Data-efficiënt machinaal leren
voor het ontwerp en de optimalisatie van complexe systemen

Data-Efficient Machine Learning for Design and Optimisation of Complex Systems

Joachim van der Herten

Promotoren: prof. dr. ir. T. Dhaene, dr. ir. I. Couckuyt
Proefschrift ingediend tot het behalen van de graad van
Doctor in de ingenieurswetenschappen: computerwetenschappen

UNIVERSITEIT
GENT

Vakgroep Informatietechnologie
Voorzitter: prof. dr. ir. B. Dhoedt
Faculteit Ingenieurswetenschappen en Architectuur
Academiejaar 2016 - 2017

ISBN 978-90-8578-996-3
NUR 984, 919
Wettelijk depot: D/2017/10.500/31

UNIVERSITEIT GENT

FACULTEIT INGENIEURSWETENSCHAPPEN EN ARCHITECTUUR

Promotoren:	prof. dr. ir. Tom Dhaene - Vakgroep Informatietechnologie
	dr. ir. Ivo Couckuyt - Vakgroep Informatietechnologie
Leden examencomissie:	em. prof. dr. ir. Daniël De Zutter (voorzitter) - Vakgroep Informatietechnologie
	PD dr. David Ginsbourger - Idiap Research Institute and University of Bern
	dr. ir. Dirk Gorissen - Oxbotica
	prof. dr. Bart Goethals - Universiteit Antwerpen
	prof. dr. Dirk Van den Poel - Faculteit Economie en Bedrijfskunde, Vakgroep Marketing
	prof. dr. ir. Joris Degroote - Vakgroep Mechanica van Stroming, Warmte en Verbranding
	prof. dr. ir. Eric Laermans (secretaris) - Vakgroep Informatietechnologie

Universiteit Gent
Faculteit Ingenieurswetenschappen en Architectuur

Vakgroep Informatietechnologie
Technologiepark 15, B-9052 Gent, België

Tel.: +32-9-331.49.00
Fax.: +32-9-331.48.99

"Stel niet uit tot morgen wat je vandaag kan doen."

— Barbara Timmer

Dankwoord

Dit boek is het resultaat van vier jaar intensief werk en een aaneenschakeling van momenten van succes en verwezelijkingen, maar ook diepe dalen en tegenslagen. Het spreekt dan ook voor zich dat zonder alle ondersteuning die ik kreeg van mijn omgeving dit werk nooit tot stand had kunnen komen. Vooraleerst wil ik mijn promotor Tom Dhaene bedanken voor alle geboden kansen en opportuniteiten waaronder de mogelijkheid om aan een doctoraat te beginnen alsook al het geduld, de begeleiding en het advies gedurende het traject. Tussen alle projecten en de hoge verwachtingen daaraan verbonden creëert hij een omgeving voor zijn onderzoekers waar ideëen in alle rust kunnen groeien en soms ook kunnen mislukken. Binnen deze omgeving had ik de mogelijkheid om veel kennis te vergaren en te groeien, zowel als wetenschapper en als mens. Verder kreeg ik de kans om meerdere conferenties te bezoeken en zo contacten te leggen met experts binnen verschillende domeinen. Zo had ik de kans om ICML en NIPS bij te wonen, twee top evenementen binnen *machine learning*.

Ook dank aan mijn copromotor Ivo Couckuyt. Vanaf het begin stond hij paraat met raad en daad, met een bijzondere invalshoek. Dit hielp enorm om mijn kennis te verbreden en te verdiepen. De vele lange discussies en gedachtenwisselingen hebben geleid tot een geleidelijke transitie van de software development binnen SUMOLab zonder grote disupties. Met de geplande projecten zal ook de visibiliteit voor het onderzoek verzekerd zijn voor de komende jaren.

Mijn dank gaat uit naar de voorzitter van de examencomissie Daniël De Zutter, en de andere alle leden van de comissie: David Ginsbourger en Dirk Gorissen om over te komen voor de interne verdediging, alsook hun gedetailleerde lezing met aandacht voor alle aspecten, inclusief de gehanteerde wiskundige formalismen. Speciale dank ben ik ook verschuldigd aan Dirk Van den Poel voor zijn nauwgezette lezing en de fouten die zo werden gevonden. Ook de opmerkingen van Joris Degroote, Bart Goethals en Eric Laermans waren terecht en hebben bijgedragen om dit werk naar een hoger niveau te tillen.

Ook mijn collega's wil ik uitdrukkelijk bedanken. Een voor een zijn ze bijzonder

intelligent en een grote hulp geweest. Ik beschouw het als een privilege om met hen samen te werken: alle discussies, alsook de gedeelde passie voor informatietechnologie, ingenieurswetenschappen en kunstmatige intelligentie waren een onuitputtelijke bron van inpisratie. In het bijzonder wil ik Dirk Deschrijver bedanken voor de interessante gedachtenwisselingen en zijn feedback. Veel dank ook aan alle huidige en voorgaande leden van SUMOLab: Francesco Ferranti, Domenico Spina, Krishnan Cheriya, Elizabeth Samuel, Selvakumar Ulaganathan, Prashant Singh, Keiichi Ito, Sean Stijven, Joeri Ruyssink, Sofie van Gassen, Leen De Baets, Arun Kaintura, Yinghao Ye, Tom Van Steenkiste, Nicolas Knudde, Roberto Medico, Diego Nieves en Kyle Foss. Onze conversaties waren cruciaal om vooruit te komen. De toffe werksfeer in het kantoor maakte ook alle kinderziektes draaglijk die we tegenkwamen bij de verhuis naar iGent.

Verder ben ik ook dank verschuldigd aan de collega's van het bureau op de Zuiderpoort: Matthias Strobbe, Jolien Coenraets, Heiko Desruelle, Dieter Blomme, Pieter-Jan Maenhaut, Elias De Coninck, Rein Houthooft, Cedric De Boom, Jeroen van der Hooft en Lander Van Herzeele. Verder ben ik ook erg dankbaar voor de hulp van cloudlet leden Steven Bohez en Tim Verbelen met enkele topics binnen het kader van het *Internet of Things* onderzoek.

Ik ben veel verschuldigd aan mijn echtegenote Cara Smits. Zij stond altijd klaar om mij op te vangen wanneer zaken niet volgens plan verliepen. Ik kan haar niet genoeg bedanken voor alle liefde, steun en het geduld dat ik van haar kreeg tijdens de voorbije jaren. Vaak zat ik op mijn laptop 's avonds bezig om enkele dingen af te krijgen of nog snel enkele dingen te proberen. Ze hield ook ons huishouden overeind terwijl ik in het buitenland verbleef tijdens conferenties, en deed veel moeite om teksten na te lezen. Verder heb ik ook altijd kunnen rekenen op mijn familie, en in het bijzonder mijn ouders Frank van der Herten en Mary Kempen die altijd voor mij hebben gevochten. Zowel tijdens mijn zware ziekten als de moeilijke perioden op school. Zelfs nu, ondanks dat ze niet altijd begrijpen waar ik mee bezig ben: ze hebben er altijd voor mij gestaan en hebben geprobeerd om mij zo goed als mogelijk bij te staan, ook als oplossingen niet direct voorhanden waren.

Ik zou ook mijn vrienden willen bedanken die ik leerde kennen tijdens mijn studies aan de Universiteit Antwerpen of terwijl ik in het buitenland zat. In het bijzonder Andie Similon, Evi De Cock, Simon van Mierlo, Michiel Vermeersch, Kevin Madsen, Margaux Dumon, Tim Goovaerts, Karel Crombecq, Wim Kegels en vele anderen die ik leerde kennen bij WINAK en EESTEC.

Ik ben er zeker van dat ik nog velen tekort doe door hen niet te vermelden. Er zijn nog alle mensen die ik tegenkwam op conferenties met wie ik zeer nuttige conversaties had. Vaak was hun enthousiasme en passie noodzakelijk om mij de waarde te doen inzien van onderzoek, alsook mijn eigen bijdrage daaraan.

Gent, Mei 2017
Joachim van der Herten

Table of Contents

List of Figures

List of Tables

List of Acronyms

A

AAE	Average Absolute Error
AGPL	Affero General Public License
AI	Artificial Intelligence
AIC	Akaike Information Criterion
ANN	Artificial Neural Network
ARD	Automatic Relevance Determination
ARE	Average Relative Error
ASK	Adaptive Sampling Kit

B

BALD	Bayesian Active Learning by Disagreement
BEEQ	Bayesian Error Estimation Quotient
BO	Bayesian Optimisation

C

CAD	Computer Aided Design
CAE	Computer Aided Engineering
CART	Classification And Regression Trees

CG Conjugate Gradient

CFD Computational Fluid Dynamics

CI Confidence Intervals

CORS Constrained Optimisation using Response Surfaces

CPU Central Processing Unit

D

DEML Data-Efficient Machine Learning

DIRECT Dividing Rectangles

DHASD Delaunay-Hybrid Adaptive Sequential Design

DoE Design of Experiments

DUT Device Under Test

E

ED Expected Decrement

EDSD Explicit Design Space Decomposition

EGO Efficient Global Optimisation

EI Expected Improvement

ELM Extreme Learning Machines

EMC Electromagnetic Compatibility

ERM Empirical Risk Minimization

ES Entropy Search

F

FEM Finite Element Methods

FIS Fuzzy Inference System

FLOLA Fuzzy Local Linear Approximation

FPR False Positive Rate

G

GA Genetic Algorithm

GAN Generative Adversarial Network

GAE Geometric Average Error

GP Gaussian Process

GPLVM Gaussian Process Latent Variable Model

GPSS General Purpose Systems Simulator

GPU Graphical Processing Unit

H

HAE Harmonic Average Error

HPC High Performance Computing

Hv Hypervolume

HvPoI Hypervolume Probability of Improvement

HvEI Hypervolume Expected Improvement

I

IR Instant Regret

K

KGCB Knowledge-Gradient with Correlated Beliefs

KGCP Knowledge-Gradient for Continuous Parameters

L

LASSO Least Absolute Shrinkage and Selection Operator

LCB Lower Confidence Bound

LHD Latin Hypercube Design

LNA Low-noise amplifier

LOLA Local Linear Approximation

LRM Linear Reference Model

LS-SVM Least-Squares Support Vector Machine

LVM Latent Variable Model

M

MAE Maximum Absolute Error

MC Monte Carlo

ME Model Error

MCMC Markov Chain Monte Carlo

ML Machine Learning

MLE Maximum Likelihood Estimation

MOR Model Order Reduction

MOSBO Multi-Objective Surrogate Based (Bayesian) Optimisation

MOSFET Metal–Oxide–Semiconductor Field-Effect Transistor

MRE Maximum Relative Error

MRS Minimum Regret Search

MSE Mean Square Error

MVN Multivariate Normal distribution

MVT Multivariate Student-t distribution

N

NF Near-Field

N-V Neighbourhood-Voronoi

O

OC Opportunity Cost

P

PCA Principal Component Analysis

PCB Printed Circuit Board

PES Predictive Entropy Search

PSO Particle Swarm Optimisation

PoI Probability of Improvement

PoF Probability of Feasibility

Q

Q-MC Quasi-Monte Carlo

R

RBF Radial Basis Function

RF Random Forests

RFC Radio-Frequency Circuit

RMSE Root-Mean-Square Error

ROC Receiver Operating Characteristic

RRSE Root-Relative-Square Error

RSM Response Surface Model

S

SA Simulated Annealing
SBO Surrogate-Based Optimisation
SEED Sequential Exploratory Experimental Design
SGD Stochastic Gradient Descent
SKO Sequential Kriging Optimisation
SOM Self-Organizing Map
SQP Sequential Quadratical Programming
SUMO SUrrogate MOdeling
SVM Support Vector Machine

T

TP Student-t Process
TPLHD Translational Propagation Latin Hypercube Design
TPR True Positive Rate

U

UCB Upper Confidence Bound

X

XML eXtensible Markup Language

Nederlandse samenvatting –Summary in Dutch–

Titel: Data-efficiënt machinaal leren voor het ontwerp en de optimalisatie van complexe systemen

Binnen de (ingenieurs-)wetenschappen is het soms onmogelijk of bijzonder onpraktisch om experimenten fysiek uit te voeren. Een fysiek experiment brengt steeds een zekere kost met zich mee in termen van tijd of gebruik van materialen en toestellen. Om deze kost binnen de perken te houden wordt tegenwoordig frequent gebruik gemaakt van geavanceerde computermodellen of simulaties: deze werkwijze is veel sneller, ongevaarlijk en vereist enkel hard- en software om deze virtuele experimenten uit te voeren.

De computersimulaties zijn typisch geparametriseerd door een set inputparameters. Indien een combinatie opgegeven waarden voor deze parameters wordt geëvalueerd resulteert dit in één of meerdere output waarden. Deze transformatie van input naar output kan verder worden bestudeerd door de inputwaarden te variëren en de output te observeren. Zodoende kan men het onderliggende systeem beter begrijpen aan een lagere kost. Bijkomend voordeel is dat het voor wetenschappers veel eenvoudiger is om de omstandigheden van virtuele experimenten te controleren. De laatste decenia heeft deze evolutie snellere en goedkopere innovatie mogelijk gemaakt en werd de afstand van idee tot de markt aanzienlijk verkleind.

Doorheen de jaren is de simulatiesoftware steeds accurater geworden, hetgeen een sterke impact heeft gehad op de tijd om één enkele simulatie te evalueren. Het is vandaag de dag niet uitzonderlijk dat een simulatie meerdere uren, dagen of zelfs weken kan duren hetgeen het bestuderen van het input/output gedrag bemoeilijkt of zelfs compleet onmogelijk maakt. Daarom wordt tegenwoordig vaak de simulatie zelf, een approximatie van een systeem of proces uit de echte wereld, benaderd door een wiskundig model dat wel snel en eenvoudig kan worden geëvalueerd. Dit metamodel staat bekend als surrogaatmodel en kan de plaats van de simulatie overnemen voor verdere analyse en onderzoek. De kunst bestaat erin om het surrogaatmodel zo accuraat mogelijk te krijgen als nodig voor de toepassing, zonder hierbij al te frequent beroep te moeten doen op computationeel dure simulaties. Doorheen het proces om een surrogaatmodel te bekomen zijn meerdere aspecten belangrijk: zowel het standpunt van de gebruiker (wetenschappers en ingenieurs)

als technische elementen spelen hierbij een rol.

Surrogaatmodellen zijn erg flexibel en kunnen worden toegepast in sterk uiteenlopende onderzoeksdomeinen, bv. werktuigbouwkundem electrotechniek of hydrologie. Zowel de keuze van de juiste approximatie, de optimalisatie van parameters, als de keuze van inputcombinaties die dienen te worden gesimuleerd hebben een sterke impact op het verloop van het modelleringsproces en de uiteindelijke kwaliteit van het surrogaatmodel. Hoewel de complexiteit het onmogelijk maakt om al deze modelleringskeuzes exhaustief te onderzoeken maakt de snelle vooruitgang in Artificiële Intelligentie een (deels) automatische afhandeling mogelijk, puur gebaseerd op data beschikbaar over het input/output gedrag. Dit onderzoeksdomein is dan ook de primaire focus van dit werk en kent brede toepassingsmogelijkheden binnen ingenieurswetenschappen alsook daarbuiten.

Een gedetailleerd overzicht van de concepten “model” en “approximatie” en toepassingen is het onderwerp van de introductie van dit werk en wordt gevolgd door een overzicht van alle aspecten en keuzes die zich opdringen bij de ontwikkeling van een surrogaatmodel in Hoofdstuk 2.

