) \tag{B.51}$$

$$= 1 - (1-\alpha)^n . \tag{B.52}$$

For (e), by independence of Z and X we have from (B.28):

$$1 - \prod_{j=1}^{m} \left(1 - \Pr\left(X \leq \alpha | Z = z_j\right)\right) = 1 - \prod_{j=1}^{m} \left(1 - \Pr\left(X \leq \alpha\right)\right) \tag{B.53}$$

$$= 1 - (1-\alpha)^m . \tag{B.54}$$

This can be taken out of the integral in (B.21) (where the integral evaluates to one), so the result follows. Case (f) follows from the lower bound in Theorem 5.4. Lastly for (g), as $n \to \infty$, the joint density (B.22) converges to the Dirac delta function in argument z_m. Considering the region of integration in the expression (5.9) where $0 \leq z_1 \leq \cdots \leq z_m$, we have that $z_m = 0$ implies $z_1 = \cdots = z_{m-1} = 0$. Therefore

$$\lim_{n\to\infty} p_{\text{success}}\left(n, m, \alpha\right) = 1 - \prod_{j=1}^{m} \left(1 - \Pr\left(X \leq \alpha | Z = 0\right)\right) \tag{B.55}$$

$$= 1 - \left(1 - \mathcal{C}_{X|Z}\left(\alpha|0\right)\right)^m . \tag{B.56}$$

B.3.7 Proof of Theorem 5.6

Let G be the random variable for the number of X-values less than or equal to the threshold x^*_α. Conditional on $G = g$, we can write the order statistics of the X-values as

$$X_{1:n} \leq \cdots \leq X_{g:n} \leq x^*_\alpha < X_{(g+1):n} \leq \cdots \leq X_{n:n}. \tag{B.57}$$

Using the additive Gaussian noise representation of the Gaussian copula in Section 5.3.1, we can treat each $X \sim \mathcal{N}\left(0, 1\right)$, while Z is formed by

$$Z = X + Y, \tag{B.58}$$

where $Y \sim \mathcal{N}\left(0, \xi^2\right)$ is independent with X, and $\xi^2 = 1/\rho^2 - 1$. Introduce the following indexing of the Z-values according to the ordering of the X-values. We denote

$$Z_{\{i\}} := X_{i:n} + Y_i \tag{B.59}$$

where the Y_i are i.i.d. $\mathcal{N}\left(0, \xi^2\right)$, since the Y-values are independent of the ordering of the X-values. Let $p^{\mathcal{N}}_{\text{success}|g}\left(n, m, \alpha, \rho\right)$ denote the conditional success probability, given $G = g$. An equivalent characterisation of the conditional success probability can be obtained from [94, Equation (2.19)]. This way, we may write

$$p^{\mathcal{N}}_{\text{success}|g}\left(n, m, \alpha, \rho\right) = \Pr\left(\min\left\{Z_{\{1\}}, \ldots, Z_{\{g\}}\right\} \leq \overset{(m)}{\min}\left\{Z_{\{g+1\}}, \ldots, Z_{\{n\}}\right\} \middle| G = g\right), \tag{B.60}$$

where $\overset{(m)}{\min}\{\cdot\}$ denotes the m^{th} smallest value of its arguments. Putting (B.59), we have

$$\begin{aligned} p^{\mathcal{N}}_{\text{success}|g}\left(n, m, \alpha, \rho\right) = \Pr\Bigg(&\min\left\{Y_1 + X_{1:n}, \ldots, Y_g + X_{g:n}\right\} && \text{(B.61)} \\ &\leq \overset{(m)}{\min}\left\{Y_{g+1} + X_{(g+1):n}, \ldots, Y_n + X_{n:n}\right\} \Bigg| G = g\Bigg) \\ = \Pr\Bigg(&\min\left\{Y_1 + \Delta X_{1:n}, \ldots, Y_g + \Delta X_{g:n}\right\} && \text{(B.62)} \\ &\leq \overset{(m)}{\min}\left\{Y_{g+1} + \Delta X_{(g+1):n}, \ldots, Y_n + \Delta X_{n:n}\right\} \Bigg| G = g\Bigg), \end{aligned}$$

where $\Delta X_{i:n} := X_{i:n} - x^*_\alpha$. Now let $\widetilde{Y}_i \sim \mathcal{N}\left(0, 1\right)$ represent a standardised random variable, so that $Y_i \underset{\text{st}}{=} \xi \widetilde{Y}_i$, and

$$\begin{aligned} p^{\mathcal{N}}_{\text{success}|g}\left(n, m, \alpha, \rho\right) = \Pr\Bigg(&\min\left\{\xi\widetilde{Y}_1 + \Delta X_{1:n}, \ldots, \xi\widetilde{Y}_g + \Delta X_{g:n}\right\} && \text{(B.63)} \\ &\leq \overset{(m)}{\min}\left\{\xi\widetilde{Y}_{g+1} + \Delta X_{(g+1):n}, \ldots, \xi\widetilde{Y}_n + \Delta X_{n:n}\right\} \Bigg| G = g\Bigg) \end{aligned}$$

$$= \Pr\left(\min\left\{\widetilde{Y}_1 + \frac{\Delta X_{1:n}}{\xi}, \ldots, \widetilde{Y}_g + \frac{\Delta X_{g:n}}{\xi}\right\}\right. \tag{B.64}$$
$$\left.\leq \overset{(m)}{\min}\left\{\widetilde{Y}_{g+1} + \frac{\Delta X_{(g+1):n}}{\xi}, \ldots, \widetilde{Y}_n + \frac{\Delta X_{n:n}}{\xi}\right\}\middle| G = g\right),$$

because $\xi > 0$. Let any fixed realisation of the random variables $\widetilde{Y}_1, \ldots, \widetilde{Y}_n, \Delta X_{1:n}, \ldots, \Delta X_{n:n}$ be denoted as $\widetilde{y}_1, \ldots, \widetilde{y}_n, \Delta x_{1:n}, \ldots, \Delta x_{n:n}$ respectively. Observe $\Delta X_{i:n} \leq 0$ for all $i \leq g$, and $\Delta X_{i:n} > 0$ for all $i > g$. So for any $\xi' < \xi$, we have

$$\widetilde{y}_i + \frac{\Delta x_{i:n}}{\xi'} < \widetilde{y}_i + \frac{\Delta x_{i:n}}{\xi}, \quad \forall i \leq g \tag{B.65}$$
$$\widetilde{y}_i + \frac{\Delta x_{i:n}}{\xi'} > \widetilde{y}_i + \frac{\Delta x_{i:n}}{\xi}, \quad \forall i > g. \tag{B.66}$$

Therefore it follows that

$$\min\left\{\widetilde{y}_1 + \frac{\Delta x_{1:n}}{\xi'}, \ldots, \widetilde{y}_g + \frac{\Delta x_{g:n}}{\xi'}\right\} \leq \min\left\{\widetilde{y}_1 + \frac{\Delta x_{1:n}}{\xi}, \ldots, \widetilde{y}_g + \frac{\Delta x_{g:n}}{\xi}\right\} \tag{B.67}$$

$$\overset{(m)}{\min}\left\{\widetilde{y}_{g+1} + \frac{\Delta x_{(g+1):n}}{\xi}, \ldots, \widetilde{y}_n + \frac{\Delta x_{n:n}}{\xi}\right\} \leq \overset{(m)}{\min}\left\{\widetilde{y}_{g+1} + \frac{\Delta x_{(g+1):n}}{\xi'}, \ldots, \widetilde{y}_n + \frac{\Delta x_{n:n}}{\xi'}\right\}. \tag{B.68}$$

Denote $\rho' = \left(\xi'^2 + 1\right)^{-1/2}$, so that $\rho < \rho'$. Then

$$p^{\mathcal{N}}_{\text{success}|g}\left(n, m, \alpha, \rho'\right) = p^{\mathcal{N}}_{\text{success}|g}\left(n, m, \alpha, \left(\xi'^2 + 1\right)^{-1/2}\right) \tag{B.69}$$
$$= \Pr\left(\min\left\{\widetilde{Y}_1 + \frac{\Delta X_{1:n}}{\xi'}, \ldots, \widetilde{Y}_g + \frac{\Delta X_{g:n}}{\xi'}\right\}\right. \tag{B.70}$$
$$\left.\leq \overset{(m)}{\min}\left\{\widetilde{Y}_{g+1} + \frac{\Delta X_{(g+1):n}}{\xi'}, \ldots, \widetilde{Y}_n + \frac{\Delta X_{n:n}}{\xi'}\right\}\middle| G = g\right)$$
$$\geq \Pr\left(\min\left\{\widetilde{Y}_1 + \frac{\Delta X_{1:n}}{\xi}, \ldots, \widetilde{Y}_g + \frac{\Delta X_{g:n}}{\xi}\right\}\right. \tag{B.71}$$
$$\left.\leq \overset{(m)}{\min}\left\{\widetilde{Y}_{g+1} + \frac{\Delta X_{(g+1):n}}{\xi}, \ldots, \widetilde{Y}_n + \frac{\Delta X_{n:n}}{\xi}\right\}\middle| G = g\right)$$
$$= p^{\mathcal{N}}_{\text{success}|g}\left(n, m, \alpha, \rho\right), \tag{B.72}$$

where the inequality is from applying (B.67) and (B.68). The random variable G is binomial distributed with parameters n, α (i.e. not affected by the value of ρ), thus

$$p^{\mathcal{N}}_{\text{success}}\left(\bar{n}, \bar{m}, \bar{\alpha}, \rho'\right) = \sum_{g=0}^{n} p^{\mathcal{N}}_{\text{success}|g}\left(\bar{n}, \bar{m}, \bar{\alpha}, \rho'\right) \Pr\left(G = g\right) \tag{B.73}$$

$$\geq \sum_{g=0}^{n} p^{\mathcal{N}}_{\text{success}|g}\left(\bar{n}, \bar{m}, \bar{\alpha}, \rho\right) \Pr\left(G = g\right) \tag{B.74}$$

$$= p^{\mathcal{N}}_{\text{success}}\left(\bar{n}, \bar{m}, \bar{\alpha}, \rho\right). \tag{B.75}$$

B.3.8 Proof of Lemma 5.1

We require the following lemma.

Lemma B.4 (Stochastic dominance of parametrised random vectors)**.** *For random vectors $\mathbf{X}$, $\mathfrak{T}$, consider the conditional distribution $[\mathbf{X}|\mathfrak{T}]$. Suppose that $[\mathbf{X}|\mathfrak{T} = \mathbf{t}_1] \underset{\text{st}}{\preceq} [\mathbf{X}|\mathfrak{T} = \mathbf{t}_2]$ whenever $\mathbf{t}_1 \leq \mathbf{t}_2$. Let $\mathbf{X}_1$ denote the variable for $\mathbf{X}$ that arises from chaining the random vector $\mathfrak{T}_1$ with $[\mathbf{X}|\mathfrak{T}_1]$, and similarly let $\mathbf{X}_2$ denote the variable for $\mathbf{X}$ that arises from chaining the random vector $\mathfrak{T}_2$ with $[\mathbf{X}|\mathfrak{T}_2]$. If $\mathfrak{T}_1 \underset{\text{st}}{\preceq} \mathfrak{T}_2$, then*

$$\mathbf{X}_1 \underset{\text{st}}{\preceq} \mathbf{X}_2. \tag{B.76}$$

Proof. For all upper sets $\mathcal{U}$, we may write using indicator variables (denoted by $\mathbb{I}$)

$$\Pr\left(\mathbf{X}_1 \in \mathcal{U}\right) = \mathbb{E}\left[\mathbb{I}_{\{\mathbf{X}_1 \in \mathcal{U}\}}\right] \tag{B.77}$$

$$= \mathbb{E}_{\mathfrak{T}_1}\left[\mathbb{E}\left[\mathbb{I}_{\{\mathbf{X} \in \mathcal{U}\}}\middle|\mathfrak{T}_1\right]\right] \tag{B.78}$$

$$= \mathbb{E}_{\mathfrak{T}_1}\left[\Pr\left(\mathbf{X} \in \mathcal{U}|\mathfrak{T}_1\right)\right] \tag{B.79}$$

$$\leq \mathbb{E}_{\mathfrak{T}_2}\left[\Pr\left(\mathbf{X} \in \mathcal{U}|\mathfrak{T}_2\right)\right] \tag{B.80}$$

$$= \Pr\left(\mathbf{X}_2 \in \mathcal{U}\right), \tag{B.81}$$

where the inequality follows by stochastic dominance (Definition 5.5), since $\Pr\left(\mathbf{X} \in \mathcal{U}|\mathbf{t}\right)$ is a weakly increasing function of $\mathbf{t}$ for all upper sets $\mathcal{U}$. □

Using the necessary and sufficient conditions for stochastic dominance of multivari-

ate Gaussians [169, Example 6.B.29], and combined with Lemma B.4 applied to the conditional distribution (5.32), since $\mathbf{Z} \underset{\text{st}}{\preceq} \widehat{\mathbf{Z}}$ it follows that $\mathbf{X} \underset{\text{st}}{\preceq} \widehat{\mathbf{X}}$. Thus

$$\widehat{p}^{\mathcal{N}}_{\text{success}}(n, m, \alpha, \rho) = \Pr\left(\min\left\{\widehat{\mathbf{X}}\right\} \leq x^*_\alpha\right) \tag{B.82}$$

$$= 1 - \Pr\left(\widehat{\mathbf{X}} > x^*_\alpha \mathbf{1}\right) \tag{B.83}$$

$$\leq 1 - \Pr\left(\mathbf{X} > x^*_\alpha \mathbf{1}\right) \tag{B.84}$$

$$= \Pr\left(\min\left\{\mathbf{X}\right\} \leq x^*_\alpha\right) \tag{B.85}$$

$$= p^{\mathcal{N}}_{\text{success}}(n, m, \alpha, \rho)\,. \tag{B.86}$$

B.3.9 Proof of Lemma 5.2

We require the following two lemmas.

Lemma B.5. *We have*

$$-p \log 4 \leq \log(1-p) \leq -p, \tag{B.87}$$

where the lower bound applies for all $p \in \left[0, \frac{1}{2}\right]$, and the upper bound applies for all $p \geq 0$.

Proof. The lower bound can be established over $p \in \left[0, \frac{1}{2}\right]$ via concavity of $\log(1-p)$, i.e. line secants lie below the graph. The upper bound can also be established over $p \geq 0$ via concavity. □

Lemma B.6. *For the Gaussian Q-function given by $Q(x) = 1 - \Phi(x)$*

$$\mathfrak{c}_1 \exp\left(-\mathfrak{c}_2 x^2\right) \leq Q(x) \leq \frac{1}{2} \exp\left(-\frac{x^2}{2}\right) \tag{B.88}$$

over $x \geq 0$, with

$$\mathfrak{c}_1 = \frac{1}{2} - \frac{\omega}{\pi} \tag{B.89}$$

$$\mathfrak{c}_2 = \frac{\cot \omega}{\pi - 2\omega} \tag{B.90}$$

for any $\omega \in \left(0, \frac{\pi}{2}\right)$.

Proof. The lower bound is due to [205, Equation (2)] and the upper bound is found in [41, Equation (5)]. □

If $\widehat{Z}_{1:n}$ stochastically dominates $Z_{1:n}$, then $\Pr\left(\widehat{Z}_{1:n} \geq z\right) \geq \Pr\left(Z_{1:n} \geq z\right)$ for all $z \in \mathbb{R}$. Or in terms of the Gaussian Q-function $Q(z) = 1 - \Phi(z)$, we require

$$Q(z)^n \leq Q\left(\frac{z - \mu_n}{\sigma_n}\right) \tag{B.91}$$

for all $z \in \mathbb{R}$, where the left-hand side can be derived with a special case of Lemma B.2. The idea is to show that this bound holds over three different intervals whose union is $\mathbb{R}$, being $(-\infty, \mu_n]$, $[\mu_n, 0]$ and $[0, \infty)$. We begin with $z \in (-\infty, \mu_n]$. Since $\mu_n \leq 0$, then via the lower bound in Lemma B.6

$$Q(z)^n \leq \left(1 - \mathfrak{c}_1 \exp\left(-\mathfrak{c}_2 z^2\right)\right)^n. \tag{B.92}$$

Since $0 \leq \mathfrak{c}_1 \exp\left(-\mathfrak{c}_2 z^2\right) \leq 1/2$, then putting $p = \mathfrak{c}_1 \exp\left(-\mathfrak{c}_2 z^2\right)$ in the upper bound from Lemma B.5, we get

$$\begin{aligned} Q(z)^n &\leq \exp\left(-n\mathfrak{c}_1 \exp\left(-\mathfrak{c}_2 z^2\right)\right) && \text{(B.93)} \\ &= \exp\left(-\exp\left(-\left(\mathfrak{c}_2 z^2 - \log\left(n\mathfrak{c}_1\right)\right)\right)\right). && \text{(B.94)} \end{aligned}$$

Now using the upper bound in Lemma B.6, we have for $z \leq \mu_n$:

$$1 - \frac{1}{2}\exp\left(-\frac{(z - \mu_n)^2}{2\sigma_n^2}\right) \leq Q\left(\frac{z - \mu_n}{\sigma_n}\right). \tag{B.95}$$

The lower bound in Lemma B.5 implies $\exp(-p\log 4) \leq 1 - p$. Applying this with $p = \frac{1}{2}\exp\left(-\frac{(z-\mu_n)^2}{2\sigma_n^2}\right)$ and after some manipulation, we arrive at

$$Q\left(\frac{z - \mu_n}{\sigma_n}\right) \geq \exp\left(-\exp\left(-\left(\frac{(z - \mu_n)^2}{2\sigma_n^2} - \log\log 2\right)\right)\right). \tag{B.96}$$

Thus a sufficient condition for $Q(z)^n \leq Q\left(\frac{z-\mu_n}{\sigma_n}\right)$ over $z \in (-\infty, \mu_n]$ is

$$\exp\left(-\exp\left(-\left(\mathfrak{c}_2 z^2 - \log\left(n\mathfrak{c}_1\right)\right)\right)\right) \leq \exp\left(-\exp\left(-\left(\frac{\left(z-\mu_n\right)^2}{2\sigma_n^2} - \log\log 2\right)\right)\right) \quad \text{(B.97)}$$

or equivalently,

$$\left(\frac{1}{\sigma_n^2} - 2\mathfrak{c}_2\right) z^2 - 2\frac{\mu_n}{\sigma_n^2} z + \frac{\mu_n^2}{\sigma_n^2} + 2\log\left(n\mathfrak{c}_1\right) - 2\log\log 2 \geq 0. \quad \text{(B.98)}$$

The roots of this quadratic are at

$$z = \frac{\mu_n \pm \sqrt{\mu_n^2 - \left(1 - 2\mathfrak{c}_2\sigma_n^2\right)\left(\mu_n^2 + 2\sigma_n^2 \log\left(n\mathfrak{c}_1\right) - 2\sigma_n^2 \log\log 2\right)}}{1 - 2\mathfrak{c}_2\sigma_n^2} \quad \text{(B.99)}$$

with discriminant Δ calculated by

$$\Delta = \sigma_n^4\left(4\mathfrak{c}_2 \log\left(n\mathfrak{c}_1\right) - 4\mathfrak{c}_2 \log\log 2\right) + \sigma_n^2\left(2\mathfrak{c}_2\mu_n^2 - 2\log\left(n\mathfrak{c}_1\right) + 2\log\log 2\right). \quad \text{(B.100)}$$

Under the same choice of ω, note $2\mathfrak{c}_2\mu_n^2 = 2\log\left(n\mathfrak{c}_1\right)$ and the discriminant becomes

$$\Delta = \sigma_n^4\left(4\mathfrak{c}_2 \log\left(n\mathfrak{c}_1\right) - 4\mathfrak{c}_2 \log\log 2\right) + \sigma_n^2\left(2\log\log 2\right). \quad \text{(B.101)}$$