In dit werk wordt gebruik gemaakt van sequentieel ontwerp: hierbij worden de simulaties niet allemaal vooraf gekozen maar worden (1) simulatie, (2) approximatie en (3) de keuze van extra input combinaties afgewisseld. Dit laat toe om de kennis vergaard bij voorgaande iteraties en de inzichten van eerdere approximaties aan te wenden bij de selectie van één of meerdere nieuwe input combinaties. Vooraleerst wordt dieper ingegaan op simulaties waarbij de output slechts enkele waarden kan aannemen (discreet) in plaats van een continu interval, de traditionele veronderstelling van surrogaatmodellering. Hiervoor worden **classificatie** approximaties gebruikt in plaats van regressiemodellen. Door deze uitbreiding worden surrogaatmodellen bruikbaar voor haalbaarheidsstudies waarbij een gegeven set inputparameters aan een aantal specificaties moet voldoen, of optimalisatietoepassingen waarbij een of meerdere beperkingen in acht moeten genomen worden die ook door middel van simulatie moet worden bepaald. Bijgevolg wordt de toepasbarheid van surrogaatmodellen sterk vergroot. Hoofdstuk 3 bevat ook een breed overzicht van bestaande technieken voor sequentieel ontwerp met bijzondere aandacht voor technieken die overweg kunnen met discrete output.

Hierna volgen enkele significante verbeteringen aan technieken voor **sequentieel ontwerp**. Eerst wordt de complexiteit van een succesvolle LOLA-Voronoi algoritme voor keuze van nieuwe simulaties om zo snel mogelijk een globaal accuraat model te bekomen gevoelig verbeterd. Het uitgangsprincipe van dit algoritme is om moeilijk outputgedrag meer in detail te onderzoeken in vergelijking met eenvoudiger outputgedrag. De stappen van het algoritme verantwoordelijk voor het maken van dit onderscheid worden aanzienlijk duurder naarmate een probleem meer input parameters heeft. In Hoofdstuk 4 wordt het beslissingsproces binnen deze stappen verbeterd door gebruik te maken van Fuzzy logic om dit nadeel te elimineren zonder hierbij in te boeten op de performantie op vlak van benodigde simulaties om een accuraat model te verkrijgen. Zodoende laat deze ontwikkeling adequate en effi-

ciënte selectie van simulaties toe, zelfs met veel input parameters. Deze oplossing is erg generiek en is breed toepasbaar, zowel binnen ingenieurswetenschappen als binnen machinaal en actief leren.

Een tweede verbetering aan een techniek voor sequentieel ontwerp situeert zich binnen het domein **optimalisatie**. Ook hierbij is de input/output transformatie belangrijk: optimalisatie zoekt immers de de inputwaarden die resulteren in een (globaal) optimum van de output. Dit wordt een bijzonder moeilijke opdracht als elke evaluatie computationeel duur of moeilijk te verkrijgen, bijgevolg is dit een erg belangrijk toepassingsdomein voor surrogaatmodellen. In dit werk wordt de toepasbaarheid van de **"kennisgradiënt"** techniek onderzocht. De kennisgradiënt tracht een surrogaatmodel snel in staat te stellen optima te identificeren, zelfs indien de output onderhevig is aan onzekerheid (stochastisch). Wanneer de output deterministisch is (zoals doorgaans het geval bij simulaties) blijft de kennisgradiënt toepasbaar maar is ze computationeel veel minder efficiënt dan andere algoritmen zoals "expected improvement". Met dit algoritme bestaat wel een relatie die in Hoofdstuk 5 verder wordt uitgewerkt, steunende op de assumptie van deterministische output waarden. Dit resulteert in een nieuwe equivalentierelatie die toelaat om (1) de kennisgradiënt zeer efficiënt te berekenen en (2) een eenvoudige interpretatie toelaat. Experimentele validatie toont aan dat vooral optimalisatie met complexer outputgedrag gebaat is met de kennisgradiënt en snellere identificatie van optima toelaat.

Bijkomend bevat dit werk twee appendices waarin enerzijds wordt aangetoond dat een recente approximatietechniek genaamd "Student-t processen", efficiënt kan worden gebruikt voor optimalisatie waarbij meerdere doelstellingen een rol spelen. Anderzijds wordt getoond hoe de methodologieën die worden toegepast binnen het surrogaatmodelleren rechtstreeks kunnen worden toegepast voor automatische approximatie van complexe systemen waarbij het dure computermodel wordt overgeslagen. Hierbij moet dan wel rekening worden gehouden met onzekerheid op de geobserveerd outputwaarden.

English summary

Title: Data-Efficient Machine Learning for Design and Optimisation of Complex Systems

Within science and engineering it can be impossible or particularly impractical to perform experiments physically in the real world. A physical experiment always involves a certain cost in terms of time or consumption of materials and devices. To reduce these costs nowadays advanced computer models or simulations are frequently used: this approach is a lot faster, not dangerous and relies only on the availability of software and hardware in order to be able to perform virtual experiments.

Computer simulations are typically parametrised by a set of input parameters. When specified values for the input parameters are evaluated, this yields one or multiple output values. This mapping of input to output can be studied in more detail by varying the inputs and observing the output. This results in a better understanding of the underlying system without the need of big investments. Additionally, it is a lot easier for scientists to control the environment of virtual experiments. Over the last decades, this evolution has resulted in cheaper innovation and a shorted time-to-market.

Throughout the years, the accuracy of computer simulations has increased significantly, which has had a strong impact on the time required to evaluate a single simulation. Today, it is not uncommon that a single simulation lasts for hours, days or even weeks which makes studying the input-output behaviour troublesome, if not infeasible. Nowadays the simulator itself, essentially an approximation of a system in the real world, is therefore approximated by a mathematical model which can be evaluated fast and easily. One approach to obtain this approximation is known as a surrogate modeling (also called response surface model, metamodel or replacement model). The surrogate model may then replace the simulator for further analysis and research. The difficulty is being able to make the surrogate model as accurately as possible for the application, without frequently evaluating the computationally expensive simulation. Throughout the surrogate model development process, several aspects are important both from the point of view of the model user (scientists and engineers) as well as technical elements.

Surrogate models are very flexible and can be applied to a wide variety of research domains, e.g. ranging from mechanical engineering to hydrology. This makes surro-

gate modelling itself a very challenging and exciting multidisciplinary research area. The choice of proper approximation methodology, the optimisation of parameters as well as the choice of simulations have a strong influence on the quality and usability of the final surrogate model. Although this complexity and choices make it impossible to be fully understood by a programmer, the strong developments within the field of artificial intelligence and machine learning allow an automatic approach, based purely on available data on the input/output behaviour. This research area is the primary focus of this work and is broadly applicable within engineering as well as other fields.

An in-depth overview of the concepts *model* and *approximation* and application domains is the subject of the introduction, and is followed by an overview of all aspects and choices appearing during the development of a surrogate model in Chapter 2.

In this work sequential design is applied: within this context the choice of simulations is not specified entirely upfront: rather, (1) simulation, (2) approximation, and (3) the choice of additional simulations are alternated. This allows exploitation of knowledge acquired during previous evaluations and the insights obtained from previous approximations during the selection of one or more input combinations. Throughout this process the focus is always on the intended use of the surrogate model. In Chapter 3, this work studies the case of simulations emitting only a limited (discrete) set of output values rather than continuous intervals as the typical assumption for surrogate modelling. For this, **classification** approximations are used instead of regression models. This broadens the applicability of surrogate models to feasibility studies for which a given set of input parameters must satisfy specifications, or optimisation with a number of constraints which must also be evaluated by means of simulation. This chapter also includes a broad overview of existing techniques for sequential design, with specific attention for techniques able to handle discrete outputs.

Next, in Chapters 4 and 5 improvements of techniques for **sequential design** are presented. First the computational complexity of the successful LOLA-Voronoi algorithm for choice of simulations aiming at global surrogate model accuracy is improved significantly. The basic principle of this algorithm is that difficult output behaviour needs to be analysed further than simple output behaviour. However, distinguishing amongst the output dynamics involve some the computationally intensive steps which scale badly with the as the number of input parameters increases. By improving the decision process within these steps by applying Fuzzy logic, this disadvantage may be lifted without reducing the performance of the algorithm in terms of required number of evaluations to obtain an accurate surrogate model. This development allows adequate and efficient selection of simulations, even when many input parameters are present. The solution is generic and broadly applicable within engineering as within machine learning and active learning.

A second improvement to a technique for sequential design is situated within the **optimisation** domain. Within this context, the input/output is also important as optimisation aims to find the input values resulting in (global) optima of the output. This becomes very challenging if each evaluation is computationally expensive or hard to obtain, hence this is another successful application domain for surrogate modelling. In this work the applicability of the **knowledge-gradient** is studied. The knowledge-gradient aims to enable the approximation to be able to identify optima even when the output is subject to uncertainty (stochastic). In case the output is deterministic instead, the knowledge-gradient remains applicable, however it is less computationally efficient compared to other methods such as the Expected Improvement. However, a relation exists with this methods which is further developed, based on the deterministic assumption. This results in an equivalence relation which provides a method for (1) efficient calculation and (2) easier interpretation of the knowledge-gradient. Experimental validation shows how optimisation with complex output behaviour benefits from the knowledge-gradient for faster identification of optima.

Finally, this work includes two appendices: the first studies the applicability of Student-t Processes, a recent approximation technique, for optimisation with multiple objectives. The second appendix shows how the methodologies applied within the context of surrogate modelling may be applied directly for data-efficient approximations of complex systems, without the intermediate layer of computer modelling. This situation however requires the uncertainty on the observed output.

Notation

Indexing

Vectors

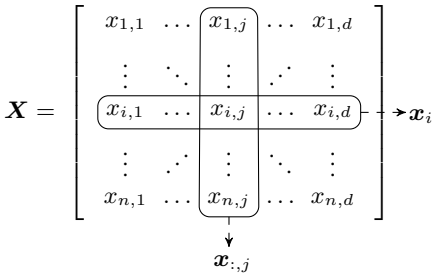
A vector $\boldsymbol{x} \in \mathbb{R}^d$ is indexed using normal font and single numeric indices:

$$\boldsymbol{x} = \left[x_1, x_2, \ldots, x_d\right]$$

Matrices

A matrix $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ is denoted in capitals and boldface.

- $\boldsymbol{x}_i \in \mathbb{R}^n$ represents the i-th row
- $\boldsymbol{x}_{:,i} \in \mathbb{R}^d$ represents the i-th column
- $x_{i,j}$ represents a single element at the i-th row and j-th column



Common symbols

Functions

Λ	Model quality estimator.
Φ	Standard cumulative probability density function.
α_n^{EI}	Expected Improvement for maximization.
α_n^{HvPoI}	Hypervolume Probability of Improvement.

α_n^{KG}	Knowledge-Gradient with Correlated Beliefs for discrete optimization.
α_n^{PoF}	Probability of Feasibility.
$\bar{\alpha}_n^{\text{KG}}$	Knowledge-Gradient for Continuous Parameters.
ℓ	Loss.
γ	Mean function.
κ	Error function (i.e. RMSE, not erf).
μ	Mean of GP/TP predictive distribution.
ϕ	Standard normal probability density function.
$\tilde{f}$	Approximation function (surrogate model).
f	Unknown (expensive) function.
k	Covariance function (kernel).
l_1	Manhattan distance.
l_2	Euclidean distance.
p	Probability function.
r	Correlation function.
s^2	Variance of GP predictive distribution.
s_{tp}^2	Variance of TP predictive distribution.

Algebra

X	Input data x collected in a set.
$\mathcal{P}$	Power set.
$\mathcal{X}$	Data input domain.
$\tilde{F}$	Predictive random variable.
$\boldsymbol{B}$	Regression matrix.
$\boldsymbol{F}$	Output data, organized as matrix (points $\times$ output dimension).
$\boldsymbol{I}$	Identity matrix.
$\boldsymbol{K}$	Covariance matrix.
$\boldsymbol{R}$	Correlation matrix.
$\boldsymbol{X}$	Input data organized as matrix (points $\times$ input dimension).

$\boldsymbol{Y}$	Output data, possibly noise corrupted, organized as matrix (points $\times$ output dimension).
$\boldsymbol{\gamma}$	Vector of means.
$\hat{\boldsymbol{x}}$	Location of the optimal decision in $\mathcal{X}$.
$\boldsymbol{\omega}$	Fuzzy neighbour weights.
$\boldsymbol{\theta}$	Vector of hyperparameters.
$\boldsymbol{f}$	In case of single output: a (column) vector collecting the real evaluations of f for all input points. In case of multiple outputs, evaluations for one input point as row vector..
$\boldsymbol{g}$	Gradient estimation.
$\boldsymbol{w}$	Model parameters.
$\boldsymbol{x}_\star$	Arbitrary input point $\in \mathcal{X}$.
$\boldsymbol{x}$	Input data point $\in \mathcal{X}$ (represented as row vector).
$\boldsymbol{y}$	In case of single output: a (column) vector collecting (noise corrupted) responses for all input points. In case of multiple outputs, noise corrupted outputs for one input point as row vector..

Constants

N_{init}	Size of the initial design.
N_{max}	Maximum number of evaluations.
N_{new}	Number of new samples selected in one iteration of a sampling policy.
λ	Regularization constant.
ν	Degrees of freedom.
σ^2	Signal variance.
σ_n^2	Noise variance.
d	Data input dimension.
n	Number of data points.
q	Data output dimension.

"Science is not an opinion."

— Jeff Schweitzer

1 Introduction

1.1 Introduction

For thousands of years, humans have always tried to describe the world they were living in and its natural phenomena. Moving away from religious explanations to more scientific descriptions introduced concepts such as prediction and replication. Whereas the last centuries were strongly focussed on theoretical foundations of science, the introduction of computers sparked technological innovation during the last decades, increasing the complexity of studied problems and phenomena. This caused an explosion of data and tools that are currently being unified into, what is often referred to as, e-Science [1]. Given this background, this chapter describes the problem domain in depth and arrives at the scope of this thesis (surrogate models).

1.2 Models: what and why?

Ever since the appearance of the Homo Sapiens on the surface of Earth, our species has been using abstractions of the real world in order to reason about it, as well as the phenomena that occur in it. In fact, anthropologists believe this ability is one

of the competitive advantages of the human species over others [2]. Combining these abstractions with observations resulted in the development of the first models, sparking innovation. Long before the era of ancient Greek models were used, with the appearance of numbers around 30000BC as one of the earliest, followed by development of mathematical formulas and theorems for solving specific problems. An excellent introduction to the history of modelling is given by Schichl [2]. Note that some famous models were far ahead of their time, such as the computation of the earth diameter by Eratosthenes, long before civilisations gave up on the flat earth model. Yet, these models were often of remarkable quality and were only validated experimentally decades or even centuries later. Even today, some parts of Einsteins relativity theory, are still being experimentally validated. Other famous models include the series of atom models (Dalton, Thomson, Rutherford, and Bohr) and of the solar system (Keppler, Hubble, Newton, and others).

In general the rationale behind models is a direct consequence of the need for understanding, explanation, reproduction and/or prediction of real world phenomena. Fundamentally it represents knowledge in a structured way, while defining preconditions and constraints. After studying the model further it may be used to derive and prove properties. Because models are by definition a simplification it is important to keep in mind there is a degree of uncertainty associated with the model which may limit its applicability. This aspect is not easily quantifiable, which is experienced frequently with weather prediction. Typical use cases for models include:

- Interpolation for missing measurements/observations
- Extrapolation & prediction
- Decision making (e.g. demographic statistics for political decision support)
- Communication (graphs, surface plots, ...)
- Handling higher dimensions (and ideally reduction of dimensions)
- Generate data (e.g. generative models)

Depending on the field of science, the exact definition and implementation of the *model* concept may vary. The interpretation of mathematical models in physics, demographic models, machine learning models, or software models is quite different. In general, this dissertation focusses on models mapping a set of inputs to one or multiple outputs.

1.3 Computer modelling

For centuries, modelling was a task performed exclusively by humans assisted by no more than pencil and paper and very little existing literature or means of communication with peers. The situation changed after the introduction of mechanical devices and was further revolutionised by the introduction of computer systems. The latter enabled scientists to perform experiments in highly controlled virtual environments using no more than hardware and computer models. These models are typically referred to as *simulators*. This marked a complete paradigm shift, as the breadth and depth at which phenomena could be studied and predicted increased tremendously. Even when analytical solutions to problems were infeasible before, computer simulation offered suitable approximations.

A remarkable first success was already achieved as part of the Manhattan Project during World War II [3], however due to the complexity of running algorithms on the computer systems used as well as their limited processing power, too much expertise was required to justify commercialisation and widespread usage. It took years of development (both for hardware and software) for the merits of computer simulation to become more widely available. During the 1960s the General Purpose Systems Simulator (GPSS) [4] package and the Simula programming language marked the start of acceptance and strong interest in simulation. During this decade, the first Winter Simulation Conference[1] took place which still exists and celebrates its 50th anniversary in 2017.

1.3.1 Computer modelling process

The process of developing computer models was discussed in detail by Kleijnen [5]. However, this section presents the simplified version by Sargent [6, 7] as depicted in Figure 1.1

The problem entity is the system or real-world problem phenomenon to be modelled. In a first step the problem must be captured in a conceptual model expressed by means of mathematical concepts, or verbal definitions resulting from an analysis and modelling phase. This is a task for the domain expert as he is familiar with the inner workings and specifics of the problem. The conceptual model is then translated into a computerised model which can be used for inference of the problem entity in the experimentation phase. For both these translation steps adequate validation and verification steps are required to assure that all assumptions and abstractions are reasonable, and that no mistakes are introduced by implementation. As a final step, the computer model must also be validated on the problem itself by assessing the

[1] http://www.wintersim.org

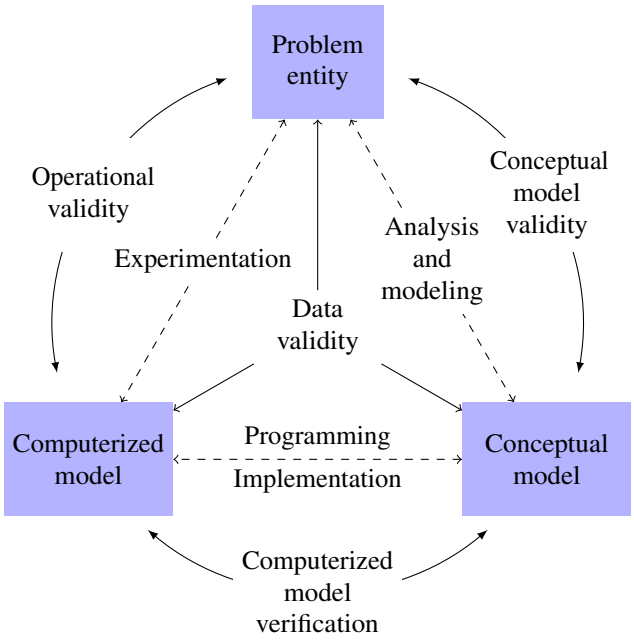


Figure 1.1: The computer modelling process.

accuracy of the model over the problem domain. Finally, the data validity assures that throughout the process the data integrity is maintained and remains valid for model building, evaluation, testing, and performing experiments.

1.3.2 Computer models for engineering

Amongst the countless research domains and fields of research revolutionised by the merits of computer simulation, the field of engineering is a prime example as it continues to benefit greatly from the introduction of computer simulations. During product design, engineers are confronted with several complex systems which need to be designed and/or optimized. These systems are parametrised by several input parameters (or factors or variables) and emit a set of outputs (responses). Together, these input parameters form the *input space* (or design space), whereas outputs form the *output space*. Traditionally, several prototypes with varying settings of the input parameters were required to observe and learn the relationship with the output(s). This in order to assure quality criteria were met, to obtain optimal solutions for design choices and/or to evaluate the behaviour of products and components under varying conditions. Each of these prototypes can be regarded as a data point in the joint input/output space. Often a single prototype

was insufficient, and lessons learnt were used to improve the design. Therefore, the development process used to involve building several prototypes in order to gain more confidence in the solutions and understand the mapping from input to output space. A direct consequence of this approach is that the development process is both slow and not cost effective.

The introduction of computer simulations resulted in a speed up of development processes at a lower cost. By bundling implementations of material, mechanical and, physical properties into a software package which simulates the desired aspects of a system and performs the tests and experiments virtually, the number of required prototypes can be drastically reduced to only a few at the very end of the design process. These prototypes can be regarded purely as a validation of the simulations as part of the process outlined in Section 1.3.1. The simulation itself can be interpreted as a model and serves as an abstract layer between the engineer and the real world phenomena. Performing a virtual experiment is faster and less expensive. A direct consequence was an acceleration of development and system design, contributing to a shorter time-to-market and a more effective process in general. In addition, it was also possible to perform more virtual experiments providing a way to achieve better products and design optimality.

1.4 Limitations and challenges

Although the introduction of computer models and simulations has revolutionised science and increased the possibilities for abstraction and reasoning on a much wider scale, some practical limitations apply as computers themselves are finite-state machines. These are summarized within in this section.

1.4.1 Computational cost

As simulation software became more precise and gained accuracy over the years, its computational cost grew tremendously. In fact, the growth of computational cost was so fast it has beaten the growth in computational power resulting in very lengthy simulations on state-of-the-art machines and high performance computing environments, mainly due to the never ending drive for finer time scales, more detail and general algorithmic complexity. For instance, a Computational Fluid Dynamics (CFD) simulation of a cooling system can take several days to complete [8], or a simulation of a single crash test was reported to take up to 36 hours to complete [9]. This introduces a new problem: large scale parameter sweeping and direct use of this type of computationally expensive simulations for evaluation

intensive tasks such as optimisation and sensitivity analysis is impractical and should be avoided.

1.4.2 Input dimensionality

One of the hardest problems of modelling in general is the explosive growth of the input space, in terms of the number of input parameters. This is referred to as the input dimensionality, moreover the problematic growth of the input space is known as the *curse of dimensionality*. Increasing the number of input parameters will spread the available data exponentially hence it quickly becomes intractable to cover the space non-sparsely, even more in combination with the computational cost of each evaluation within the scope of computer modelling. Side-effects of this phenomenon are that every sample turns into an outlier, distance measures (can) become non-informative (see Section 4.6.1), all points are near a boundary of the space as the centre of the space hardly exists any more (more coordinates yields a higher probability that at least one of them is near a boundary), and a lot of intuitive geometrical properties no longer hold (related to the mathematical field of topology).

As a small example of the counter intuitive effects: consider two input parameters with a continuous domain defined as the interval $[-1, 1]$. We define the centre of both intervals as $[-0.25, 0.25]$. The percentage of the area defined as centre with respect to the total area of the input space is given by $0.5^2/2^2 = 6.250\%$. Suppose instead of two input parameters we have eight: the size of the centre has now shrunk to $0.5^8/2^8 \approx 0.002\%$ of the size of the entire input space!

1.4.3 Interoperability

Large-scale simulations are typically achieved by linking several simulation codes and models of smaller systems. For instance when modelling the Earth, it makes sense to integrate high-fidelity models of the atmosphere, the oceanic system, geological, models of the internal dynamics as well as models describing the influence of external elements such as the Sun. It should be clear that this is challenging from a computational point of view, but also introduces the very complex task for software engineers to have all software interoperate correctly.

1.4.4 Proprietary systems and legacy code

A last practical limitation of computer models is that the code may be proprietary and thus prohibitively expensive and inaccessible to scrutiny or modification. Al-

ternatively, the simulation code may be legacy or the platform it runs on may be unmaintained making it arcane to set up and use.

1.5 Approximation

To counter the aforementioned limitations of computer models, an additional layer of abstraction between the complex system in the real world and the engineer was proposed, more specifically between the computer model (simulation) and the engineer. Rather than interacting directly, an approximation is constructed aiming to provide the following advantages:

- The approximation helps to avoid computationally expensive simulations
- Faster integration of (legacy) simulation codes
- Provide a generic and multi-platform solution
- Enable exploration and visualisation to provide insight
- Cloud-based deployment of models

To achieve these advantages, the approximations should both be simple and cheap-to-evaluate. Roughly three approaches to obtain this approximation exist, as briefly discussed in the following sections.

1.5.1 Model-driven approximation

The model-driven approximation approaches are often referred to as Model Order Reduction (MOR), physical models, or system-theory based metamodelling. These approaches take a top-down approach starting from the original equations implemented by the simulator. By applying mathematical and algebraic methods as well as projections, a lower order approximation is derived which can be evaluated more efficiently. This is a very elegant approach for systems where the internal state and dynamics can be expressed by means of (differential) equations such as in the field of electronics [10], transportation models, or CFD. As downside, model-driven approaches are usually specifically tailored for one application as the reductions, approximations, and derivations are designed for one system.

1.5.2 Data-driven approximation

At the other end of the approximation spectrum, we find the data-driven methods which take a bottom-up approach. The original system and/or simulator are

considered a black-box, with zero knowledge available about their inner working. The input-output behaviour is observed by means of evaluating a selection of inputs. Based on these observations, an approximation model is constructed, using regression or classification techniques such as

- Artificial Neural Network
- Support Vector Machine
- Least-Squares Support Vector Machine
- Gaussian Process
- Radial Basis Functions
- Random Forests
- Kriging
- Polynomial/rational models
- Splines

Due to the expensive nature of the data, the approximation methodologies can be non-parametric as not many training data is available. Techniques for pattern recognition and mining of larger datasets are difficult to apply [11, 12].

Because this data-driven methodology is very generic and disregards anything but the input/output behaviour it can be applied widely to virtually any problem. As a direct consequence these approximations, however, lack traceability.

1.5.3 Hybrid approximation methods

In real approximation processes, the situation is not black and white and more complicated than a choice between top-down or bottom-up. Often, the approximation methodology is somewhat in the middle in a large grey zone of combining the best of both approaches. The approximation could be data-driven, but the choice of evaluations and the modelling are affected by *prior* knowledge about the system such as rules and constraints. It is also possible to complement a model-driven approach with simulation data for further approximation or validation [13].

It is clear that hybrid approaches fill in the traceability needs of data-driven approaches by explicitly formulating expert knowledge. However, in the process the application domain of the hybrid method is restricted.

1.6 Scope: surrogate modelling

The different types of approximation each have their use and successful applications. The scope of this thesis is restricted to automated methods for construction of (mostly) data-driven approximations for (simulations of) complex systems. These types of data-driven approximations are typically referred to as surrogate models, metamodels, replacement models, emulators or Response Surface Models (RSMs). Hereafter, the simulation of the complex system being approximated is referred to as the *reference model* to maintain generality: simulators are a prime example of an expensive evaluation target, nevertheless this term can be interpreted broadly. Training a model with supplied hyperparameters, performing real-life experiments, or capturing data from an operating complex system may also be considered simulations. These approaches however result in some output uncertainty. To further restrict the scope of this dissertation we make following assumptions:

1. The reference model has **static** input/output behaviour, which does not evolve over time. This excludes time series and prediction of future states.
2. For most of the contributions of this dissertation, the output is produced by computer simulations which are **deterministic** and quasi noise-free (with the exception of some numerical and discretisation noise). Occasionally, the role of output uncertainty (noise) is further investigated. One major exception in this dissertation can be found in Appendix B, in which data is observed directly from a running complex system (the real world) without a computer model in-between.
3. A single evaluation of the reference model is **expensive** to obtain, and limited information on the inner workings is available (black-box).

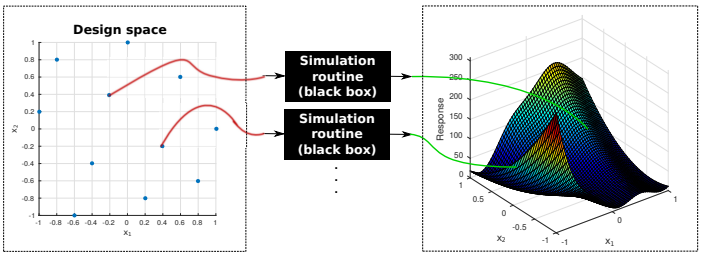


Figure 1.2: Surrogate modelling: an experimental design with the design space is evaluated by the reference model. Its responses are used to obtain a data-driven approximation of the input/output behaviour.

The workflow to develop a surrogate model typically consists of the following steps:

1. *Formulate model*: this involves a better understanding of the system by inquiring from domain experts, and mapping their knowledge into bounded input/output parameters. This results in the formulation of the design space, and any prior knowledge which may influence other choices regarding the experimental design, model fitting or quality assessment.

2. *Experimental design*: select the type of experimental design to use and, if chosen, how sequential design is to be applied. This results in a set of *samples* to be evaluated by the reference model. Selection of a set of samples is discussed further in Section 2.5 and Chapter 3.

3. *Model fitting*: define the approach for model fitting. A specific model type may be chosen, or this may be done automatically. Also the choice of the model parameters and their optimisation is a relevant question for this step. A formal approach is given in Section 2.2.

4. *Model quality assessment*: define how the model quality will be quantified and validated. This is one of the hardest questions to answer as it involves several stakeholders. Sections 2.2 and 2.3 provide a detailed overview.

5. *Gain insight*: Applying the trained model for its task results in insight into the system.

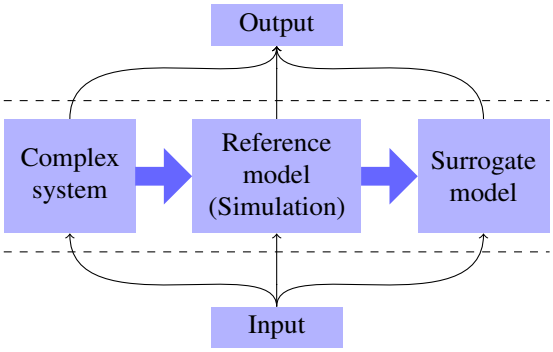


Figure 1.3: Surrogate modelling: modelling hierarchy.

The surrogate model mimics the response behaviour of the reference model. Typical tasks or use cases involving use of surrogates include:

- Optimisation
- Perform exploration over a large design space to focus further development on some specific areas.
- Perform sensitivity analysis

- Visualisation of the design space, trade-offs, feasibility, optimality etc.
- Replacement of legacy reference models

All of these tasks can be achieved by training a globally accurate model over the entire design space. The surrogate model then replaces the underlying system or reference model for evaluations. However, for some of these use cases, more specific and more efficient (in terms of number of required evaluations of the reference model) approaches exist.

1.6.1 Surrogate-based optimisation

Surrogates excel in the optimisation of expensive objective functions [14]. This discipline is often referred to as Surrogate-Based Optimisation (SBO). A globally accurate surrogate model can be built and optimised using traditional optimisation methods such as gradient descent, or meta-heuristics such as particle swarm optimisation [15]. Although this approach is correct and works faster than evaluating each call of the objective function on the reference model, it is not necessarily the most efficient methodology. When seeking a minimum, less samples can be devoted to regions that clearly show to be the opposite. Applying sequential sampling to explore the search space for optima and exploit the available knowledge to refine optima results in a more optimal process.