The quadratic inequality is satisfied everywhere if the discriminant is non-positive, so put $\Delta = 0$ and taking the positive solution for σ_n^2, giving

$$\sigma_n^2 = \frac{-\log\log 2}{2\mathfrak{c}_2\left(\log\left(n\mathfrak{c}_1\right) - \log\log 2\right)}. \quad \text{(B.102)}$$

Therefore the inequality is satisfied provided $n\mathfrak{c}_1 > 1$, which occurs for sufficiently large n, since $\mathfrak{c}_1 > 0$. Next we show that the stochastic dominance condition is satisfied for $z \in [\mu_n, 0]$, under the proposed choice of μ_n and σ_n above. Over this interval, we can use the same upper bound on $Q(z)^n$ as before, and now we have the lower bound

$$Q\left(\frac{z-\mu_n}{\sigma_n}\right) \geq \exp\left(-\left(\mathfrak{c}_2\left(\frac{z-\mu_n}{\sigma_n}\right)^2 - \log\mathfrak{c}_1\right)\right). \quad \text{(B.103)}$$

Thus we want to show that

$$n\mathfrak{c}_1 \exp\left(-\mathfrak{c}_2 z^2\right) \geq \mathfrak{c}_2 \left(\frac{z-\mu_n}{\sigma_n}\right)^2 - \log \mathfrak{c}_1. \tag{B.104}$$

Fix z, and recognise that $\frac{\mu_n^2}{\sigma_n^2} = O\left((\log n)^2\right)$ in the right-hand side, while the left-hand side is $O(n)$. Therefore

$$O(n) \geq \left((\log n)^2\right) \tag{B.105}$$

since $O(e^n) \geq O\left(n^2\right)$. Lastly for the interval $z \in [0, \infty)$, we use the upper bound in Lemma B.6 to give

$$Q(z)^n \leq \frac{1}{2^n} \exp\left(-\frac{nz^2}{2}\right) \tag{B.106}$$

and we can use the same lower bound as in the preceding interval. In the same vein as above, we want to show

$$\left(\frac{n}{2} - \frac{\mathfrak{c}_2}{\sigma_n^2}\right) z^2 - \frac{2\mathfrak{c}_2\mu_n}{\sigma_n^2} + n\log 2 + \frac{\mu_n^2}{\sigma_n^2} - \log \mathfrak{c}_1 \geq 0. \tag{B.107}$$

The discriminant of the quadratic is non-positive when

$$\left(\frac{n}{2} - \frac{\mathfrak{c}_2}{\sigma_n^2}\right)\left(n\log 2 + \frac{\mu_n^2}{\sigma_n^2} - \log \mathfrak{c}_1\right) \geq \frac{\mathfrak{c}_2^2\mu_n^2}{\sigma_n^4}. \tag{B.108}$$

The left-hand side is $O\left(n^2\right)$ and the right-hand side is $O\left((\log n)^3\right)$, thus this inequality is also satisfied for sufficiently large n.

B.3.10 Proof of Theorem 5.7

The right inequality in (5.44) occurs as a univariate special case of the approximation scheme (5.38), in conjunction with the constructed stochastically dominating approximation in Lemma 5.2 satisfying the conditions of Lemma 5.1. The left inequality in (5.44) is a result of monotonicity in m from Theorem 5.3(b). The inequalities in (5.46) follow naturally from the class of inequalities in (5.44), by directly taking the supremum with respect to ω.

B.4 Implementation of Lower Bounds in Theorem 5.7

In continuation of the discussion from Remark 5.1, the lower bounds in Theorem 5.7 can be implemented numerically. This is done by using sufficient conditions found in the proof of Lemma 5.2 to check whether $n \geq n^*(\omega)$ for a given n and ω. We are required to check whether the inequality (B.91) is satisfied over each of the intervals $(\infty, \mu_n]$, $[\mu_n, 0]$ and $[0, \infty)$. The inequality is satisfied over $(\infty, \mu_n]$ by construction provided $n\mathfrak{c}_1 > 1$, whereas (B.108) contains the sufficient condition for the interval $[0, \infty)$. As for the bounded interval $[\mu_n, 0]$, we can directly evaluate (up to the available numerical precision) whether (B.91) is satisfied. Pseudocode to implement this numerical certificate is provided in Algorithm B.1. Using this certificate, we can implement the optimised lower bound (5.46) in Theorem 5.7. Pseudocode for this is found in Algorithm B.2.

Algorithm B.1 Numerical certification of sufficient conditions for $n \geq n^*(\omega)$ in Theorem 5.7

function NUMERICALCERT(n, ω)
    $\mathfrak{c}_1 \leftarrow \frac{1}{2} - \frac{\omega}{\pi}, \quad \mathfrak{c}_2 \leftarrow \frac{\cot \omega}{\pi - 2\omega}$
    $\mu_n \leftarrow -\sqrt{\frac{\log(n\mathfrak{c}_1)}{\mathfrak{c}_2}}, \quad \sigma_n^2 \leftarrow \frac{-\log\log 2}{2\mathfrak{c}_2(\log(n\mathfrak{c}_1) - \log\log 2)}$
    if $n\mathfrak{c}_1 \leq 1$ **then** ▷ Check sufficient condition for the interval $(\infty, \mu_n]$
        return `False`
    else if (B.91) fails over $[\mu_n, 0]$ **then** ▷ Check sufficient condition for the interval $[\mu_n, 0]$
        return `False`
    else if (B.108) fails **then** ▷ Check sufficient condition for the interval $[0, \infty)$
        return `False`
    else
        return `True`

Algorithm B.2 Implementation of lower bounds in Theorem 5.7

function LOWERBOUND(n, α, ρ, ω) ▷ Lower bound in (5.44)
    if NUMERICALCERT(n, ω) **then**
        return Right-hand side of (5.44)
    else
        return 0
function OPTIMISEDLOWERBOUND(n, α, ρ) ▷ Optimised lower bound in (5.46)
    return $\max_{\omega \in (0,\pi/2)}$ LOWERBOUND(n, α, ρ, ω)

B.5 Experiment Details

B.5.1 Constraints

To formulate the matrices M, f, E, h in the constraints, we start from the actuator lower and upper limits

$$\mathsf{u}_{\mathrm{lb}} = \begin{bmatrix} 0 & 0 & 20 \end{bmatrix}^{\top} \tag{B.109}$$

$$\mathsf{u}_{\mathrm{ub}} = \begin{bmatrix} 95 & 90 & 95 \end{bmatrix}^{\top}. \tag{B.110}$$

Additionally, we impose the saturation constraint in trimmed coordinates

$$\widetilde{\mathsf{u}}_{\mathrm{sat}} = \begin{bmatrix} 15 & 15 & 15 \end{bmatrix}^{\top} \tag{B.111}$$

so that in trimmed coordinates, the upper and lower bounds are

$$\widetilde{\mathsf{u}}_{\max} = \min\left\{\mathsf{u}_{\mathrm{lb}} - \mathsf{u}_{\mathrm{ss}}\left(\mathfrak{p}_k\right), \widetilde{\mathsf{u}}_{\mathrm{sat}}\right\} \tag{B.112}$$

$$\widetilde{\mathsf{u}}_{\min} = \max\left\{\mathsf{u}_{\mathrm{ub}} - \mathsf{u}_{\mathrm{ss}}\left(\mathfrak{p}_k\right), -\widetilde{\mathsf{u}}_{\mathrm{sat}}\right\}, \tag{B.113}$$

where the maximum and minimum are taken component-wise. Then

$$\underbrace{\begin{bmatrix} I \\ -I \end{bmatrix}}_{\mathsf{E}} \widetilde{\mathsf{u}}_{k|i} \leq \underbrace{\begin{bmatrix} \widetilde{\mathsf{u}}_{\max} \\ -\widetilde{\mathsf{u}}_{\min} \end{bmatrix}}_{\mathsf{h}}. \tag{B.114}$$

As for the augmented trimmed state constraints, we enforce the non-negativity of the compressor flow $\mathrm{W}_{\mathrm{comp}} \geq 0$, and the EGR rate constraint of $\mathrm{y}_{\mathrm{EGR}} \in [0, 1]$ in a similar manner:

$$\underbrace{\begin{bmatrix} 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \end{bmatrix}}_{\mathsf{M}} \widetilde{\mathsf{x}}'_{k|i+1} \leq \underbrace{\begin{bmatrix} -\left(0 - \mathrm{W}_{\mathrm{comp,ss}}\left(\mathfrak{p}_k\right)\right) \\ \left(1 - \mathrm{y}_{\mathrm{EGR,ss}}\left(\mathfrak{p}_k\right)\right) \\ -\left(0 - \mathrm{y}_{\mathrm{EGR,ss}}\left(\mathfrak{p}_k\right)\right) \end{bmatrix}}_{\mathsf{f}}. \tag{B.115}$$

B.5.2 Meta-Cost Function

A step response from one grid point to another is evaluated using the preference function on 8 features from (3.31). That is, for grid point j from initial grid point j', the performance is evaluated as

$$\begin{aligned}\overline{J}_{j\leftarrow j'} = 0.03\mathtt{RT}_1 + 0.0238\mathtt{ST}_1 + 0.0113\mathtt{OS}_1 + 0.0295\mathtt{US}_1 \\ + 0.0465\mathtt{RT}_2 + 0.0455\mathtt{ST}_2 + 0.4107\mathtt{OS}_2 + 0.4029\mathtt{US}_2,\end{aligned} \quad \text{(B.116)}$$

where the coefficients have been obtained from preference learning in Section 3.3, and the features $\mathtt{RT}_1$, $\mathtt{ST}_1$, etc. are defined in Table 3.1, computed from the closed-loop response with respect to the candidate controller $[\boldsymbol{\theta}]_j$ for grid point j, with trimmed initial condition

$$\widetilde{\mathsf{x}}_0 = \mathsf{x}_{\mathrm{ss}}\left(\overline{\mathsf{p}}^{\{j'\}}\right) - \mathsf{x}_{\mathrm{ss}}\left(\overline{\mathsf{p}}^{\{j\}}\right) \quad \text{(B.117)}$$

and previously held input

$$\widetilde{\mathsf{u}}_{-1} = \mathsf{u}_{\mathrm{ss}}\left(\overline{\mathsf{p}}^{\{j'\}}\right) - \mathsf{u}_{\mathrm{ss}}\left(\overline{\mathsf{p}}^{\{j\}}\right). \quad \text{(B.118)}$$

The control law in offline simulation is as in (7.87) (i.e. without integrator augmentation), while the $\mathsf{P}^{\{j\}}$ matrix is generated by solving the Riccati equation (2.11). However, as we are tuning for drive-cycles which traverse through the operating space, the initial grid point may not always be the same. Thus for each grid point, a weighted average of performances from adjacent grid points is taken. For example, at grid point $j = 7$ in Figure 6.1, we take

$$\overline{J}_7 = \mathtt{w}_{\mathrm{north}}\overline{J}_{7\leftarrow 6} + \mathtt{w}_{\mathrm{south}}\overline{J}_{7\leftarrow 8} + \mathtt{w}_{\mathrm{east}}\overline{J}_{7\leftarrow 11} + \mathtt{w}_{\mathrm{west}}\overline{J}_{7\leftarrow 2} \quad \text{(B.119)}$$

where $\mathtt{w}_{\mathrm{north}}$, $\mathtt{w}_{\mathrm{south}}$, $\mathtt{w}_{\mathrm{east}}$, $\mathtt{w}_{\mathrm{west}}$ are weighted by the empirical frequencies of typical transitions between grid points over the UDC, EUDC and WLTC.

Appendix C
Sequential Learning Algorithm

C.1 Proof of Lemma 7.2

We prove the upper tail concentration bound; the steps for the lower tail are similar and analogous. From Definition 7.2, the population Kendall correlation κ and the associated Gaussian copula correlation ρ are related by $\rho = \sin(\pi\kappa/2)$. So for $r > 0$ we have

$$\Pr(\widehat{\rho}_n - \rho > r) = \Pr\left(\sin\left(\frac{\pi}{2}\max\{\widehat{\kappa}_n, 0\}\right) - \sin\left(\frac{\pi}{2}\kappa\right) > r\right) \tag{C.1}$$

$$= \Pr\left(\sin\left(\frac{\pi}{2}\widehat{\kappa}_n\right) - \sin\left(\frac{\pi}{2}\kappa\right) > r\right). \tag{C.2}$$

where we able to take $\max\{\widehat{\kappa}_n, 0\} = \widehat{\kappa}_n$ since $\widehat{\kappa}_n \geq 0$ is necessary for $\widehat{\rho}_n - \rho$, as $\rho > 0$ by Assumption 7.1. Note that the event $\frac{\pi}{2}(\widehat{\kappa}_n - \kappa) > r$ together with $r > 0$ implies that $\widehat{\kappa}_n - \kappa > 0$. Since $\sin(\cdot)$ is 1-Lipschitz continuous, then generally

$$\left|\sin\left(\frac{\pi}{2}\widehat{\kappa}_n\right) - \sin\left(\frac{\pi}{2}\kappa\right)\right| \leq \left|\frac{\pi}{2}\widehat{\kappa}_n - \frac{\pi}{2}\kappa\right|. \tag{C.3}$$

However as we have established the sign of $\widehat{\kappa}_n - \kappa$, then the event $\frac{\pi}{2}(\widehat{\kappa}_n - \kappa) > r$ together with $r > 0$ further implies that

$$\sin\left(\frac{\pi}{2}\widehat{\kappa}_n\right) - \sin\left(\frac{\pi}{2}\kappa\right) \leq \frac{\pi}{2}(\widehat{\kappa}_n - \kappa). \tag{C.4}$$

Thus

$$\Pr\left(\widehat{\kappa}_n - \kappa > \frac{2r}{\pi}\right) = \Pr\left(\frac{\pi}{2}(\widehat{\kappa}_n - \kappa) > r\right) \tag{C.5}$$

$$\geq \Pr\left(\sin\left(\frac{\pi}{2}\widehat{\kappa}_n\right) - \sin\left(\frac{\pi}{2}\kappa\right) > r\right) \tag{C.6}$$

$$= \Pr\left(\widehat{\rho}_n - \rho > r\right). \tag{C.7}$$

Using the fact that $\widehat{\kappa}_n$ is an unbiased estimator for κ, and moreover a U-statistic with a second-order kernel bounded between -1 and 1, we use [95, Equation (5.7)] to obtain

$$\Pr\left(\widehat{\kappa}_n - \kappa > \frac{2r}{\pi}\right) \leq \exp\left(-\left\lfloor\frac{n}{2}\right\rfloor \frac{2r^2}{\pi^2}\right). \tag{C.8}$$

Combining with (C.7) completes our proof.

C.2 Optimised Bound for Theorem 7.2

The lower bound (7.51) for the distribution of the stopping time can be optimised by

$$\Pr(\tau \leq n) \geq 1 - \min_{(\alpha^*,\rho^*)\in\mathbb{A}} \left\{\exp\left(-2n\left(\alpha_0 - \alpha^* - b_1\right)^2\right) + \exp\left(-\left\lfloor\frac{n}{2}\right\rfloor \frac{2\left(\rho_0\rho^* - b_2\right)^2}{\pi^2}\right)\right\}, \tag{C.9}$$

where

$$\mathbb{A} := \left\{(\alpha^*, \rho^*) \in (0,1]^2 : \alpha_0 - b_1 > \alpha^*, \rho_0 - b_2 > \rho^*, \underline{p}^{\mathcal{N}}_{\text{success}}(n, 1, \alpha^*, \rho^*) \geq 1 - \delta\right\} \tag{C.10}$$

is the region of $(0,1]^2$ where the gaps $(\alpha_0 - b_1) - \alpha^*$ and $(\rho_0 - b_2) - \rho^*$ are positive. Moreover, because the bound is improving with the gaps $(\alpha_0 - b_1) - \alpha^*$ and $(\rho_0 - b_2) - \rho^*$, and also because of the monotonicity properties in Theorems 5.3(c) and 5.6, the optimum will lie on the Pareto front $\underline{p}^{\mathcal{N}}_{\text{success}}(n, 1, \alpha^*, \rho^*) = 1 - \delta$ for $(\alpha^*, \rho^*) \in (0,1]^2$. For a given ω, we can instead analytically determine the Pareto front along $\underline{p}^{\mathcal{N}}_{\text{success},\omega}(n, 1, \alpha^*, \rho^*) = 1 - \delta$ using (5.44). Letting

$$\Phi\left(\frac{\Phi^{-1}(\alpha) - \rho\mu_n}{\sqrt{1 - \rho^2 + \rho^2\sigma_n^2}}\right) = 1 - \delta, \tag{C.11}$$

and putting the definitions of μ_n, σ_n^2 and rearranging, we obtain a quadratic form in α, ρ given by

$$\mathfrak{d}_1\rho^2 + \mathfrak{d}_2\Phi^{-1}(\alpha)\rho + \mathfrak{d}_3\Phi^{-1}(\alpha)^2 + \mathfrak{d}_4 = 0, \tag{C.12}$$

where

$$\mathfrak{d}_1 = -\frac{2\log(n\mathfrak{c}_1)^2}{\log\log 2} + 2\log(n\mathfrak{c}_1) - \frac{2\mathfrak{c}_2\Phi^{-1}(1-\delta)^2\log(n\mathfrak{c}_1)}{\log\log 2} + 2\mathfrak{c}_2\Phi^{-1}(1-\delta)^2 - \Phi^{-1}(1-\delta)^2 \tag{C.13}$$

$$\mathfrak{d}_2 = -\frac{4\sqrt{\mathfrak{c}_2}\log(n\mathfrak{c}_1)^{3/2}}{\log\log 2} + 4\sqrt{\mathfrak{c}_2}\log(n\mathfrak{c}_1)^{1/2} \tag{C.14}$$

$$\mathfrak{d}_3 = -\frac{2\mathfrak{c}_2\log(n\mathfrak{c}_1)}{\log\log 2} + 2\mathfrak{c}_2 \tag{C.15}$$

$$\mathfrak{d}_4 = \frac{2\mathfrak{c}_2\Phi^{-1}(1-\delta)^2\log(n\mathfrak{c}_1)}{\log\log 2} - 2\mathfrak{c}_2\Phi^{-1}(1-\delta)^2. \tag{C.16}$$

Thus given ω, n, δ, we can solve for ρ in terms of α with

$$\rho_\omega(\alpha) = \frac{1}{2\mathfrak{d}_1}\left[-\left(\mathfrak{d}_2\Phi^{-1}(\alpha)\right) + \sqrt{\left(\mathfrak{d}_2\Phi^{-1}(\alpha)\right)^2 - 4\mathfrak{d}_1\left(\mathfrak{d}_3\Phi^{-1}(\alpha)^2 + \mathfrak{d}_4\right)}\right]. \tag{C.17}$$

Alternatively given ω, n, δ, we can solve for α in terms of ρ with

$$\alpha_\omega(\rho) = \Phi\left(\frac{-(\mathfrak{d}_2\rho) + \sqrt{(\mathfrak{d}_2\rho)^2 - 4\mathfrak{d}_3(\mathfrak{d}_1\rho^2 + \mathfrak{d}_4)}}{2\mathfrak{d}_3}\right), \tag{C.18}$$

where we have taken the positive solutions of the quadratics since $\alpha > 0$, $\rho > 0$. We may then proceed to optimise with respect to α^* (and ρ^* implicitly in terms of α^*) with an inner minimisation for a given ω, and then optimise with respect to ω in an outer minimisation. Explicitly, (C.9) becomes

$$\Pr(\tau \le n) \ge 1 - \inf_{\omega\in\Omega_n}\left\{\min_{\alpha^*\in\mathbb{A}'_\omega}\left\{\exp\left(-2n(\alpha_0 - \alpha^* - b_1)^2\right) + \exp\left(-\left\lfloor\frac{n}{2}\right\rfloor\frac{2(\rho_0 - \rho_\omega(\alpha^*) - b_2)^2}{\pi^2}\right)\right\}\right\}, \tag{C.19}$$

where

$$\mathbb{A}'_\omega := \{\alpha \in (0,1] : \alpha_\omega(\rho_0 - b_2) \le \alpha \le \alpha_0 - b_1\}, \tag{C.20}$$

and $\omega \in \Omega_n \subset (0, \pi/2)$ is defined the same as in (5.46). The inner minimisation in (C.19) is quasiconvex, thus the optimised bound is not too difficult to numerically implement. This is further illustrated in Figure C.1.