1.6.2 Sensitivity analysis

A different use-case of surrogate models is sensitivity analysis of the complex system. Especially when many input parameters are present, it is very difficult to achieve global accuracy due to the exponential growth of the input space. Fortunately not all input parameters contribute equally to the output variability, in fact some might not have any impact at all [16]. The surrogate models can be used directly for evaluation-based sensitivity analysis methods such as Sobol Indices [17], Interaction Indices [18] or gradient-based methods.

For some kernel-based modelling methods analytical computation of sensitivity measures is possible resulting in faster and more reliable estimation schemes, even before global accuracy is achieved [19]. This intuitively makes sense as model belief on the input sensitivity may be interpreted as a necessary precondition to high accuracy.

1.6.3 Inverse surrogate modelling

All tasks described so far were forward tasks, mapping samples from a design space to the output or objective space. It is also possible to reverse the method, referred to as inverse surrogate modelling, which can be interpreted as identifying the areas of the design space corresponding to a certain desired or feasible output range.

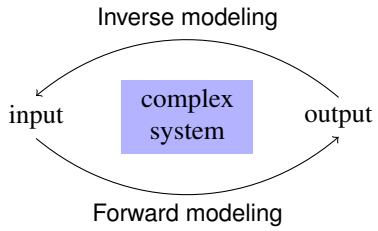


Figure 1.4: Forward versus inverse surrogate modelling

Typical approaches involve training a (forward) surrogate model first, then optimising the model using an error function between the output and the desired output as objective function. This optimisation is typically preferred to be robust to account for the error of the forward surrogate model [20]. Next, specific sampling schemes to identify these regions directly were also proposed [21]. Finally, it is possible to translate the inverse problem into a forward problem involving discretising the output (feasible/infeasible point) and learning the class boundaries, using the approaches described in Chapter 3.

1.7 Research goals and outline

This research aims to provide novel methodologies within the surrogate modelling workflow for approximating black-box deterministic systems or simulations. Strong focus is on the adaptive sampling methodologies in order to sequentially gain insight into the black-box and extract knowledge from it in order to achieve our final, predefined goal. Improvements are interpreted broadly, aiming both at reducing the number of required evaluations of the reference model, the computational complexity itself as well as expanding the capabilities of the surrogate modelling workflow. In summary:

- The surrogate modelling concept with sequential design is extended to discrete outputs, using classifier models and specific sampling strategies,
- The computational complexity of a successful sequential sampling method is significantly reduced. This enables application of the method for problems

with higher dimensionality

- A novel formulation for a sequential sampling criterion for optimization is derived, assuming deterministic responses. The new formulation allows faster computation and is interpretable.

In Chapter 2 the concept of surrogate modelling is introduced in depth with a detailed description of all technical aspects as well as potential conflicts with requirements as specified by the user and introduces all key concepts used in later chapters. Next, the chapter also positions the surrogate modelling research area as multi-disciplinary and discusses the relationship with other fields, in particular the field of machine learning.

Next, the extension of the surrogate modelling framework to reference models with discrete outputs are outlined in Chapter 3. This situation arises frequently in the context of constraint satisfaction problems or optimisation with black-box constraints. In addition this chapter provides an overview of existing sequential sampling methodologies with a focus on sequential sampling under discrete output settings.

This is followed by discussing two improvements to existing sequential sampling methodologies in Chapters 4 and 5. First, an existing algorithm for global surrogate modelling, known as Local Linear Approximation (LOLA)-Voronoi, is improved for input spaces with higher dimensionality by reducing its computational complexity significantly without decreasing its performance. This algorithm has already proven its efficiency previously in several studies [22–27]. Then, a new formulation for computation of an acquisition function for SBO, known as the Knowledge-Gradient for Continuous Parameters (KGCP) is derived, based on the assumption that data is deterministic. This formulation allows for easier interpretation of the score, as well as a very intuitive comparison with Expected Improvement (EI), the state-of-the-art method for SBO. This algorithm is then benchmarked under different settings, and compared to a number of common methods for SBO.

After the global conclusions and an overview of interesting follow-up research in Chapter 6, Appendix A studies the applicability of a recent non-parametric prior over functions for multi-objective optimisation applications. This prior allows for additional flexibility in comparison to GP and can be of interest when response surfaces are difficult to approximate. Finally, Appendix B shows the applicability of surrogate modelling methodologies in a real-world application. Here, there is no more reference model but the data-driven approximation is applied directly to data of an operating complex system (a coldstore factory) to assess the available power flexibility: a relevant challenging open issue in smart-grid management.

1.8 Publications

1.8.1 International journals

- **J. van der Herten**, I. Couckuyt, D. Deschrijver, and T. Dhaene. "Fast Calculation of the Knowledge Gradient for Optimization of Deterministic Engineering Simulations". Submitted to *Journal of Machine Learning Research (JMLR)* (2017).

- T. Van Steenkiste, **J. van der Herten**, I. Couckuyt, and T. Dhaene. "A Complete Sensitivity Analysis Algorithm for the Analysis of Expensive Black-Box Simulators". Submitted to *European Journal of Operational Research (EJOR)* (2017).

- **J. van der Herten**, I. Couckuyt, D. Deschrijver, and T. Dhaene. "Adaptive classification under computational budget constraints using sequential data gathering". *Advances in Engineering Software* 99 (2016), pp. 137–146.

- J. Ruyssinck, **J. van der Herten**, R. Houthooft, F. Ongenae, I. Couckuyt, B. Gadeyne, K. Colpaert, J. Decruyenaere, F. De Turck, and T. Dhaene. "Random Survival Forests for Predicting the Bed Occupancy in the Intensive Care Unit". *Computational and Mathematical Methods in Medicine* 2016 (2016).

- P. Singh, **J. van der Herten**, D. Deschrijver, I. Couckuyt, and T. Dhaene. "A sequential sampling strategy for adaptive classification of computationally expensive data". *Structural and Multidisciplinary Optimization* (2016), pp. 1–14.

- **J. van der Herten**, I. Couckuyt, D. Deschrijver, and T. Dhaene. "A Fuzzy Hybrid Sequential Design Strategy for Global Surrogate Modeling of High-Dimensional Computer Experiments". *SIAM Journal on Scientific Computing* 37.2 (2015), A1020–A1039.

- R. Houthooft, J. Ruyssinck, **J. van der Herten**, S. Stijven, I. Couckuyt, B. Gadeyne, F. Ongenae, K. Colpaert, J. Decruyenaere, T. Dhaene, F. De Turck. "Predictive modelling of survival and length of stay in critically ill patients using sequential organ failure scores". *Artificial intelligence in medicine* 63.3 (2015), pp. 191–207.

1.8.2 Conferences

- N. Knudde, **J. van der Herten**, I. Couckuyt, and T. Dhaene. "How to Avoid Unwelcome Surprises, Analytically - A Bayesian Approach to Robust Optimisation". *Submission under review for inclusion in the proceedings of the 31st Neural Information Processing Systems Conference.* Long Beach, CA, USA: Curran Associates, Inc., 2017.
- V. Dutordoir, N. Knudde, **J. van der Herten**, I. Couckuyt, and T. Dhaene. "Deep Gaussian Process Metamodeling of Sequentially Sampled Non-stationary Response Surfaces". *Submission under review for inclusion in the proceedings of the 50th Conference on Winter Simulation*. Ed. by W. K. V. Chan, A. D'Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer, and E. Page. Las Vegas, NV, USA: Institute of Electrical and Electronics Engineers, Inc, 2017.
- **J. van der Herten**, V. Dutordoir, I. Couckuyt and T. Dhaene. "Surrogate Modeling with Sequential Design for Design and Analysis of Electronic Systems". *Accepted for inclusion in the proceedings of the International Conference on Electromagnetics in Advanced Applications (ICEAA)*. Verona Italy: Institute of Electrical and Electronics Engineers, Inc, 2017.
- **J. van der Herten**, I. Couckuyt, D. Deschrijver, P. Demeester, and T. Dhaene. "Adaptive modeling and sampling methodologies for Internet of Things applications". *18th Mediterranean Electrotechnical Conference (MELECON)*. IEEE. Limassol, Cyprus: Institute of Electrical and Electronics Engineers, Inc, 2016, pp. 1–5.
- **J. van der Herten**, I. Couckuyt, D. Deschrijver, and T. Dhaene. "Active Learning for Approximation of Expensive Functions with Normal Distributed Output Uncertainty". *DEML 2016 Workshop at the the 33rd International Conference on Machine Learning (ICML-16)*. Manhattan, New York, 2016.
- **J. van der Herten**, I. Couckuyt, D. Deschrijver, and T. Dhaene. "Multi-objective variable subset selection using heterogeneous surrogate modeling and sequential design". *Congress on Evolutionary Computation (CEC)*. IEEE. Vancouver, BC, Canada: Institute of Electrical and Electronics Engineers, Inc, 2016, pp. 1634–1641.
- **J. van der Herten**, I. Couckuyt, and T. Dhaene. "Hypervolume-based Multi-objective Bayesian Optimization with Student-t Processes". *BayesOpt 2016 Workshop at the 30th Conference on Neural Information Processing Systems.* Barcelona, Spain, 2016.

- **J. van der Herten**, F. Depuydt, L. De Baets, D. Deschrijver, M. Strobbe, C. Develder, T. Dhaene, R. Bruneliere, and J.-W. Rombouts. "Energy Flexibility Assessment of an Industrial Coldstore Process". *International Energy Conference (ENERGYCON)*. Leuven, Belgium: Institute of Electrical and Electronics Engineers, Inc, 2016, pp. 1–6.
- **J. van der Herten** and T. Dhaene. "Feature and Model Type Selection using Multi-Objective Optimization for AutoML". *BENELEARN*. Kortrijk, Belgium, 2016.
- T. Van Steenkiste, **J. van der Herten**, I. Couckuyt, and T. Dhaene. "Sensitivity Analysis of Expensive Black-box Systems using Metamodeling". *Proceedings of the 49th Conference on Winter Simulation*. Ed. by T. M. K. Roeder, P. I. Frazier, R. Szechtman, E. Zhou, T. Huschka, and S. E. Chick. Washington, D.C., USA: Institute of Electrical and Electronics Engineers, Inc, 2016.
- **J. van der Herten**, I. Couckuyt, D. Deschrijver, and T. Dhaene. "Constructing classifiers of expensive simulation-based data by sequential experimental design". *Proceedings of the 48th Conference on Winter Simulation*. Ed. by L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti. Huntington Beach, CA, USA: Institute of Electrical and Electronics Engineers, Inc, 2015, pp. 3166–3167.
- **J. van der Herten**, D. Deschrijver, and T. Dhaene. "Fuzzy local linear approximation-based sequential design". *Computational Intelligence for Engineering Solutions (CIES)*. IEEE. Orlando, FL, USA: Institute of Electrical and Electronics Engineers, Inc, 2014, pp. 17–21.

1.8.3 Book chapters

- **J. van der Herten**, T. Van Steenkiste, I. Couckuyt, and T. Dhaene. "Surrogate Modelling with Sequential Design for Expensive Simulation Applications" Accepted for inclusion in *Computer Simulation*. Ed. by D. Cvetković InTech, 2017. ISBN : 978-953-51-5217-0.

1.8.4 Abstracts & posters

- **J. van der Herten** and T. Dhaene. "Fast hybrid sequential design for high-dimensional surrogate modeling". BENELEARN Abstract. 2015.
- **J. van der Herten** and T. Dhaene. "Fuzzy Sequential Space-filling Design". BENELEARN Abstract. 2014.

1.9 References

[1] J. Oden, T. Belytschko, T. Hughes, C. Johnson, D. Keyes, A. Laub, L. Petzold, D. Srolovitz, and S. Yip. *Revolutionizing engineering science through simulation*. Tech. rep. Arlington, VA, USA: National Science Foundation (NSF): Blue Ribbon Panel on Simulation-Based Engineering Science, 2006.

[2] H. Schichl. "Models and the History of Modeling". *Models and the History of Modeling*. Ed. by J. Kallrath. Vol. 88. Applied Optimization. Springer US, 2004. Chap. 2, pp. 25–36. ISBN: 978-1-4613-7945-4.

[3] D. Hira and P. Gupta. *Operations Research*. 5th ed. S. Chand & Company, 2014. ISBN: 9788121902816.

[4] G. Gordon. "The Development of the General Purpose Simulation System (GPSS)". *SIGPLAN Not.* 13.8 (Aug. 1978), pp. 183–198. ISSN: 0362-1340.

[5] J. P. Kleijnen. "Verification and validation of simulation models". *European journal of operational research* 82.1 (1995), pp. 145–162.

[6] R. G. Sargent. "Verification and validation of simulation models". *Proceedings of the 38th Conference on Winter Simulation*. Ed. by M. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines. Orlando, FL, USA: Institute of Electrical and Electronics Engineers, Inc, Dec. 2005, pp. 130–143.

[7] R. G. Sargent. "Verification and validation of simulation models". *Journal of simulation* 7.1 (2013), pp. 12–24.

[8] K. Goethals, I. Couckuyt, T. Dhaene, and A. Janssens. "Sensitivity of night cooling performance to room/system design: Surrogate models based on CFD". *Building and Environment* 58 (2012), pp. 23–36.

[9] T. Simpson, J. D. Poplinski, P. N. Koch, and J. K. Allen. "Metamodels for Computer-based Engineering Design: Survey and recommendations." *Eng. Comput. (Lond.)* 17.2 (2001), pp. 129–150.

[10] E. R. Samuel. "Parameterized modeling and model order reduction for large electrical systems". eng. PhD thesis. Ghent University, 2015, pp. XXV, 179. ISBN: 9789085787747.

[11] T. Calders and B. Goethals. "Mining all non-derivable frequent itemsets". *Principles of Data Mining and Knowledge Discovery*. Springer Berlin Heidelberg. 2002, pp. 74–86.

[12] S. Moens, E. Aksehirli, and B. Goethals. “Frequent itemset mining for big data”. *International Conference on Big Data*. IEEE. Santa Clara, CA, USA: Institute of Electrical and Electronics Engineers, Inc, 2013, pp. 111–118.

[13] K. Chemmangat Manakkal Cheriya. “Sequential sampling strategies for automated parametric macromodeling of high-frequency electrical systems”. eng. PhD thesis. Ghent University, 2014, pp. XXIII, 182. ISBN: 9789085786856.

[14] F. A. Viana, T. W. Simpson, V. Balabanov, and V. Toropov. “Metamodeling in Multidisciplinary Design Optimization: How Far Have We Really Come?” *AIAA Journal* 52.4 (2014), pp. 670–690.

[15] J. Kennedy and R. Eberhart. “Particle swarm optimization”. *Proceedings of the IEEE International Conference on Neural Networks*. IEEE. Perth, WA, Australia: Institute of Electrical and Electronics Engineers, Inc, 1995.

[16] A. Saltelli. “Sensitivity analysis for importance assessment”. *Risk Analysis* 22.3 (2002), pp. 579–590.

[17] I. M. Sobol. “Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates”. *Mathematics and computers in simulation* 55.1 (2001), pp. 271–280.

[18] K. Ito, I. Couckuyt, S. Poles, and T. Dhaene. “Variance-based interaction index measuring heteroscedasticity”. *Computer Physics Communications* 203 (2016), pp. 152–161.

[19] T. Van Steenkiste, J. van der Herten, I. Couckuyt, and T. Dhaene. “Sensitivity Analysis of Expensive Black-box Systems using Metamodeling”. *Proceedings of the 49th Conference on Winter Simulation*. Ed. by T. M. K. Roeder, P. I. Frazier, R. Szechtman, E. Zhou, T. Huschka, and S. E. Chick. Washington, DC, USA: Institute of Electrical and Electronics Engineers, Inc, 2016.

[20] G. Dellino, J. P. Kleijnen, and C. Meloni. “Robust simulation-optimization using metamodels”. *Proceedings of the 42th Conference on Winter Simulation*. Ed. by M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls. Austin, TX, USA: Institute of Electrical and Electronics Engineers, Inc, 2009, pp. 540–550.

[21] I. Couckuyt, J. Aernouts, D. Deschrijver, F. D. Turck, and T. Dhaene. “Identification of quasi-optimal regions in the design space using surrogate modeling”. *Engineering with Computers* 29.2 (2013), pp. 127–138.

[22] J. Aernouts, I. Couckuyt, K. Crombecq, and J. J. Dirckx. "Elastic characterization of membranes with a complex shape using point indentation measurements and inverse modelling". *International Journal of Engineering Science* 48.6 (2010), pp. 599–611.

[23] D. Stephens, D. Gorissen, K. Crombecq, and T. Dhaene. "Surrogate based sensitivity analysis of process equipment". *Applied Mathematical Modelling* 35.4 (2011), pp. 1676–1687.

[24] S. Koziel and L. Leifsson. *Surrogate-Based Modeling and Optimization: Applications in Engineering*. SpringerLink : Bücher. Springer New York, 2013. ISBN: 9781461475514.

[25] D. Deschrijver, K. Crombecq, H. M. Nguyen, and T. Dhaene. "Adaptive Sampling Algorithm for Macromodeling of Parameterized-Parameter Responses". *Microwave Theory and Techniques, IEEE Transactions on* 59.1 (2011), pp. 39–45.

[26] D. Deschrijver, F. Vanhee, D. Pissoort, and T. Dhaene. "Automated near-field scanning algorithm for the EMC analysis of electronic devices". *Electromagnetic Compatibility, IEEE Transactions on* 54.3 (2012), pp. 502–510.

[27] S. Aerts, D. Deschrijver, W. Joseph, L. Verloock, F. Goeminne, L. Martens, and T. Dhaene. "Exposure assessment of mobile phone base station radiation in an outdoor environment using sequential surrogate modeling". *Bioelectromagnetics* (2013).

"There are three types of lies: lies, damned lies and big data"

— Neil Lawrence

2

Data-Efficient Machine Learning for Surrogate Modelling

2.1 Introduction

The background and motivation for surrogate models were outlined in the last chapter, and is now followed by a more technical overview of the difficulties and trade-offs which come into play during their construction. Starting from the formal definition by Gorissen [1], different aspects ranging from modelling to the selection of training data are further discussed.

2.2 Adaptive modelling formalism

The process of constructing a surrogate model can be mathematically expressed as follows: given an unknown function $f : \mathcal{X} \rightarrow \mathbb{C}^q$ defined over the input domain $\mathcal{X} \subset \mathbb{R}^d$, whose function values $F = \{ f(\boldsymbol{x}_0), ..., f(\boldsymbol{x}_{n-1}) \}$ are known for a set of n distinct query points $X = \{\boldsymbol{x}_0, ..., \boldsymbol{x}_{n-1}\}$. These observations

are exact due to the deterministic assumption[1]. Together, they form the data set $D = \{ (\boldsymbol{x}_0, f(\boldsymbol{x}_0)), ...(\boldsymbol{x}_{n-1}, f(\boldsymbol{x}_{n-1})) \} \subset \mathcal{X} \times \mathbb{C}^q$. A suitable function $\tilde{f}$ from an approximation space S with $\tilde{f} : \mathcal{X} \to \mathbb{C}^q \in S$ has to be chosen. The form of the approximations and the approximation space is discussed in more detail in Section 2.3.

The choice of approximation is based on a criterion ξ which introduces an order on S, and is a combination of 3 different aspects:

$$\xi = (\Lambda, \kappa, \tau) .$$

Given $\kappa \in E$, an error function from the set of error functions which measures the similarity between a vector of predictions $\tilde{f}(\boldsymbol{x}_i)$ and the observed responses $f(\boldsymbol{x}_i)$. An overview of typical error functions is given in Section 2.6. The value τ is the target error for κ and is specified by the user. Finally, Λ is a quality estimator with $\Lambda : E \times S \times \mathcal{P}(\mathcal{X} \times \mathbb{C}^q) \to \mathbb{R}^+$ (usually higher quality means higher values for Λ). The adaptive modelling can now be expressed as:

$$\arg\max_{t \in T} \arg\max_{\theta \in \Theta} \quad \Lambda(\kappa, \tilde{f}_{t,\theta}, D). \tag{2.1}$$

Here we denote $\tilde{f}_{t,\theta}$ as approximation (or surrogate model) of type t from the allowed set of model types T with hyperparameters $\boldsymbol{\theta} \in \Theta$, the associated hyperparameter space. We assume all model parameters are optimally determined by a model training procedure as is discussed in Section 2.3. When the optimal approximation $\tilde{f}^\star_{t,\theta}$ as defined by Equation (2.1) is identified and satisfies $\Lambda(\kappa, \tilde{f}^\star_{t,\theta}, D) \geq \tau$, it means an approximation was found which satisfies the predefined modelling goals.

The outer maximisation handles the **model type selection** aspect of the surrogate modelling process: this step handles the selection of the type of approximation. Some common approximation types were listed in Section 1.5.2. In practise, often a practitioner limits this choice to a single model type, although heterogeneous approaches and automated selection procedures have been proposed [2, 3]. The second optimisation problem tunes the **hyperparameters** of a specific model type t to find an optimum of the model quality estimator Λ. An example of a hyperparameter optimisation is tuning the kernel parameter and the regularisation constant of a Support Vector Machine (SVM), identifying the optimal order of a polynomial or determining the most appropriate architecture for an Artificial Neural Network (ANN).

The role of the quality estimator is crucial to guide the search over the model type t and hyperparameters $\boldsymbol{\theta}$ to obtain an *optimal* approximation function $\tilde{f}^\star \in S$. It must

[1] in case f can not be observed directly, a different set $Y = \{ \boldsymbol{y}_0, ..., \boldsymbol{y}_{n-1} \}$ with $\boldsymbol{y}_i = f(\boldsymbol{x}_i) + \epsilon_i$ and $\epsilon_i \sim \mathcal{N}(0, \sigma_n^2)$ is obtained instead.

encode the user requirements for the final model in terms of accuracy, smoothness, occurrence of poles or correspondence to physical behaviour. These requirements must be matched with some technical considerations to assure the search over the model hyperparameters $\boldsymbol{\theta}$ results in a model with good *generalisation performance* as discussed in more detail in Section 2.3. Specifying an appropriate quality estimator is often overlooked, requiring several iterations of the process to obtain satisfactory results. Consulting the users of the surrogate model and defining what their expectations from the model are, is a good starting point. These requirements can then be formally translated into a good quality estimator. Unfortunately, defining Λ does not end with casting user requirements. Some important aspects of machine learning must also be handled carefully: this is where the role of the data scientist comes in. Specific expertise is required to join both objectives.

For instance, a straightforward approach is minimizing the error between the surrogate model response and the true responses Y for the training input data X. This is often referred to as *training error* or *sample error* and pushes the hyperparameter optimisation to favour models interpolating the data points perfectly. Although this solution might be considered satisfactory at first sight, in reality this rarely provides a good model as the optimisation problems do not consider model quality in $\mathcal{X} \setminus X$ at all. This approach inevitably leads to very unreliable responses when points $\boldsymbol{x}_\star \notin \mathcal{X}$ are to be predicted. These technical considerations are now discussed in more detail.

2.3 Empirical risk minimisation

To further explore the issue of model quality from a machine learning point of view and to explain the pitfalls of choosing Λ, it is important to explore empirical risk minimisation and the bias-variance trade-off. These concepts are crucial to understand how the training of models works, and are required knowledge for data scientists to avoid poor modelling results. This section provides a brief theoretical analysis of the bias-variance trade-off and motivates the need for more complex strategies for model quality estimation (Λ), rather than sample error. An excellent in-depth overview of these statistical learning aspects can be found in Vapnik [4] and Hastie et al. [5].

Starting point is Ξ, an infinite set of non-linear basis functions of $\boldsymbol{x}$. We can now formally define the approximation space as $S = \text{span}(\Xi)$, hence all approximations take the form:

$$\tilde{f}(\boldsymbol{x}) = \sum_{i=0}^{T} w_i \varphi_i(\boldsymbol{x}). \tag{2.2}$$

Identifying approximations which satisfy predefined model requirements from the infinite amount of possible approximations is challenging and consists of several steps. First of all the approximation form is characterised by model parameters $\boldsymbol{w}$ which include the coefficients of the linear combination w_i, but can also include parameters of the basis functions[2]. These model parameters are learnt from D by a training procedure, based on Empirical Risk Minimization (ERM)[3].

Definition 1 (Empirical Risk Minimization (with Tikhonov regularisation))**.** *Given the unknown mapping $f : \mathcal{X} \to \mathbb{C}^q$ and a candidate approximation $\tilde{f}$. Defining $\Omega = \mathcal{X} \times \mathbb{C}^q$, we may assume a joint probability space $(\Omega, \mathcal{F}, p_\Omega)$ with $\mathcal{F}$ the σ-algebra of Borel sets on Ω. It follows that the training data $D \in \mathcal{F}$. The risk associated to this approximation given model parameters $\boldsymbol{w}$ can be expressed as:*

$$\int_\Omega \ell\left(\tilde{f}\left(\boldsymbol{x}\right), \boldsymbol{y} \;\middle|\; \boldsymbol{w}\right) \mathrm{d}p_\Omega(\boldsymbol{x}, \boldsymbol{y}) \,.$$

In general this integral can not be computed as the probability measure of the probability space is unknown. Instead we compute an approximation known as the empirical risk, and add a regulariser with associated regularisation constant λ as motivated in Section 2.3.2. The model can now be trained, i.e., its parameters $\boldsymbol{w}$ can be determined by optimizing a non-negative real-valued loss function ℓ as follows:

$$\arg\min_{\boldsymbol{w}} \sum_{i=0}^{n-1} \underbrace{\ell\left(\tilde{f}\left(\boldsymbol{x}_i\right), \boldsymbol{y}_i \;\middle|\; \boldsymbol{w}\right)}_{\textit{loss}} + \underbrace{\lambda\chi(\boldsymbol{w})}_{\textit{regulariser}} \,.$$

Additionally it can be observed how approximations of the form outlined by Equation (2.2) are further specified by the number of terms, as well as the choice of basis functions φ. Typically, the choice of a specific model type constraints the form of the basis functions: for instance a SVM with Radial Basis Function (RBF) kernel the basis functions are products of the observations and kernels defined between $\boldsymbol{x}$ and the training points. A model type also defines the exact training procedure for the model parameters and includes specific settings such as the specification of the loss function.

A model type then typically has a set of parameters affecting its complexity and training. Roughly, these parameters can be categorised as follows:

- Parameters controlling the optimisation method to determine the model parameters (e.g, momentum parameter in certain optimisers for ANN).
- Parameters controlling the loss function and the regulariser.

[2] The basis functions φ are part of S themselves and may involve additional model parameters.

[3] Despite the terminology, this also holds for non-parametric methods. For instance kernel SVMs have a complexity which grows as more data points are added, yet requires coefficients to be learnt.

- Parameters which affect the complexity of the model: this can be interpreted as the number of terms in Equation (2.2) or parameters controlling the complexity of the basis functions.

These parameters are referred to as hyperparameters $\boldsymbol{\theta}$, the joint space of all these parameters is referred to as the hyperparameter space Θ and is specific for each model type. Their values must be determined prior to model training. An appropriate choice is crucial to obtain an appropriate approximation, as discussed throughout this section.

The search over S can now be summarized as a two-layered optimisation process: at the top, the model type and hyperparameters are optimised using a quality estimator as objective function. Each candidate $\tilde{f}_{t,\theta}$ then has a specified structure and complexity. Within the quality estimator the second optimisation of Definition 1 is performed on a subspace of S, as only model parameters are optimised using a loss function on the training data as objective. A good example of the described hierarchical model training is a neural network. The training method (i.e., back-propagation) optimises the weights to directly reduce the loss function (such as the Mean Square Error (MSE)) on the training data, whereas the hyperparameter optimisation involves the network architecture and training parameters such as regularisation and the learning rate to improve the Λ metric.

This approach raises some relevant questions: why is this two-layer approach required? Wouldn't it be sufficient to jointly optimise $\boldsymbol{w}$, $\boldsymbol{\theta}$ and t using only the loss function? Why is the regulariser required and why is a separate Λ needed? The short answer is that due to our limited view on f through the set of evaluations, focussing only on the performance in the training points (as in Definition 1) does not account for the *generalisation* of the approximation, i.e., the quality of the predictions of an approximation for unobserved points $\boldsymbol{x}_\star \notin \mathcal{X}$.

For the remainder of this section the model type is no longer explicitly mentioned and will be assumed to be pre-specified[4]. First a theoretical analysis of the distribution of the error of a given approximation $\tilde{f}$ over $\mathcal{X}$ is performed, assuming the hyperparameters are pre-specified. This analysis helps to understand the relation between the error distribution and the generalisation performance. In addition it also explains the role of regularisation and its impact on the error distribution as discussed in Section 2.3.2. Furthermore, it is also shown how regularisation can be achieved through data corruption. Finally, a description is given how to specify Λ to assess model quality in function of its hyperparameters (and model type) without compromising model generalisation which facilitates adaptive modeling as specified in Equation (2.1).

[4]It can also be regarded as a hyperparameter itself as in Gorissen et al. [2] and Couckuyt et al. [3]

2.3.1 Loss

The loss function ℓ usually corresponds to an error function such as the squared error evaluated on the prediction of $\tilde{f}$ on the training data. However, ultimately we want our model to predict any $\boldsymbol{x}_\star \in \mathcal{X}$ well (and have good generalisation performance). We proceed with a theoretical analysis of the loss by computing its expected value. This will result in a clear decomposition of the error in three interpretable terms which provide insight in the requirements for generalisation performance. For completeness the possibility of noisy observations is included in our theoretical analysis, hence $y = f(\boldsymbol{x}) + \epsilon,\ \epsilon \sim \mathcal{N}(0, \sigma_n^2)$. If σ_n^2 equals zero the normal distribution becomes a Dirac delta function and hence $y = f(\boldsymbol{x})$.

Assuming f were known, the expected value of the loss can be computed over the probability space $(\mathcal{X}, \mathcal{F}, p_\mathcal{X})$. Here $\mathcal{F}$ represents the σ-algebra of Borel sets on $\mathcal{X}$, and $p_\mathcal{X}$ the Lebesgue measure on $\mathcal{X}$[5]. We are interested in the decomposition of the expected error for a given $\tilde{f}$ with fixed complexity and training parameters so no notion of hyperparameters is included in this section. Furthermore, a given set of model parameters is assumed. The expected value of the loss can now be computed as:

$$\begin{aligned}
\mathbb{E}\left[\ell\left(\tilde{f}(\boldsymbol{x}), y \,\middle|\, \boldsymbol{w}\right)\right] &= \int_\mathcal{X} \ell\left(\tilde{f}(\boldsymbol{x}), y \,\middle|\, \boldsymbol{w}\right) \mathrm{d}p_\mathcal{X}(\boldsymbol{x}) \,, \\
&= \int \ell\left(\tilde{f}(\boldsymbol{x}), y \,\middle|\, \boldsymbol{w}\right) p(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \,, \\
&= \int \ell\left(\tilde{f}(\boldsymbol{x}), y \,\middle|\, \boldsymbol{w}\right) \mathrm{d}\boldsymbol{x} \,.
\end{aligned}$$

From the probability space, it follows the randomness of the loss originates from $\boldsymbol{x}$. This turns the loss into a measurable function of a random variable which justifies the second step. From the definition of the probability space it follows $\boldsymbol{x}$ is uniformly distributed over the compact set $\mathcal{X}$. Hence the expected value turns into the integral of the loss function. Defining following quantities (omitting the arguments $\boldsymbol{x}$ and the dependence on $\boldsymbol{w}$ for clarity):

$$\begin{aligned}
\text{Bias}\left[\tilde{f}\right] &= \mathbb{E}\left[\tilde{f} - f\right] \,, \\
\text{Var}\left[\tilde{f}\right] &= \mathbb{E}\left[\tilde{f}^2\right] - \mathbb{E}\left[\tilde{f}\right]^2 \,.
\end{aligned}$$

[5] This definition of the probability space assumes $\mathcal{X}$ is compact and consists of continuous parameters.