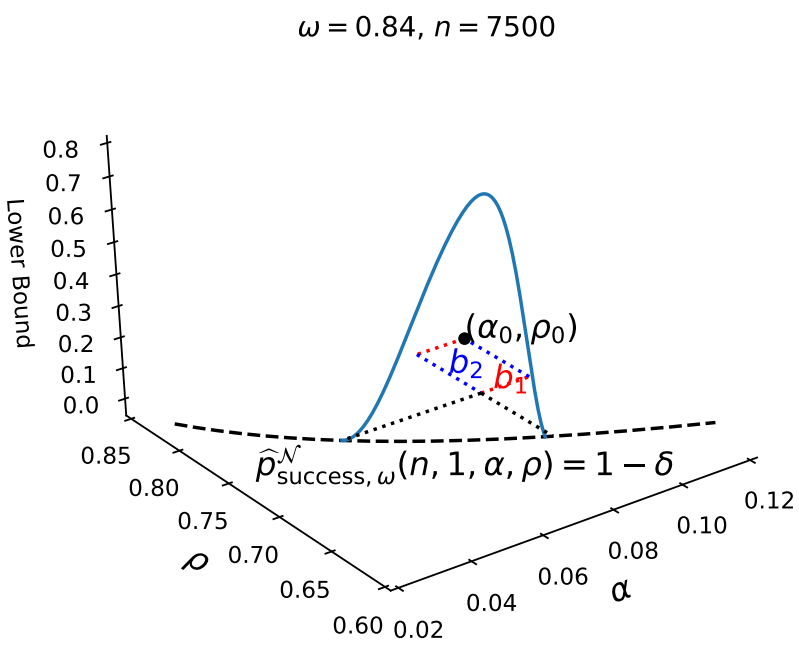


Figure C.1: A visual depiction of how the bound (C.19) is optimised. The solid curve is a plot of the original bound (7.51) along the Pareto front (dashed curve), for given $\omega = 0.84$, and with $n = 7500$, $\alpha_0 = 0.1$, $\rho_0 = 0.8$, $\delta = 0.1$, $\beta_1 = 0.05$, $\beta_2 = 0.05$. The dotted lines indicate how the gaps $(\alpha_0 - b_1) - \alpha^*$ and $(\rho_0 - b_2) - \rho^*$ must be positive in order for the bound to be informative.

Bibliography

[1] "Birmingham environment for academic research (BEAR)," http://www.birmingham.ac.uk/bear.

[2] *Toyota Fortuner Technical Specifications*. [Online]. Available: https://web.archive.org/web/20210118034301/https://www.toyota.com.au/-/media/toyota/main-site/vehicle-hubs/fortuner/files/fortuner_spec_table_dec2020.pdf

[3] K. J. Åström and T. Hägglund, *PID Controllers: Theory, Design and Tuning*. International Society for Measurement and Control, 1995.

[4] F. B. Aicha, F. Bouani, and M. Ksouri, "Automatic tuning of GPC synthesis parameters based on multi-objective optimization," in *XIth International Workshop on Symbolic and Numerical Methods, Modeling and Applications to Circuit Design (SM2ACD)*. IEEE, 2010.

[5] Y. Akimoto, Y. Nagata, I. Ono, and S. Kobayashi, "Theoretical foundation for CMA-ES from information geometry perspective," *Algorithmica*, vol. 64, no. 4, pp. 698–716, 2011.

[6] A. Al-Ghazzawi, E. Ali, A. Nouh, and E. Zafiriou, "On-line tuning strategy for model predictive controllers," *Journal of Process Control*, vol. 11, no. 3, pp. 265–284, 2001.

[7] T. Alamo, R. Tempo, A. Luque, and D. R. Ramirez, "Randomized methods for design of uncertain systems: Sample complexity and sequential algorithms," *Automatica*, vol. 52, pp. 160–172, 2015.

[8] M. Alhajeri and M. Soroush, "Tuning guidelines for model-predictive control," *Industrial & Engineering Chemistry Research*, vol. 59, no. 10, pp. 4177–4191, 2020.

[9] E. Ali and E. Zafiriou, "Optimization-based tuning of nonlinear model predictive control with state estimation," *Journal of Process Control*, vol. 3, no. 2, pp. 97–107, 1993.

[10] T. Alpcan, T. Basar, and R. Tempo, "Randomized algorithms for stability and robustness analysis of high speed communication networks," in *IEEE Conference on Control Applications*. IEEE, 2003.

[11] J. E. Angus, "The probability integral transform and related results," *SIAM Review*, vol. 36, no. 4, pp. 652–654, 1994.

[12] B. C. Arnold, N. Balakrishnan, and H. N. Nagaraja, *A First Course in Order Statistics*. SIAM, 2008.

[13] K. J. Aström and P. Eykhoff, "System identification - a survey," *Automatica*, vol. 7, no. 2, pp. 123–162, 1971.

[14] A. Auger, N. Hansen, J. P. Zerpa, R. Ros, and M. Schoenauer, "Experimental comparisons of derivative free optimization algorithms," in *8th International Symposium on Experimental Algorithms*, 2009.

[15] R. Ayyagari, "A fresh approach to teaching state-space methods in an undergraduate course," in *12th IFAC Symposium on Advances in Control Education*, vol. 52, no. 9. Elsevier BV, 2019, pp. 97–102.

[16] V. Bachtiar, C. Manzie, W. H. Moase, and E. C. Kerrigan, "Analytical results for the multi-objective design of model-predictive control," *Control Engineering Practice*, vol. 56, pp. 1–12, 2016.

[17] P. Bagheri and A. Khaki-Sedigh, "Tuning of dynamic matrix controller for FOPDT models using analysis of variance," in *18th IFAC World Congress*, 2011.

[18] P. Bagheri and A. K. Sedigh, "Analytical approach to tuning of model predictive control for first-order plus dead time models," *IET Control Theory & Applications*, vol. 7, no. 14, pp. 1806–1817, 2013.

[19] M. Baric, M. Baotic, and M. Morari, "On-line tuning of controllers for systems with constraints," in *44th IEEE Conference on Decision and Control*. IEEE, 2005.

[20] N. Barile and A. Feelders, "Active learning with monotonicity constraints," in *SIAM International Conference on Data Mining*. Society for Industrial and Applied Mathematics, 2012, pp. 756–767.

[21] G. Bayraksan and P. Pierre-Louis, "Fixed-width sequential stopping rules for a class of stochastic programs," *SIAM Journal on Optimization*, vol. 22, no. 4, pp. 1518–1548, 2012.

[22] M. Ben-Akiva, D. McFadden, and K. Train, "Foundations of stated preference elicitation: Consumer behavior and choice-based conjoint analysis," *Foundations and Trends® in Econometrics*, vol. 10, no. 1-2, pp. 1–144, 2019.

[23] D. Bertsekas, *Dynamic Programming and Optimal Control: Volume I*, 3rd ed. Athena Scientific, 2005.

[24] I. Bilionis and N. Zabaras, "Multi-output local gaussian process regression: Applications to uncertainty quantification," *Journal of Computational Physics*, vol. 231, no. 17, pp. 5718–5746, 2012.

[25] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer-Verlag New York Inc., 2006.

[26] F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, 2017.

[27] M. Boutahar and C. Deniau, "A proof of asymptotic normality for some VARX models," *Metrika*, vol. 42, no. 1, pp. 331–339, 1995.

[28] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[29] R. A. Bradley and M. E. Terry, "Rank analysis of incomplete block designs: I. the method of paired comparisons," *Biometrika*, vol. 39, no. 3/4, p. 324, 1952.

[30] C. Briat, *Linear Parameter-Varying and Time-Delay Systems: Analysis, Observation, Filtering & Control*. Springer, 2015.

[31] E. Brochu, N. de Freitas, and A. Ghosh, "Active preference learning with discrete choice data," in *Advances in Neural Information Processing Systems*, 2007.

[32] G. A. Bunin, F. F. Tirado, G. François, and D. Bonvin, "Run-to-run MPC tuning via gradient descent," *Computer Aided Chemical Engineering*, pp. 927–931, 2012.

[33] S. D. Cairano and A. Bemporad, "Model predictive controller matching: Can mpc enjoy small signal properties of my favorite linear controller?" in *European Control Conference*, 2009.

[34] S. D. Cairano and I. V. Kolmanovsky, "Automotive applications of model predictive control," in *Handbook of Model Predictive Control*. Springer International Publishing, 2019, pp. 493–527.

[35] G. Calafiore, F. Dabbene, and R. Tempo, "Randomized algorithms in robust control," in *42nd IEEE International Conference on Decision and Control*. IEEE, 2003.

[36] G. Calafiore and M. Campi, "The scenario approach to robust control design," *IEEE Transactions on Automatic Control*, vol. 51, no. 5, pp. 742–753, 2006.

[37] E. F. Camacho and C. Bordons, *Model Predictive Control*. Springer-Verlag GmbH, 2004.

[38] M. Capinski and P. E. Kopp, *Measure, Integral and Probability*. Springer, 2004.

[39] H.-C. Chen, C.-H. Chen, and E. Yucesan, "Computing efforts allocation for ordinal optimization and discrete event simulation," *IEEE Transactions on Automatic Control*, vol. 45, no. 5, pp. 960–964, 2000.

[40] G. Cheng, P. Li, and P. Shi, "A new algorithm based on copulas for VaR valuation with empirical calculations," *Theoretical Computer Science*, vol. 378, no. 2, pp. 190–197, 2007.

[41] M. Chiani, D. Dardari, and M. Simon, "New exponential bounds and approximations for the computation of error probability in fading channels," *IEEE Transactions on Wireless Communications*, vol. 24, no. 5, pp. 840–845, 2003.

[42] R. Chin, A. I. Maass, N. Ulapane, C. Manzie, I. Shames, D. Nešić, J. E. Rowe, and H. Nakada, "Active learning for linear parameter-varying system identification," in *IFAC World Congress*, 2020.

[43] R. Chin, C. Manzie, A. Ira, D. Nešić, and I. Shames, "Gaussian processes with monotonicity constraints for preference learning from pairwise comparisons," in *IEEE Conference on Decision and Control*. IEEE, 2018.

[44] R. Chin, C. Manzie, I. Shames, D. Nešić, and J. E. Rowe, "A sequential learning algorithm for probabilistically robust controller tuning," 2021, arXiv preprint arXiv:2102.09738.

[45] R. Chin and J. E. Rowe, "Re-parametrising cost matrices for tuning model predictive controllers," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2019.

[46] R. Chin, J. E. Rowe, I. Shames, C. Manzie, and D. Nešić, "Ordinal optimisation and the offline multiple noisy secretary problem," 2021, arXiv preprint arXiv:2106.01185.

[47] D. Chindamo and M. Gadola, "What is the most representative standard driving cycle to estimate diesel emissions of a light commercial vehicle?" in *1st IFAC Workshop on Integrated Assessment Modelling for Environmental Systems*. Elsevier, 2018.

[48] P. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," in *Advances in Neural Information Processing Systems*, 2017, arXiv.

[49] W. Chu and Z. Ghahramani, "Preference learning with gaussian processes," in *22nd International Conference on Machine Learning*, 2005.

[50] C. Criens, "Air-path control of clean diesel engines," Ph.D. dissertation, Eindhoven University of Technology, 2014.

[51] H. A. David and H. N. Nagaraja, *Order Statistics*. John Wiley, 2005.

[52] A. Davtyan, S. Hoffmann, and R. Scheuring, "Optimization of model predictive control by means of sequential parameter optimization," in *Computational Intelligence in Control and Automation (CICA)*. IEEE, 2011.

[53] G. M. de Almeida, M. A. D. S. Cuadro, R. P. P. Amaral, and J. L. F. Salles, "Optimal tuning parameters of the dynamic matrix predictive controller with ant colony optimization," in *11th IEEE/IAS International Conference on Industry Applications*. IEEE, 2014.

[54] G. Debreu, "Representation of a preference ordering by a numerical function," in *Decision Process*. John Wiley, 1954.

[55] M. Deng and Y.-C. Ho, "An ordinal optimization approach to optimal control problems," *Automatica*, vol. 35, no. 2, pp. 331–338, 1999.

[56] M. L. Derouiche, S. Bouallègue, J. Haggège, and G. Sandou, "LabVIEW perturbed particle swarm optimization based approach for model predictive control tuning," in *4th IFAC Conference on Intelligent Control and Automation Sciences*, 2016.

[57] L. Desborough and R. Miller, "Increasing customer value of industrial control performance monitoring—honeywell's experience," in *6th AIChE International Conference on Chemical Process Control*, 2002.

[58] I. Dewancker, M. McCourt, and S. Ainsworth, "Interactive preference learning of utility functions for multi-objective optimization," in *Future of Interactive Learning Machines Workshop at NIPS*, 2016.

[59] S. X. Ding, L. Li, and M. Krüger, "Application of randomized algorithms to assessment and design of observer-based fault detection systems," *Automatica*, vol. 107, pp. 175–182, 2019.

[60] P. L. dos Santos, T. P. A. Perdicoúlis, C. Novara, J. A. Ramos, and D. E. Rivera, Eds., *Linear Parameter-varying System Identification: New Developments and Trends*. World Scientific Pub Co Inc, 2012.

[61] D. Dougherty and D. J. Cooper, "Tuning guidelines of a dynamic matrix controller for integrating (non-self-regulating) processes," *Industrial & Engineering Chemistry Research*, vol. 42, no. 8, pp. 1739–1752, 2003.

[62] S. Drogies and D. D. Geest, "Predictive control: propositions for the design methodology," in *American Control Conference*. IEEE, 1999.

[63] D. P. Dubhashi and A. Panconesi, *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.

[64] A. E. Elo, "Rating system theory," in *The Rating of Chess Players, Past and Present*, 2nd ed. Ishi Press International, 1986.

[65] L. Eriksson and L. Nielsen, *Modeling and Control of Engines and Drivelines*. Wiley, 2014.

[66] V. Exadaktylos and C. J. Taylor, "Multi-objective performance optimisation for model predictive control by goal attainment," *International Journal of Control*, vol. 83, no. 7, pp. 1374–1386, 2010.

[67] J. Fan, G. Stewart, and G. Dumont, "Two-dimensional frequency response analysis and insights for weight selection in cross-directional model predictive control," in *42nd IEEE International Conference on Decision and Control*. IEEE, 2003.

[68] R. M. Fontes, M. A. F. Martins, and D. Odloak, "An automatic tuning method for model predictive control strategies," *Industrial & Engineering Chemistry Research*, vol. 58, no. 47, pp. 21 602–21 613, 2019.

[69] M. Foo and E. Weyer, "On reproducing existing controllers as model predictive controllers," in *Australian Control Conference*, 2011.

[70] Y. Fujisaki and Y. Oishi, "Guaranteed cost regulator design: A probabilistic solution and a randomized algorithm," *Automatica*, vol. 43, no. 2, pp. 317–324, 2007.

[71] J. Fürnkranz and E. Hüllermeier, "Preference learning: An introduction," in *Preference Learning*. Springer Berlin Heidelberg, 2010, pp. 1–17.

[72] M. Gallieri, *Lasso-MPC – Predictive Control with ℓ1-Regularised Least Squares*. Springer, 2016.

[73] C. E. García, D. M. Prett, and M. Morari, "Model predictive control: Theory and practice—a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[74] J. L. Garriga and M. Soroush, "Model predictive controller tuning via eigenvalue placement," in *American Control Conference*. IEEE, 2008.

[75] ——, "Model predictive control tuning methods: A review," *Industrial & Engineering Chemistry Research*, vol. 49, no. 8, pp. 3505–3515, 2010.

[76] G. Geenens, A. Charpentier, and D. Paindaveine, "Probit transformation for nonparametric kernel estimation of the copula density," *Bernoulli*, vol. 23, no. 3, pp. 1848–1873, 2017.

[77] A. Genz and F. Bretz, "Comparison of methods for the computation of multivariate t probabilities," *Journal of Computational and Graphical Statistics*, vol. 11, no. 4, pp. 950–971, 2002.

[78] P. Glynn and S. Juneja, "A large deviations perspective on ordinal optimization," in *Winter Simulation Conference*. IEEE, 2004.

[79] P. Golchoubian and N. L. Azad, "Real-time nonlinear model predictive control of a battery–supercapacitor hybrid energy storage system in electric vehicles," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 11, pp. 9678–9688, 2017.

[80] G. H. Golub and C. F. V. Loan, *Matrix Computations*, 4th ed. The Johns Hopkins University Press, 2013.

[81] C. A. Gomez-Uribe and N. Hunt, "The netflix recommender system," *ACM Transactions on Management Information Systems*, vol. 6, no. 4, pp. 1–19, 2015.

[82] G. C. Goodwin, "Optimal input signals for nonlinear-system identification," *Proceedings of the Institution of Electrical Engineers*, vol. 118, no. 7, p. 922, 1971.

[83] G. C. Goodwin, S. F. Graebe, and M. E. Salgado, *Control System Design*. Addison Wesley, 2000.

[84] G. Gous and P. de Vaal, "Using MV overshoot as a tuning metric in choosing DMC move suppression values," *ISA Transactions*, vol. 51, no. 5, pp. 657–664, 2012.

[85] R. M. Gray, *Entropy and Information Theory*. Springer, 2011.

[86] N. Hansen, "Benchmarking a bi-population CMA-ES on the BBOB-2009 function testbed," in *Genetic and Evolutionary Computation Conference*, 2009.

[87] ——, "The CMA evolution strategy: A tutorial," University of Paris-Saclay, Tech. Rep., 2016.

[88] N. Hansen, A. Auger, R. Ros, S. Finck, and P. Pošík, "Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009," in *Genetic and Evolutionary Computation Conference*, 2010.

[89] M. Harder, "Exchangeability of copulas," Ph.D. dissertation, Ulm University, 2016.

[90] H. Hayakawa, "Lexicographic preferences and consumer theory," *Journal of Behavioral Economics*, vol. 7, no. 1, pp. 17–51, 1978.

[91] R. Heckel, M. Simchowitz, K. Ramchandran, and M. Wainwright, "Approximate ranking from pairwise comparisons," in *International Conference on Artificial Intelligence and Statistics*, 2018, pp. 1057–1066.

[92] Y. C. Ho, R. S. Sreenivas, and P. Vakili, "Ordinal optimization of DEDS," *Discrete Event Dynamic Systems*, vol. 2, no. 1, pp. 61–88, 1992.

[93] Y.-C. Ho and M. E. Larson, "Ordinal optimization approach to rare event probability problems," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 5, no. 2-3, pp. 281–301, 1995.

[94] Y.-C. Ho, Q.-C. Zhao, and Q.-S. Jia, *Ordinal Optimization: Soft Optimization for Hard Problems*. Springer, 2007.

[95] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 13–30, 1963.

[96] D. Hrovat, S. D. Cairano, H. Tseng, and I. Kolmanovsky, "The development of model predictive control in automotive industry: A survey," in *IEEE International Conference on Control Applications*. IEEE, 2012.

[97] M. Huang, "Low complexity model predictive control of a diesel engine airpath," phdthesis, University of Michigan, 2016.

[98] M. Huang, H. Nakada, K. Butts, and I. Kolmanovsky, "Nonlinear model predictive control of a diesel engine air path: A comparison of constraint handling and computational strategies," in *5th IFAC Conference on Nonlinear Model Predictive Control*, vol. 48, no. 23. Elsevier BV, 2015, pp. 372–379.