In case ℓ is defined as square loss the hypothetical expected model error can be decomposed as follows:

$$\begin{aligned}
\mathbb{E}\left[\left(y-\tilde{f}\right)^2\right] &= \mathbb{E}\left[\tilde{f}^2 - 2y\tilde{f} + y^2\right], \\
&= \mathbb{E}\left[\tilde{f}^2\right] - 2\mathbb{E}\left[y\tilde{f}\right] + \mathbb{E}\left[y^2\right], \\
&= \text{Var}\left[\tilde{f}\right] + \mathbb{E}\left[\tilde{f}\right]^2 - 2f\mathbb{E}\left[\tilde{f}\right] + \text{Var}\left[y\right] + \mathbb{E}\left[y\right]^2, \\
&= \left(f - \mathbb{E}\left[\tilde{f}\right]\right)^2 + \text{Var}\left[\tilde{f}\right] + \sigma_n^2, \\
&= \text{Bias}\left[\tilde{f}\right]^2 + \text{Var}\left[\tilde{f}\right] + \sigma_n^2.
\end{aligned}$$

The observation noise is a term of the expected error and is irrecoverable. However, the remainder of the error is composed of two components. The ideal $\tilde{f}$ (with respect to the MSE) minimises both:

- *Bias*: this part of the error is inherently caused by the choice of model type and its hyperparameters. For instance, first order linear regression assumes the data to be on a hyperplane regardless of the training data, even if the data was sampled from, e.g., a parabola. This is clearly the case in Figure 2.1a: the linear regression is unable to represent the wave accurately. High bias error typically indicates the model is not flexible enough.

- *Variance*: high variance indicates a model which is complex and memorises behaviour that does not generalise, such as the irrecoverable noise error. This implies that for a different set of training data sampled from the same f, the model changes significantly. An illustration of a model with high variance is given in Figure 2.1b. The polynomial of order 19 perfectly interpolates all data points, but the wild oscillations in between are unlikely to represent f.

As the dataset D only offers a limited (possibly noise corrupted) view on the true f it is usually impossible to reduce both of these quantities and a trade-off is involved. For instance, the simplest model available is a constant model predicting the mean of the output values for all inputs: this model usually results in a high bias error. We could increase the number of parameters and terms of the model to make it more flexible until it interpolates every training point perfectly (even when observation noise is present). Such model would likely be too complex, will not generalise and have high variance error: the latter is referred to as *overfitting*.

2.3.2 Regularisation

Without regularisation, ERM is generally an ill-posed problem. However as Tikhonov and Arsenin [6] pointed out, an appropriate choice of S can guarantee well-posedness and by extent generalisation of the ERM [4, 7]. The role of regularisation is exactly this: for a given set of model hyperparameters, it trades variance for bias during the training, i.e., while optimising $\boldsymbol{w}$. Strong regularisation drives the variance down and introduces bias in the model: this can be interpreted as forcing the model to be *simpler*. For a mathematical foundation of this effect, the reader is referred to [4, 5].

Adding appropriate regularisation to the training of the polynomial of order 19 significantly improves the approximation in Figure 2.1c: the shape of the curve reflects the trend of the data much better. The fluctuation of the error over the entire domain is therefore smaller, whereas most data points are still interpolated. In fact, it is much more likely the failure to interpolate on the left side of the curve is due to noisy observations. The regularisation function χ used was the l_2 norm which tries to prevent parameters growing to large values. Another popular regularisation function is the l_1 norm which tries to reduce the number of non-zero elements in $\boldsymbol{w}$ as is also the driving principle in Compressed Sensing [8]. Ridge regression is a prime example of l_2 regularisation whereas LASSO applies l_1 regularisation. Finally the Elastic Net approach combines both regularisers.

Although the analysis presented is theoretical (f is unknown, hence we can not compute the bias and variance explicitly over $\mathcal{X}$), we can make an important conclusion: should a method to compute an exact regularisation method applicable to all (hyper-)parameters be available, training models would be a lot simpler and (hyper-)parameters could be optimised based on training error. Such a method is unavailable unfortunately, and additionally the typical regularisers can not be applied on non-continuous (hyper-)parameters such as kernels or network architectures. A straightforward solution would be optimising λ along with $\boldsymbol{w}$, however this implies the regularisation constant is estimated from the training data and does not account for generalisation performance. Therefore, many model types have an explicit regularisation constant which is added as a hyperparameter, in addition to other hyperparameters influencing the model complexity and by extent, its variance. These hyperparameters are left to be determined prior to model training. In the context of adaptive surrogate modelling, our aim is to do this automatically as further discussed in Section 2.3.4

2.3.3 Regularisation through corruption

A different approach to regularisation builds on the idea that the root of the problem is that due to a limited set of training data, the knowledge of f is too limited and estimation of parameters is therefore troublesome. If hypothetically, an infinite set of training data was supplied, f is known and identifying a model of the right complexity minimizing both bias and variance is possible. This motivates the idea to corrupt a training set according to a known distribution and sample a new larger training set from the corrupting distribution. Examples of corrupting distributions are the Normal distribution, the Poisson distribution, or the Bernoulli distribution (randomly setting values of the training vectors to zero). This idea was presented as learning with *corrupted features*. It was shown that for some loss functions and corrupting distributions it is possible to consider an infinite number of corruptions for each training point, effectively marginalizing the corrupting distribution [9–11] and avoiding the additional computational cost of training on a larger training set.

The idea of corruption for regularisation is also very successful in the context of deep learning and is often referred to as *dropout* which corresponds to explicit generation of corrupted features in a network during training [12], and has lately been successfully applied to the layer level to enhance the generalisation performance of residual networks [13]. Also the frequently applied *data augmentation* pre-processing step can be partially regarded as an explicit corruption process[6]. In fact, even corruption due to implementation errors contribute to regularisation as Patrick Simard mentioned during a talk:

> *Gradient descent is so robust to errors in the gradient, your code probably has bugs that you are unaware of, but you don't need to fix them. It will act as an additional form of regularisation.*

2.3.4 Hyperparameter optimisation

The previous sections discussed the bias-variance trade-off involved in the choice of $\tilde{f}$, and how regularisation impacts this trade-off during the training but usually results in an extra hyperparameter[7]. What is still missing is a formulation for Λ to guide the search over the hyperparameter space to find optimal regularisation and configure model complexity. The formulation should reach beyond training error.

[6] Data augmentation is also required to be able to handle types of data symmetry that are not learned automatically by sparse network architectures: for instance convolutional neural networks excel at translational symmetry but need modification to be able to handle rotational symmetry if the training data is not augmented with rotated instances [14, 15].

[7] In case corruption is applied, the corrupting distribution and its parameters can also be regarded as extra hyperparameters.

At the same time the formulation should be sufficiently flexible to incorporate all requirements by the domain experts and users of the model. Figure 2.1d illustrates the effect of hyperparameter optimisation on the generalisation performance and overall quality of $\tilde{f}$: here the order of the polynomial model as well as the regularisation were optimised using the hereafter mentioned cross-validation technique as quality estimator. Instead of order 19, the optimisation over the order resulted in an order of 5. Comparing Figures 2.1c and 2.1d the right side of the interval is smoother and is intuitively a better approximation.

The most popular workaround available to determine the hyperparameters is to exclude part of the available data for training. One could randomly select 20% of the rows of X and Y as X_{test} and Y_{test} respectively, and keep it aside as test set. The approximation $\tilde{f}_\theta$ is then trained on the remaining 80% of the data, and Λ is defined as $\kappa(\tilde{f}_\theta(X_{\text{test}}), Y_{\text{test}})$. Doing so, the generalisation performance is (partially) tested explicitly at the cost of sacrificing some training data. However, this approach implies the test set is unbiased. A more involved (and more computationally intensive) strategy is K-fold cross-validation, where the data is randomly split into K folds. For each fold, a model is trained on all other $K-1$ folds and scored by evaluating κ on the on the model prediction for fold K. Λ is then the average score over the folds. Doing so, we can assure that during the optimisation, the model is verified on unseen data. Hence this approach aims to reduce variance.

Some other methods to estimate the model quality including a notion of the model complexity and associated risks with regards to the bias-variance trade-off include Akaike Information Criterion (AIC) [16], Linear Reference Model (LRM) [17], jack-knife [18, 19], etc. Finally, Λ may also be composed as a linear combination of several quality estimators, or the optimisation problem of Equation (2.1) can be optimised multi-objectively [20]. For the latter case not a single optimal model is found, but a pareto-optimal set $P \subset S$ representing the trade-off between the quality estimators. The latter combination approaches are very interesting to incorporate the model requirements as specified by the user.

2.4 The Bayesian view: Gaussian processes

The previous sections highlighted that, due to the unknown f and only a partial set of potentially noisy observations of it (D), there is an inherent uncertainty about the approximation. For many methods this uncertainty reappears in the optimisation of model (hyper)parameters and regularisation, hence estimation techniques are required to assure good generalisation.

A completely different view on the problem is offered by Bayesian statistics. This approach starts from the premise that the inherent uncertainty on $\tilde{f}$ should be

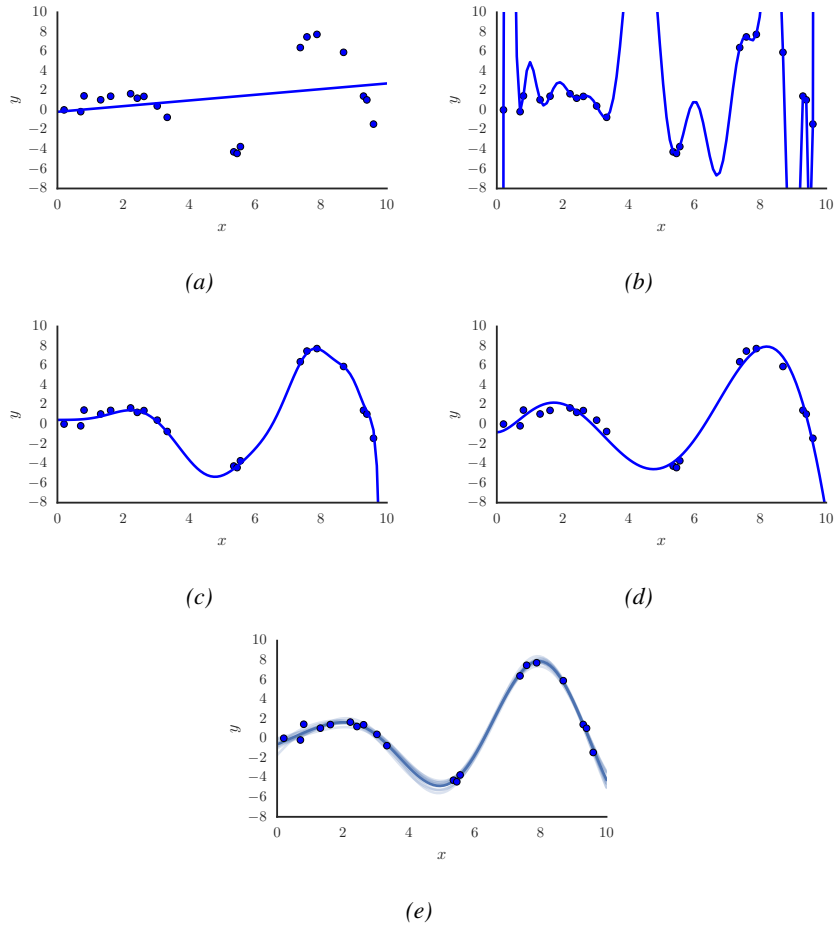


Figure 2.1: Empirical Risk Minimization: approximation of a 1D noisy signal with several models. a) First order polynomial model: this model is not sufficiently complex and suffers from high bias. b) Polynomial model of order 19, interpolating the training data literally, suffering from high variance. c) Polynomial model of order 19 with l_2 regularisation. Due to the order the model is too complex to approximate the data, however the regularisation has severely improved the generalisation performance. d) Both the polynomial order and regularisation constant are cross-validated, resulting in a satisfactory approximation. e) 10 samples (potential models) drawn from the posterior predictive distribution with RBF kernel after determining $\boldsymbol{\theta}$ with MLE. With exception of the gap in the middle the GP is quite confident about its prediction.

fully quantified. This idea is further explained by means of the popular Gaussian Process (GP). A GP is essentially a generalisation of a multivariate Gaussian distribution to an infinite number of dimensions. Drawing a sample from a GP results in a random function (like drawing a sample from n-dimensional Gaussian distribution results in a n-dimensional vector). The analogy continues as a GP is defined by a mean and covariance functions $\gamma(\boldsymbol{x})$ and $k(\boldsymbol{x}, \boldsymbol{x}')$ respectively, just like a multivariate Gaussian distribution is defined by its mean vector $\boldsymbol{\gamma}$ and covariance matrix $\boldsymbol{K}$.

Formally we can define GPs as follows:

Definition 2 (Gaussian Process)**.** *A GP is a collection of random variables, any finite number of which have a joint Gaussian distribution [21]. The GP can be used as a non-parametric prior over a latent function f:*

$$\begin{aligned} f &\sim \mathcal{GP}\left(\gamma(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')\right), \\ \gamma(\boldsymbol{x}) &= \mathbb{E}\left[f(\boldsymbol{x})\right], \\ k(\boldsymbol{x}, \boldsymbol{x}') &= \mathbb{E}\left[\left(f(\boldsymbol{x}) - \gamma(\boldsymbol{x})\right)\left(f(\boldsymbol{x}') - \gamma(\boldsymbol{x}')\right)\right]. \end{aligned}$$

The typical choice for the mean function is $\gamma(\boldsymbol{x}) = 0$: this is also the convention for this section. This may seem counter-intuitive at first, but can be achieved by shifting the training data prior to constructing a GP, or by constructing a hierarchical k and adding a bias kernel. The covariance function is chosen upfront and defines some properties such as smoothness, periodicity, trends or bias. It is usually parametrised by a set of kernel hyperparameters $\boldsymbol{\theta}_k$.

In a modelling scenario, an inherently infinite model specifies a finite Gaussian distribution due to the limited set of training data. This assumes the remainder of $\mathcal{X}$ is marginalised. The GP prior is conditioned on the training data, which results in a posterior distribution over $\boldsymbol{f}$ that "fits" the data. Denoting the training input data $\boldsymbol{X}$ and observations collected in $\boldsymbol{f} \in \mathbb{R}^n$, this can be written as $p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{\theta}_k)$. Here we assume $q = 1$, although GPs can also be used for multi-output functions f. The model specifies a finite Gaussian distribution for the training data:

$$\begin{aligned} \boldsymbol{f}|\boldsymbol{X}, \boldsymbol{\theta}_k &\sim \mathcal{N}(0, \boldsymbol{K}_{ff}), \\ p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{\theta}_k) &= (2\pi)^{-\frac{n}{2}} |\boldsymbol{K}_{ff}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\boldsymbol{f}^T \boldsymbol{K}_{ff}^{-1} \boldsymbol{f}\right), \end{aligned}$$

with the square covariance matrix $\boldsymbol{K}_{ff} \in \mathbb{R}^{n \times n}$ constructed by evaluating the covariance function k on the samples $\boldsymbol{x}_i$:

$$\boldsymbol{K}_{ff} = \begin{bmatrix} k(\boldsymbol{x}, \boldsymbol{x}_0) & \dots & k(\boldsymbol{x}_0, \boldsymbol{x}_{n-1}) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_{n-1}, \boldsymbol{x}_0) & \dots & k(\boldsymbol{x}_{n-1}, \boldsymbol{x}_{n-1}) \end{bmatrix}.$$

If $\boldsymbol{f}$ can be observed directly (and no noise is present) the expression for $p(\boldsymbol{f}|\boldsymbol{X})$ has no latent variables, but still depends on the kernel hyperparameters. Before further discussing these parameters, we first extend the GP formalism to incorporate observation noise. Formally the noise corruption is assigned a Gaussian prior:

$$\begin{aligned} y &= f(\boldsymbol{x}) + \epsilon, \\ \epsilon &\sim \mathcal{N}(0, \sigma_n^2). \end{aligned}$$

This turns $\boldsymbol{f}$ into a latent variable, for which the posterior distribution $p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{\theta})$ can be computed with Bayes rule. Here, $\boldsymbol{\theta} = (\boldsymbol{\theta}_k, \sigma_n^2)$. By specifying the likelihood distribution

$$\boldsymbol{y}|\boldsymbol{f}, \sigma_n^2 \sim \mathcal{N}(\boldsymbol{f}, \sigma_n^2 \boldsymbol{I}),$$

marginalising $\boldsymbol{f}$ is tractable and results in an analytical expression for the marginal likelihood. This is obtained by integrating the product of the prior on $\boldsymbol{f}$ (conditioned on $\boldsymbol{X}$) and the likelihood[8]:

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) &= \int p(\boldsymbol{y}|\boldsymbol{f}, \sigma_n^2)\, p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{\theta}_k)\, \mathrm{d}\boldsymbol{f}\,, \\ &= (2\pi)^{-\frac{n}{2}} |\boldsymbol{K}_{ff} + \sigma_n^2 \boldsymbol{I}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2} \boldsymbol{y}^T (\boldsymbol{K}_{ff} + \sigma_n^2 \boldsymbol{I})^{-1} \boldsymbol{y}\right). \end{aligned} \tag{2.3}$$

Note that the marginal likelihood incorporates the bias-variance trade-off: the determinant term restricts model complexity and reduces variance, whereas the exponential term promotes fitting the data. We now further explore the role of the hyperparameters and how they should be handled.

Ideally, in order to obtain predictions we are able to marginalise the hyperparameters and obtain the posterior $p(\boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{X})$ analytically according to Bayes rule:

$$p(\boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{X}) = \frac{p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})\, p(\boldsymbol{\theta})}{p(\boldsymbol{y}|\boldsymbol{X})}. \tag{2.4}$$

Hereafter, conditioning on $\boldsymbol{y}$ and $\boldsymbol{X}$ will be denoted by D. Under this setting, the posterior distribution on $f(\boldsymbol{x}_\star)$ of the *marginal GP* would be computed by marginalising $\boldsymbol{\theta}$:

$$p(f(\boldsymbol{x}_\star)|\boldsymbol{x}_\star, D) = \int p(f(\boldsymbol{x}_\star)|\boldsymbol{x}_\star, D, \boldsymbol{\theta})\, p(\boldsymbol{\theta}|D)\, \mathrm{d}\boldsymbol{\theta}\,. \tag{2.5}$$

This formulation for the posterior predictive distribution is completely (hyper-) parameter free and hence does not require any further optimisation. The first term is

[8] Marginal likelihood as in: marginalised over $\boldsymbol{f}$.

not problematic. Under the definition of GPs, the *posterior predictive distribution* is joint-Gaussian with the distribution over the observations:

$$\begin{bmatrix} \boldsymbol{y} \\ f(\boldsymbol{x}_\star) \end{bmatrix} \Bigg| \boldsymbol{\theta} \sim \mathcal{N}\left(0, \begin{bmatrix} \boldsymbol{K}_{ff} + \sigma_n^2 \boldsymbol{I} & \boldsymbol{K}_{f\star} \\ \boldsymbol{K}_{\star f} & \boldsymbol{K}_{\star\star} \end{bmatrix}\right).$$

In this expression $\boldsymbol{K}_{f\star} = \boldsymbol{K}_{\star f}^T$ represents the cross-covariance between $\boldsymbol{x}_\star$ and the training samples, and $\boldsymbol{K}_{\star\star} = k(\boldsymbol{x}_\star, \boldsymbol{x}_\star)$. Obtaining the posterior for $f(\boldsymbol{x}_\star)$ requires obtaining the conditional distribution given $\boldsymbol{y}$. This is straightforward as both random vectors are jointly Gaussian, and results in another (Gaussian) distribution:

$$\begin{aligned} f(\boldsymbol{x}_\star)|\boldsymbol{x}_\star, D, \boldsymbol{\theta} &\sim \mathcal{N}\left(\mu(\boldsymbol{x}_\star|\boldsymbol{\theta}), s^2(\boldsymbol{x}_\star|\boldsymbol{\theta})\right), \\ \mu(\boldsymbol{x}_\star|\boldsymbol{\theta}) &= \boldsymbol{K}_{\star f}\left(\boldsymbol{K}_{ff} + \sigma_n^2 \boldsymbol{I}\right)^{-1} \boldsymbol{y}, \qquad (2.6a) \\ s^2(\boldsymbol{x}_\star|\boldsymbol{\theta}) &= \boldsymbol{K}_{\star\star} - \boldsymbol{K}_{\star f}\left(\boldsymbol{K}_{ff} + \sigma_n^2 \boldsymbol{I}\right)^{-1} \boldsymbol{K}_{f\star}. \qquad (2.6b) \end{aligned}$$

Unfortunately, the second density of the integral in Equation (2.5) is often problematic to compute. The denominator of Equation (2.4) is the root of the problem as the hyperparameters typically occur non-linearly in kernel functions, making marginalisation of $\boldsymbol{\theta}$ intractable in most cases. An excellent description of the difficulty of propagating distributions through non-linearities, a key problem for Bayesian methods, is given by Damianou [22].

Instead, the common way to proceed is to obtain a point estimate for $\boldsymbol{\theta}$ by numerically optimizing the (log of the) numerator of Equation (2.4), which is tractable. This approach is referred to as Maximum Likelihood Estimation (MLE) and represents a point estimate of Equation (2.5). The results represents the most likely posterior predictive distribution for the latent function f (w.r.t. $\boldsymbol{\theta}$) which corresponds to a Gaussian distribution of approximating functions interpolating the observations (apart from a "tolerance" defined by σ_n^2). Hence the posterior predictive distribution can be regarded as an analytical weighting function for an infinite ensemble of approximating functions. This is illustrated in Figure 2.1e: here 10 samples for f are drawn from $p(f|\boldsymbol{x}, \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{\theta}), \forall \boldsymbol{x} \in \mathcal{X}$. In practise, the mean of this posterior predictive distribution is typically used as $\tilde{f}$, whereas its variance may be used for different applications such as sampling as, for instance, is the case in the Efficient Global Optimisation (EGO) algorithm.

In case sufficient data is available, and an appropriate kernel was chosen which represents the covariance structure of f, the optimisation of the numerator of Equation (2.4) is usually successful as the global optimum is isolated and quite sharp. For some applications such as for instance Bayesian Optimisation (BO), the point estimate can be insufficient as the likelihood surface becomes multimodal. It is then interesting to incorporate the uncertainty on $\boldsymbol{\theta}$ somehow. One

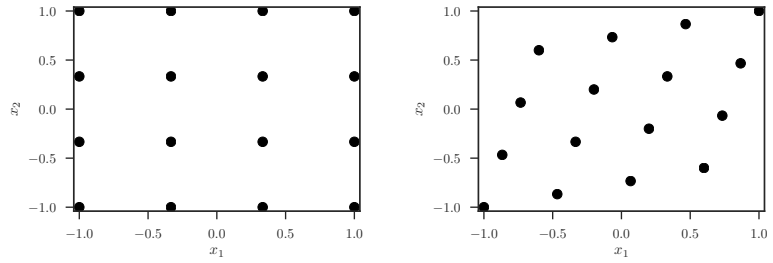
option is to approximate Equation (2.5) by sampling $\boldsymbol{\theta}$ from the numerator of Equation (2.4) with Markov Chain Monte Carlo (MCMC). Under this setting each sample corresponds to a different posterior distribution on $f(\boldsymbol{x}_\star)$, hence the mean and variance of the posterior predictive distribution of the marginal GP can be approximated. Note that this distribution is not necessarily Gaussian: it was for instance shown that marginalising the common σ^2 kernel parameter (which can still be tractable) the predictive distribution changes into a Student-t distribution [23]. Given P hyperparameter $\boldsymbol{\theta}_i$ sampled from the numerator of Equation (2.4) and following the law of total cumulance [24]:

$$
\begin{aligned}
\mathbb{E}\left[f(\boldsymbol{x}_\star)|\boldsymbol{x}_\star, D\right] &= \mathbb{E}_{p(\boldsymbol{\theta}|D)}\left[\mathbb{E}\left[f(\boldsymbol{x}_\star)|\boldsymbol{x}_\star, D, \boldsymbol{\theta}\right]\right], \\
&= \mathbb{E}_{p(\boldsymbol{\theta}|D)}\left[\mu(\boldsymbol{x}_\star|\boldsymbol{\theta})\right], \\
&\overset{\text{MCMC}}{\approx} \frac{1}{P}\sum_{i=1}^{P}\mu(\boldsymbol{x}_\star|\boldsymbol{\theta}_i), \\
&= \tilde{\mathbb{E}}\left[p(f(\boldsymbol{x}_\star)|\boldsymbol{x}_\star, D)\right]. \\
\text{Var}\left[f(\boldsymbol{x}_\star)|\boldsymbol{x}_\star, D\right] &= \mathbb{E}_{p(\boldsymbol{\theta}|D)}\left[\text{Var}\left[f(\boldsymbol{x}_\star)|\boldsymbol{x}_\star, D, \boldsymbol{\theta}\right]\right] \\
&\quad + \text{Var}_{p(\boldsymbol{\theta}|D)}\left[\mathbb{E}\left[f(\boldsymbol{x}_\star)|\boldsymbol{x}_\star, D, \boldsymbol{\theta}\right]\right], \\
&= \mathbb{E}_{p(\boldsymbol{\theta}|D)}\left[s^2(\boldsymbol{x}_\star|\boldsymbol{\theta})\right] + \text{Var}_{p(\boldsymbol{\theta}|D)}\left[\mu(\boldsymbol{x}_\star|\boldsymbol{\theta})\right], \\
&\overset{\text{MCMC}}{\approx} \frac{1}{P}\sum_{i=1}^{P}s^2(\boldsymbol{x}_\star|\boldsymbol{\theta}_i) + \left(\mu(\boldsymbol{x}_\star|\boldsymbol{\theta}_i) - \tilde{\mathbb{E}}\left[p(f(\boldsymbol{x}_\star)|\boldsymbol{x}_\star, D)\right]\right)^2.
\end{aligned}
$$

For both MCMC approximations, the samples $\boldsymbol{\theta}_i$ are drawn from the numerator of Equation (2.4). A comparison of MLE and MCMC approximation of the marginal GP for BO is part of Chapter 5. Finally, some analytical approximations of the marginal GP exist such as the method proposed by Garnett et al. [25]. In the same work, an information-theoretic sampling method is proposed known as Bayesian Active Learning by Disagreement (BALD) which aims to select observations to reduce the uncertainty on $\boldsymbol{\theta}$, and hence enhances the quality of the point estimate.

2.5 Experimental design

So far unspecified is the choice of D. Roughly the data for constructing the surrogate models can be completely specified upfront (in the form of a dataset or a set of data points X to be evaluated), or iteratively constructed. For the latter approach, a refined set of model requirements and the goal of the process is required as it greatly affects the choice of samples to be evaluated. The choice of samples X to be evaluated is referred to as the *experimental design*.



(a) *(b)*

Figure 2.2: One-shot experimental designs: (a) 16-point grid design (4-level factorial design). (b) 16-point optimised LHD which accounts for (one-dimensional) projective properties. Note that even this fairly optimised design misses two corners.

2.5.1 One-shot design

The traditional approaches to generate an experimental design are the one-shot designs. Prior to any evaluation, all samples are selected in a space-filling manner: at this point no further information is available due to the black-box assumption on the simulator itself (as part of the data-driven approach). Therefore the information density should be approximately equal over the entire design space and the samples are to be distributed uniformly. To this end, several approaches related to Design of Experiments (DoE) have been developed. However, only the space-filling aspect has an impact within the context of computer experiments, as other criteria such as blocking and replication lose their relevance [26]. This led to the transition and extension of these existing statistical methods to computer experiments [26, 27]. Widely applied are the factorial designs (grid-based) [28] and optimal (maximin) Latin Hypercube Designs (LHDs) [29]. An LHD avoids collapsing points should the input space be projected onto a single parameter. This concept was further generalised by Roshan et al. [30] by considering optimal space-fillingness in all possible subspaces. The latter approach however is very difficult and computationally demanding. Other approaches include maximin and minimax designs [31], Box-Behken [32], central composite designs [33], Maximum Projection designs [30] and (quasi) Monte Carlo methods [34–36].

Despite their widespread usage, these standard approaches to generate experimental designs come with a number of disadvantages. First and foremost: the most qualitative designs (with the best space-filling properties) can be extremely complex to generate (especially for problems with a high-dimensional input space) due to their geometric properties. For instance, generating an LHD with optimal maximin distance is very time-consuming process. In fact, the generation of an optimal

LHD is almost a discipline on its own, with several different methods for faster and reliable generation [29, 37]. Fortunately, once a design is generated, it can be reused. For some other design methodologies, it is not possible to generate them for an arbitrary size. Given the expensive nature of each evaluation this can result in an unacceptable growth of required simulation time. Factorial designs for instance always have size k^d with level k and dimension d, making them infeasible choices for problems with many input parameters.

Another disadvantage of one-shot methodologies is the arbitrary choice of size of the design. The choice should depend entirely on the nature of the problem (i.e., larger design spaces with more complex behaviour require more evaluations). However, this information is unavailable at the time the design is generated due to the black-box assumption. Hence, one-shot approaches risk selecting too few data points resulting in an underfitted model, or selecting too much data points causing loss of time and computational resources.

2.5.2 Sequential design

As a solution, sequential design was adopted. This methodology starts from a very small one-shot design to initiate the process. After evaluation of these samples a model is built, and a loop is initiated which is only terminated when either one of the specified stopping criteria is met. Within the loop, an adaptive sampling algorithm is run to select additional data points for evaluation which are used to update the model.

This approach has a number of advantages. First of all, constraints on the surrogate modelling process can be explicitly imposed through the stopping criteria. Typical criteria include how well the model satisfied the model requirements, a maximum number of allowed evaluations, or a maximum runtime. Secondly, the adaptive sampling method can be designed to select new data points specifically in terms of the requirements. Sampling to obtain a globally accurate model will differ from sampling to discover class boundaries, or sampling to obtain optima. These choices can also be guided by all information available about the input-output behaviour: when n samples have been selected, a history of intermediate models and all simulator responses is available to guide the selection of new samples. Because of the information available, this selection no longer has to be purely based on a black-box approach, and information can be exploited.