[99] M. Huang, H. Nakada, S. Polavarapu, K. Butts, and I. Kolmanovsky, "Rate-based model predictive control of diesel engines," in *IFAC Symposium on Advances in Automotive Control*, 2013.

[100] M. Huang, K. Zaseck, K. Butts, and I. Kolmanovsky, "Rate-based model predictive controller for diesel engine air path: Design and experimental evaluation," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 6, pp. 1922–1935, 2016.

[101] B. Ibarz, J. Leike, T. Pohlen, G. Irving, S. Legg, and D. Amodei, "Reward learning from human preferences and demonstrations in atari," in *Advances in Neural Information Processing Systems 31*, 2018.

[102] S. ichi Amari, *Information Geometry and Its Applications*. Springer, 2016.

[103] A. S. Ira, C. Manzie, I. Shames, R. Chin, D. Nešić, H. Nakada, and T. Sano, "Tuning of multivariable model predictive controllers through expert bandit feedback," *International Journal of Control*, 2020.

[104] A. S. Ira, I. Shames, C. Manzie, R. Chin, D. Nesic, H. Nakada, and T. Sano, "A machine learning approach for tuning model predictive controllers," in *International Conference on Control, Automation, Robotics and Vision*, 2018.

[105] H. Ishii and R. Tempo, "Las vegas randomized algorithms in distributed consensus problems," in *American Control Conference*. IEEE, 2008.

[106] J. L. Jerez, E. C. Kerrigan, and G. A. Constantinides, "A condensed and sparse QP formulation for predictive control," in *IEEE Conference on Decision and Control and European Control Conference*. IEEE, 2011.

[107] A. Jern, C. G. Lucas, and C. Kemp, "Evaluating the inverse decision-making approach to preference learning," in *Advances in Neural Information Processing Systems*, 2011.

[108] A. Jiang and A. Jutan, "Response surface tuning methods in dynamic matrix control of a pressure tank system," *Industrial & Engineering Chemistry Research*, vol. 39, no. 10, pp. 3835–3843, 2000.

[109] H. Joe, *Dependence Modeling with Copulas*. CRC Press, 2014.

[110] G. A. N. Júnior, M. A. Martins, and R. Kalid, "A PSO-based optimal tuning strategy for constrained multivariable predictive controllers with model uncertainty," *ISA Transactions*, vol. 53, no. 2, pp. 560–567, 2014.

[111] M. Kendall and J. D. Gibbons, *Rank Correlation Methods*, 5th ed. Oxford University Press, 1990.

[112] A. A. Khalate, X. Bombois, R. Tóth, and R. Babuŝka, "Optimal experimental design for LPV identification using a local approach," in *15th IFAC Symposium on System Identification*. Elesvier, 2009.

[113] P. Khargonekar and A. Tikku, "Randomized algorithms for robust control analysis and synthesis have polynomial complexity," in *35th IEEE Conference on Decision and Control*. IEEE, 1996.

[114] K.-K. K. Kim, D. E. Shen, Z. K. Nagy, and R. D. Braatz, "Wiener's polynomial chaos for the analysis and control of nonlinear dynamical systems with probabilistic uncertainties," *IEEE Control Systems Magazine*, vol. 33, no. 5, pp. 58–67, 2013.

[115] V. Koltchinskii, C. T. Abdallah, M. Ariola, P. Dorato, and D. Panchenko, "Improved sample complexity estimates for statistical learning control of uncertain systems," *IEEE Transactions on Automatic Control*, vol. 45, no. 12, pp. 2383–2388, 2000.

[116] H. Kong, G. Goodwin, and M. M. Seron, "Predictive metamorphic control," *Automatica*, vol. 49, no. 12, pp. 3670–3676, 2013.

[117] R. Kyng, A. Rao, and S. Sachdeva, "Fast, provable algorithms for isotonic regression in all lp-norms," in *Advances in Neural Information Processing Systems 28*, 2015.

[118] L. Lafayette, G. Sauter, L. Vu, and B. Meade, "Spartan performance and flexibility: An HPC-cloud chimera," in *OpenStack Summit*, 2016.

[119] S. Lahiri, *Multivariable Predictive Control: Applications in Industry*. John Wiley & Sons Ltd, 2017.

[120] T. Lattimore and C. Szepesvári, *Bandit Algorithms*. Cambridge University Press, 2020.

[121] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*. SIAM, 1995.

[122] M. Levin, "Optimum estimation of impulse response in the presence of noise," *IRE Transactions on Circuit Theory*, vol. 7, no. 1, pp. 50–56, 1960.

[123] S. Lin and Y. Ho, "Universal alignment probability revisited," *Journal of Optimization Theory and Applications*, vol. 113, no. 2, pp. 399–407, 2002.

[124] H. Liu, F. Han, M. Yuan, J. Lafferty, and L. Wasserman, "High-dimensional semiparametric gaussian copula graphical models," *The Annals of Statistics*, vol. 40, no. 4, pp. 2293–2326, 2012.

[125] T.-Y. Liu, *Learning to Rank for Information Retrieval*. Springer, 2011.

[126] L. Ljung, *System Identification*. Prentice Hall, 1999.

[127] H. Lütkepohl, *New Introduction to Multiple Time Series Analysis*. Springer, 2005.

[128] A. I. Maass, C. Manzie, I. Shames, R. Chin, D. Nešić, N. Ulapane, and H. Nakada, "Tuning of model predictive engine controllers over transient drive cycles," in *21st IFAC World Congress*, 2020.

[129] A. I. Maass, C. Manzie, I. Shames, and H. Nakada, "Controller tuning via numerical optimisation with zeroth-order random matrix oracles," 2020, submitted to IEEE Transactions on Control Systems Technology.

[130] ——, "Zeroth-order optimisation on subsets of symmetric matrices with application to mpc tuning," 2021, arXiv preprint arXiv:2106.14359.

[131] J. Maciejowski, *Predictive Control with Constraints*. Prentice Hall, 2002.

[132] D. J. C. MacKay, "Information-based objective functions for active data selection," *Neural Computation*, vol. 4, no. 4, pp. 590–604, 1992.

[133] J. S. Maritz, *Empirical Bayes Methods with Applications*. Chapman and Hall/CRC, 1989.

[134] A. Maxim, D. Copot, C. Copot, and C. M. Ionescu, "The 5w's for control as part of industry 4.0: Why, what, where, who, and when—a PID and MPC control perspective," *Inventions*, vol. 4, no. 1, p. 10, 2019.

[135] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[136] D. McFadden, "Conditional logit analysis of qualitative choice behavior," in *Frontiers in Econometrics*. Academic Press, 1974.

[137] H. Mei, L. Y. Wang, and G. Yin, "Almost sure convergence rates for system identification using binary, quantized, and regular sensors," *Automatica*, vol. 50, no. 8, pp. 2120–2127, 2014.

[138] A. K. Menon and R. C. Williamson, "Bipartite ranking: a risk-theoretic perspective," *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 6766–6867, 2016.

[139] Y. Mizushima, I. Okawa, and K. Nonaka, "Model predictive control for autonomous vehicles with speed profile shaping," *10th IFAC Symposium on Intelligent Autonomous Vehicles*, vol. 52, no. 8, pp. 31–36, 2019.

[140] A. Mohammadi, H. Asadi, S. Mohamed, K. Nelson, and S. Nahavandi, "Optimizing model predictive control horizons using genetic algorithm for motion cueing algorithm," *Expert Systems with Applications*, vol. 92, pp. 73–81, 2018.

[141] S. L. Morgan and C. Winship, *Counterfactuals and Causal Inference: Methods and Principles for Social Research*, 2nd ed. Cambridge University Press, 2015.

[142] F. Mosteller, "Remarks on the method of paired comparisons: I. the least squares solution assuming equal standard deviations and equal correlations," *Psychometrika*, vol. 16, no. 1, pp. 3–9, 1951.

[143] K. Motchon, L. Rajaoarisoa, L. Etienne, and S. Lecoeuche, "On experiment design for local approach identification of LPV systems," in *18th IFAC Symposium on System Identification*. Elesvier, 2018.

[144] T. Nagler and K. Wen, *kdecopula: Kernel Smoothing for Bivariate Copula Densities*, 2018, r package version 0.9.2.

[145] R. B. Nelsen, *An Introduction to Copulas*, 1st ed. Springer, 1999.

[146] Y. Nesterov and V. Spokoiny, "Random gradient-free minimization of convex functions," *Foundations of Computational Mathematics*, vol. 17, no. 2, pp. 527–566, 2017.

[147] A. Y. Ng and S. J. Russell, "Algorithms for inverse reinforcement learning," in *17th International Conference on Machine Learning*, 2000, pp. 663–670.

[148] D. H. Olesen, J. K. Huusom, and J. B. Jorgensen, "A tuning procedure for ARX-based MPC of multivariate processes," in *American Control Conference*. IEEE, 2013.

[149] Y. Ollivier, L. Arnold, A. Auger, and N. Hansen, "Information-geometric optimization algorithms: A unifying picture via invariance principles," *Journal of Machine Learning Research*, vol. 18, no. 18, pp. 1–65, 2017.

[150] M. Ono and B. C. Williams, "Iterative risk allocation: A new approach to robust model predictive control with a joint chance constraint," in *47th IEEE Conference on Decision and Control*. IEEE, 2008.

[151] G. Pannocchia and J. B. Rawlings, "Disturbance models for offset-free model-predictive control," *AIChE Journal*, vol. 49, no. 2, pp. 426–437, 2003.

[152] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018.

[153] Y. Peng, E. K. P. Chong, C.-H. Chen, and M. C. Fu, "Ranking and selection as stochastic control," *IEEE Transactions on Automatic Control*, vol. 63, no. 8, pp. 2359–2373, 2018.

[154] K. Rani and H. Unbehauen, "Study of predictive controller tuning methods," *Automatica*, vol. 33, no. 12, pp. 2243–2248, 1997.

[155] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[156] C. Reeves and J. E. Rowe, *Genetic Algorithms: Principles and Perspectives: A Guide to GA Theory*. Springer Verlag, 2002.

[157] R. D. Reiss, *Approximate Distributions of Order Statistics: With Applications to Non-parametric Statistics*. Springer, 1989.

[158] J. Riihimäki and A. Vehtari, "Gaussian processes with monotonicity information," in *International Conference on Artificial Intelligence and Statistics*, 2010.

[159] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Artech House Radar Library, 2004.

[160] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*. Springer, 2004.

[161] J. Rodrigues, E. Toledo, and R. M. Filho, "A tuned approach of the predictive-adaptive GPC controller applied to a fed-batch bioreactor using complete factorial design," *Computers & Chemical Engineering*, vol. 26, no. 10, pp. 1493–1500, 2002.

[162] G. S. Sankar, W. H. Moase, R. C. Shekhar, T. J. Broomhead, and C. Manzie, "Towards systematic design of MPC to achieve time domain specifications," in *Australian Control Conference*, 2015.

[163] G. S. Sankar, R. C. Shekhar, C. Manzie, and H. Nakada, "Efficient calibration of real-time model-based controllers for diesel engines — part II: Incorporating practical robustness guarantees," in *IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE, 2017.

[164] G. S. Sankar, R. C. Shekhar, C. Manzie, T. Sano, and H. Nakada, "Model predictive controller with average emissions constraints for diesel airpath," *Control Engineering Practice*, vol. 90, pp. 182–189, 2019.

[165] ——, "Fast calibration of a robust model predictive controller for diesel engine airpath," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 4, pp. 1505–1519, 2020.

[166] G. S. Sankar, "Robust predictive control of diesel airpath," Ph.D. dissertation, The University of Melbourne, 2019.

[167] C. Schmid and L. Biegler, "Quadratic programming methods for reduced hessian SQP," *Computers & Chemical Engineering*, vol. 18, no. 9, pp. 817–832, 1994.

[168] B. Settles, *Active Learning*. Morgan & Claypool, 2012.

[169] M. Shaked and J. G. Shanthikumar, *Stochastic Orders*. Springer, 2007.

[170] S. Shariati and D. Abel, "Model predictive control in two days: Educating a new way of thinking," in *11th IFAC Symposium on Advances in Control Education*, vol. 49, no. 6. Elsevier BV, 2016, pp. 40–45.

[171] R. C. Shekhar and C. Manzie, "Optimal move blocking strategies for model predictive control," *Automatica*, vol. 61, pp. 27–34, 2015.

[172] R. C. Shekhar, G. S. Sankar, C. Manzie, and H. Nakada, "Efficient calibration of real-time model-based controllers for diesel engines — part i: Approach and drive cycle results," in *IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE, 2017.

[173] R. Shepard, S. R. Brozell, and G. Gidofalvi, "The representation and parametrization of orthogonal matrices," *The Journal of Physical Chemistry A*, vol. 119, no. 28, pp. 7924–7939, 2015.

[174] R. Shridhar and D. J. Cooper, "A tuning strategy for unconstrained SISO model predictive control," *Industrial & Engineering Chemistry Research*, vol. 36, no. 3, pp. 729–746, 1997.

[175] M. Simchowitz, H. Mania, S. Tu, M. I. Jordan, and B. Recht, "Learning without mixing: Towards a sharp analysis of linear system identification," in *Conference On Learning Theory*, 2018, pp. 439–473.

[176] R. Soeterboek, *Predictive Control: A Unified Approach*. Prentice Hall, 1992.

[177] J. C. Spall, *Introduction to Stochastic Search and Optimization*. Wiley-Interscience, 2003.

[178] B. Storvik, G. Storvik, and R. Fjortoft, "On the combination of multisensor data using meta-gaussian distributions," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, no. 7, pp. 2372–2379, 2009.

[179] C.-L. Sung, R. B. Gramacy, and B. Haaland, "Exploiting variance reduction potential in local gaussian process search," *Statistica Sinica*, 2018.

[180] R. Suzuki, F. Kawai, H. Ito, C. Nakazawa, Y. Fukuyama, and E. Aiyoshi, "Automatic tuning of model predictive control using particle swarm optimization," in *IEEE Swarm Intelligence Symposium*. IEEE, 2007.

[181] R. Tempo, E. W. Bai, and F. Dabbene, "Probabilistic robustness analysis: explicit bounds for the minimum number of samples," in *35th IEEE Conference on Decision and Control*. IEEE, 1996.

[182] R. Tempo, G. Calafiore, and F. Dabbene, *Randomized Algorithms for Analysis and Control of Uncertain Systems*, 1st ed. Springer London, 2005.

[183] ——, *Randomized Algorithms for Analysis and Control of Uncertain Systems With Applications*, 2nd ed. Springer, 2013.

[184] L. L. Thurstone, "A law of comparative judgment," *Psychological Review*, vol. 34, no. 4, pp. 273–286, 1927.

[185] R. Toth, *Modeling and Identification of Linear Parameter-Varying Systems*. Springer, 2010.

[186] K. Train, *Qualitative Choice Analysis: Theory, Econometrics, and an Application to Automobile Demand*. MIT Press, 1986.

[187] K. E. Train, *Discrete Choice Methods with Simulation*. Cambridge University Press, 2009.

[188] Q. N. Tran, R. Octaviano, L. Ozkan, and A. Backx, "Generalized predictive control tuning by controller matching," in *American Control Conference*. IEEE, 2014.

[189] C.-C. Tsai, F.-J. Teng, and S.-C. Lin, "Direct self-tuning model following predictive control of a variable-frequency oil-cooling machine," in *American Control Conference*. IEEE, 2003.

[190] K. Tsukida and M. R. Gupta, "How to analyze paired comparison data," University of Washington, Tech. Rep., 2011.

[191] E. Tzirakis, K. Pitsas, F. Zannikos, and S. Stournas, "Vehicle emissions and driving cycles: comparison of the athens driving cycle (ADC) with ECE-15 and european driving cycle (EDC)," *Global NEST Journal*, vol. 8, no. 3, pp. 282–290, 2006.

[192] M. Vallerio, J. V. Impe, and F. Logist, "Tuning of NMPC controllers via multi-objective optimisation," *Computers & Chemical Engineering*, vol. 61, pp. 38–50, 2014.

[193] M. J. van Nieuwstadt, I. V. Kolmanovsky, P. E. Moraal, A. Stefanopoulou, and M. Jankovic, "EGR-VGT control schemes: experimental comparison for a high-speed diesel engine," *IEEE Control Systems Magazine*, vol. 20, no. 3, pp. 63–79, 2000.

[194] P. Vega, M. Francisco, and F. Tadeo, "Multiobjective optimization for automatic tuning of robust model based predictive controllers," in *17th IFAC World Congress*, 2008.

[195] M. Vidyasagar, "Statistical learning theory and randomized algorithms for control," *IEEE Control Systems Magazine*, vol. 18, no. 6, pp. 69–85, 1998.

[196] ——, "Randomized algorithms for robust controller synthesis using statistical learning theory," *Automatica*, vol. 37, no. 10, pp. 1515–1528, 2001.

[197] J. von Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*. Princeton University Press, 1955.

[198] J. Wahlstrom and L. Eriksson, "Modelling diesel engines with a variable-geometry turbocharger and exhaust gas recirculation by optimization of model parameters for capturing non-linear system dynamics," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 225, no. 7, pp. 960–986, 2011.

[199] J. Wahlstrom, "Control of egr and vgt for emission control and pumping work minimization in diesel engines," phdthesis, Linkoping University, 2009.

[200] H. Waschl, D. Alberer, and L. del Re, "Numerically efficient self tuning strategies for MPC of integral gas engines," in *18th IFAC World Congress*, vol. 44, no. 1. Elsevier BV, 2011, pp. 2482–2487.

[201] ——, "Automatic tuning methods for MPC environments," in *Computer Aided Systems Theory EUROCAST 2011*. Springer Berlin Heidelberg, 2012, pp. 41–48.

[202] C. Wirth, R. Akrour, G. Neumann, and J. Fürnkranz, "A survey of preference-based reinforcement learning methods," *Journal of Machine Learning Research*, vol. 18, no. 136, pp. 1–46, 2017.

[203] H. S. Witsenhausen, "A counterexample in stochastic optimum control," *SIAM Journal on Control*, vol. 6, no. 1, pp. 131–147, 1968.

[204] W. Wojsznis, J. Gudaz, T. Blevins, and A. Mehta, "Practical approach to tuning MPC," *ISA Transactions*, vol. 42, no. 1, pp. 149–162, 2003.

[205] M.-W. Wu, Y. Li, M. Gurusamy, and P.-Y. Kam, "A tight lower bound on the gaussian Q-function with a simple inversion algorithm, and an application to coherent optical communications," *IEEE Communications Letters*, vol. 22, no. 7, pp. 1358–1361, 2018.

[206] A. S. Yamashita, A. C. Zanin, and D. Odloak, "Tuning the model predictive control of a crude distillation unit," *ISA Transactions*, vol. 60, pp. 178–190, 2016.

[207] M. Yang and L. Lee, "Ordinal optimization with subset selection rule," *Journal of Optimization Theory and Applications*, vol. 113, no. 3, pp. 597–620, 2002.

[208] X. Zeng, J. Ren, M. Sun, S. Marshall, and T. Durrani, "Copulas for statistical signal processing (part II): Simulation, optimal selection and practical applications," *Signal Processing*, vol. 94, pp. 681–690, 2014.

[209] F. Zhang, J. Cao, K. Hwang, K. Li, and S. U. Khan, "Adaptive workflow scheduling on cloud computing platforms with iterative ordinal optimization," *IEEE Transactions on Cloud Computing*, vol. 3, no. 2, pp. 156–168, 2015.

[210] Y. Zhang, G. Lu, H. Xu, and Z. Li, "Tuneable model predictive control of a turbocharged diesel engine with dual loop exhaust gas recirculation," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 232, no. 8, pp. 1105–1120, 2017.

[211] Y. Zhang, Q. Zhou, Z. Li, J. Li, and H. Xu, "Intelligent transient calibration of a dual-loop EGR diesel engine using chaos-enhanced accelerated particle swarm optimization algorithm," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 233, no. 7, pp. 1698–1711, 2018.