Roughly, all methods for sequential adaptive sampling are based on any of the following criteria (discussed more in detail in Chapter 3):

- Distance to neighbouring points (space-filling designs)

- Identification of optima
- Model uncertainty
- Non-linearity of the response
- Feasibility of the candidate point with respect to constraints

Depending on the goal and model requirements, a strategy can be designed involving a complex combination of these criteria. In surrogate modelling, the concept of sequential design has been applied in several succesful applications [38–43] and is crucial to obtain data efficiency. In fact all chapters of this dissertation involve innovations and application of sequential design. A more in depth overview of sequential design and existing methodologies is given in Chapter 3.

2.6 Error functions

The last aspect of the adaptive modelling formalism to be discussed briefly is the error function. Formally the error function introduces an order relation on $\mathbb{C}^{n\times q}$ based on similarity to the observations of f collected in the matrix $\boldsymbol{Y}$. For notational simplicity $q = 1$ is assumed hereafter[9] and $\boldsymbol{y}$, $\tilde{\boldsymbol{y}}$ denote the observations and predictions respectively. In addition we also introduce $\bar{\boldsymbol{y}}$, the mean of $\boldsymbol{y}$.

Distinguishing between all available error functions can be tricky, and classifying them is difficult due to the many aspects. Roughly all error functions can be categorised in two groups:

1. Absolute error: non-unitless functions, this implies their range depends on the range of the output parameters. This leads to interpretation issues as it can be difficult to determine the actual prediction quality represented by a certain score. This is undesirable in the context of surrogate modelling, as the score reported by the error function is usually used as a stopping criterion. The most common example is the Root-Mean-Square Error (RMSE) which is very popular within the field of machine learning [44]. However it has also been criticised for being very *pessimistic* as the error is dominated by large errors whereas small errors are ignored. Furthermore this error measure is troublesome to interpret.
2. Relative error: this type of error functions is more popular within engineering, as it allows formulation of goals and stopping criteria without knowing the output range upfront. A figure of 5% refers to some kind of global average relative error, but can be computed in several different ways (depending on

[9] This is without loss of generality, as multiple outputs can either be stacked into a single vector, or the aggregation mechanisms of the error function can be expanded over multiple columns.

what kind of reference and aggregation is used). A popular example is the Root-Relative-Square Error (RRSE) error function which compares the error of the predicted values to the errors of the predictions by the simplest model available: the constant mean model (denoted by $\bar{\boldsymbol{y}}$). The RRSE is related to coefficient of determination, a popular metric in statistics.

Throughout the choice of the error function it is important to keep the properties of the error functions in mind. The choice between absolute or relative error is highly related to the available information on the output (such as the presence of some pre-evaluated data, or physical constraints). A second consideration is the amount of outliers and their relevance to the model quality estimation: this should motivate the choice for a pessimistic or an optimistic function. Generally, maximum aggregation is most pessimistic as the largest error outplays all (potentially good) predictions. Arithmetic averages are less pessimistic but are not as optimistic as error functions applying geometric averages. However, error functions based on the latter mechanism sometimes suffer from divisions by zero if one of the predictions is perfect and makes no error (depends on the formulation, e.g., the Harmonic Average Error (HAE) is not affected). An excellent detailed survey can be found in Gorissen et al. [45]. Another in-depth description is given by Li and Zhao [44].

2.7 Application domains

Because the concept of surrogate modelling is both flexible as well as generic, allowing several modifications tailored for the task at hand, it has been applied in wide range of fields including:

- **Economics**: Sensitivity analysis in investment problems [46]
- **Operations Research**: modelling business networks [47]
- **Robotics**: evolution of gait patterns of four-legged walking robots [48]
- **Electronics**: Mobile antenna designs [49, 50]
- **Physics**: study of proton beams [51]
- **Chemistry**: prediction of fibrinogen absorption onto polymer surfaces [52]
- **Automotive**: study the effect of frontal impact on a vehicle [53], traffic flow [54]
- **Biology**: prediction and explanation of biodiversity data [55, 56]
- **Geology**: modelling of (oil, gas, water,...) reservoirs [57, 58]

Table 2.1: Error functions: overview of common error functions for observations $\boldsymbol{y}$ and predictions $\tilde{\boldsymbol{y}}$. For relative error functions, $\bar{\boldsymbol{y}}$ denotes the mean of $\boldsymbol{y}$.

Name	Formula
Mean Square Error (MSE)	$\sum_{i=0}^{n-1} \frac{(y_i - \tilde{y}_i)^2}{n}$
Root-Mean-Square Error (RMSE)	$\sqrt{\text{MSE}}$
Average Absolute Error (AAE)	$\frac{1}{n} \sum_{i=0}^{n-1} \lvert y_i - \tilde{y}_i \rvert$
Maximum Absolute Error (MAE)	$\max_{i=0,\ldots,n-1} \lvert y_i - \tilde{y}_i \rvert$
Geometric Average Error (GAE)	$\left(\prod_{i=0}^{n-1} \sqrt{(y_i - \tilde{y}_i)^2} \right)^{\frac{1}{n}}$
Harmonic Average Error (HAE)	$\left(\frac{1}{n} \sum_{i=0}^{n-1} \frac{1}{\sqrt{(y_i - \tilde{y}_i)^2}} \right)^{-1}$
Average Relative Error (ARE)	$\frac{1}{n} \sum_{i=0}^{n-1} \frac{\lvert y_i - \tilde{y}_i \rvert}{\lvert y_i \rvert}$
Maximum Relative Error (MRE)	$\max_{i=0,\ldots,n-1} \frac{\lvert y_i - \tilde{y}_i \rvert}{\lvert y_i \rvert}$
Root-Relative-Square Error (RRSE)	$\sqrt{\frac{\sum_{i=0}^{n-1} (y_i - \tilde{y}_i)^2}{\sum_{i=0}^{n-1} (y_i - \bar{y})^2}}$
Coefficient of deteriniation (R^2)	$R^2 = 1 - \text{RRSE}^2$
Bayesian Error Estimation Quotient (BEEQ)	$\left(\prod_{i=0}^{n-1} \frac{\lvert y_i - \tilde{y}_i \rvert}{\lvert y_i - \bar{y} \rvert} \right)^{\frac{1}{n}}$

- **Meteorology**: studying the effect of emission reduction on ozone concentrations [59]
- **Sociology**: modelling innovation diffusion [60]
- **Medicine**: modelling colon coloration [61]
- **Machine Learning**: approximating real-world expensive benchmark applications [62], Bayesian optimisation, Hyperparameter optimisation [63]

2.8 Criticism

Due to the wide range of applications, use cases in many fields and the links with machine learning and distributed computing, the concept of surrogate modelling can be regarded as very multidisciplinary. This makes surrogate modelling a very exciting and fruitful area to research but also holds a risk: being left with no identity as science advances and due to overlaps with other fields, as well as a very scattered terminology because of the many synonyms for similar concepts. During discussions with colleagues, or the review phase of publications, projects proposals or grants this was a recurring item of criticism.

To some degree, this criticism is justified. The observation of some overlap between surrogate modelling research and other fields is correct and the overlap is perhaps the strongest with the field of machine learning. Evolutions such as the focus on *big-data* and the rise of *deep learning* had a large impact on technology, the internet and society. This led to major attention from researchers and investors and resulted in broad attention for Artificial Intelligence (AI) and Machine Learning (ML) in general, including coverage in the press and an explosion in the number of conference attendees and AI start-ups.

Due to this strong evolution the overlap has only grown over the years. There is no difference between surrogate modelling based on a fixed set of data and predictive modelling as applied in machine learning (and statistics). The idea of adaptive modelling is strongly related with three of the fundamental principles of AutoML [64] and sequential design shares many similarities with active learning [65–68]. Surrogate-Based Optimisation (SBO) has been applied under the name of BO with its own recurring workshop at the popular Neural Information Processing Systems (NIPS) conference. Distributed computing and High Performance Computing (HPC) in general have been fundamental to accomplish big-data processing and deep learning which have resulted in popular computation frameworks such as TensorFlow [69].

However, despite the strong and successful efforts of the machine learning com-

munity to avoid pure theoretical work, attempts to value application on real data and efforts to provide implementations, many of the advancements of the last years in machine learning do not reach the field of engineering. This creates a wide and interesting area for countless improvements. Vice versa, this distance between communities results in the reinvention of concepts such as the use of SBO/BO for expensive hyperparameter optimisation [63, 70]. Eventually, the surrogate modelling concept also found a way into machine learning itself such as their use to produce benchmark applications closer to real-world scenarios [71].

Within the field of machine learning, surrogate modelling can be identified with the subfield of Data-Efficient Machine Learning (DEML). Because of the distance between the engineering world and the machine learning community, the need for work in this gap is now even more important than it was before. This is exactly the goal for surrogate modelling researchers as they continue to study, adapt and implement the advancements of machine learning into engineering applications and design and analysis of complex systems, as is also the scope and focus of this dissertation. In dialogue with engineers it is also possible to move from a purely data-driven approach to a more hybrid approach.

2.9 SUMO Toolbox

Designed as a research platform for sequential sampling and adaptive modelling using MATLAB, the SUMO toolbox [17] has grown into a mature design tool for surrogate modelling with sequential design offering a large variety of algorithms for simulators with continuous output. The software design is fully object-oriented allowing high-extensibility of its capabilities. By default, the platform follows the integrated modelling flow with sequential design, but can also be configured to approximate data sets, use a one-shot design etc. Recently, the platform has also been extended to offer support for several classification algorithms by including several implementations and linking the WEKA library [72].

Figure 2.3 illustrates the design goals of the SUMO Toolbox. Expensive computer simulations of complex black-box systems with several design parameters are approximated by a cheap-to-evaluate model, and the toolbox can also approximate outputs with a discrete set of labels by training a classifier. To obtain these goals, the SUMO Toolbox offers sequential sampling and adaptive modelling in a highly configurable environment which is easy to extend due to the microkernel design philosophy as illustrated in Figure 2.4. Distributed computing support for evaluations of data points is also available, as well as multi-threading to support the usage of multi-core architectures for regression modelling and classification.

Many different plug-ins are available for each of the different sub-problems: model

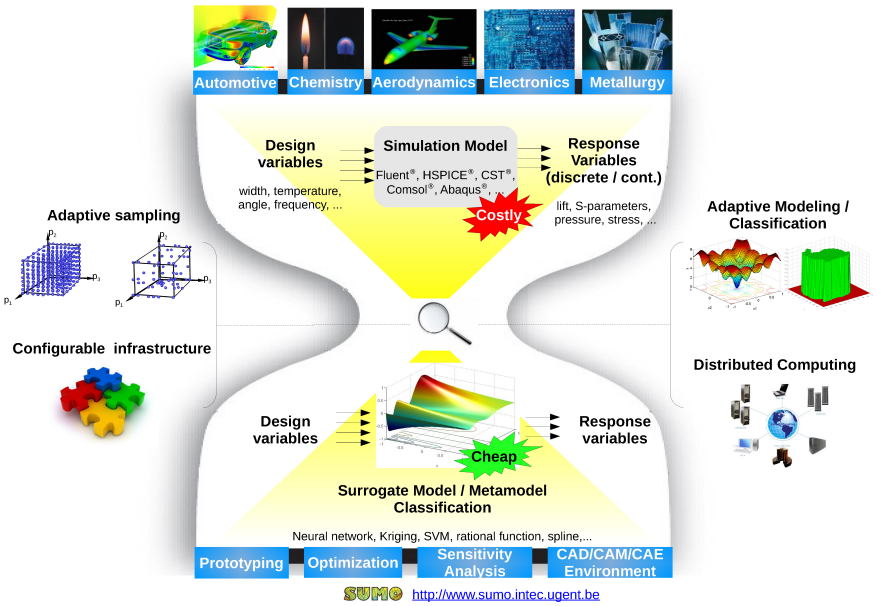


Figure 2.3: SUMO Toolbox: design philosophy.

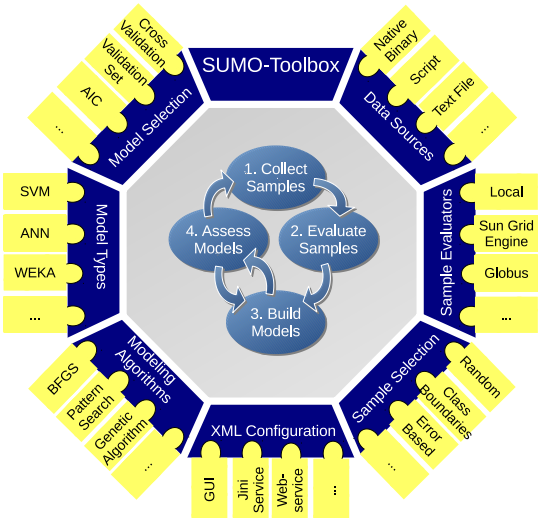


Figure 2.4: SUMO Toolbox: microkernel architecture.

types (rational functions, Kriging [73], splines, Support Vector Machines (SVM) [74–76], ANN, Extreme Learning Machines (ELM) [77], Least-Squares Support Vector Machine (LS-SVM) [78], Random Forests (RF) [79]), hyperparameter

optimisation algorithms including Particle Swarm Optimisation (PSO) [80], EGO [81], Simulated Annealing (SA), Genetic Algorithm (GA) and sample selection (random, error based, density based [82, 83], hybrid [84]), Design of Experiments (LHD [29, 37], Box-Bhenken [32]), and sample evaluation methods (local, on a cluster or grid). The behaviour of each software component is configurable through a central eXtensible Markup Language (XML) file and components can easily be added, removed or replaced by custom implementation.

During the adaptive modelling step, the Toolbox uses the methodology described in Section 2.3.4 for model selection to guide the hyperparameter optimisation. By combining several quality measures into a single multi-objective measure Λ, the toolbox supports multi-objective hyperparameter optimisation. This results in a set of Pareto-optimal solutions representing the trade-off between both objectives, instead of a single optimal solution [20]. In addition, several error functions κ are supported.

The SUMO Toolbox is free for academic use and is available for download at http://sumo.intec.ugent.be. It can be installed on any platform supported by MATLAB. In addition, a link can be found to the available documentation and tutorials to install and configure the toolbox including some of its more advanced features. News items concerning new releases, additional features and updates can also be found at the same web page.

2.10 Conclusion

Starting from the framed scope as outlined by the introduction, all important subproblems and different aspects regarding surrogate modelling were outlined including the data collection strategy and the model selection problem. It was shown how the latter is a cross-over of different points of view including the user requirements as well as technical considerations. It was highlighted that the questions raised during this process do not necessarily come with a single response or off-the-shelf solutions: e.g., the choice of error function, the optimisation of the hyperparameters and selection of new samples are crucial aspects to obtain an appropriate model which satisfies the specified requirements. There is an important role here for the field of surrogate modelling. With these foundations, the important terminology and concepts were covered and the research goals of this dissertation on adaptive modelling and sequential sampling can be explored.

2.11 References

[1] D. Gorissen. "Grid-enabled adaptive surrogate modeling for computer aided engineering". PhD thesis. Ghent University, 2010, XIV, 362 [in multiple pagination]. ISBN: 9789085783503.

[2] D. Gorissen, T. Dhaene, and F. D. Turck. "Evolutionary model type selection for global surrogate modeling". *Journal of Machine Learning Research* 10.Sep (2009), pp. 2039–2078.

[3] I. Couckuyt, D. Gorissen, F. De Turck, and T. Dhaene. "Automatic surrogate model type selection during the optimization of expensive black-box problems". *Proceedings of the 44th Conference on Winter Simulation*. Phoenix, AZ, USA: Institute of Electrical and Electronics Engineers, Inc, Dec. 2011, pp. 4274–4284.

[4] V. Vapnik. *The Nature of Statistical Learning Theory*. 2nd ed. Information Science and Statistics. Springer-Verlag New York, 2000. ISBN: 978-0-387-98780-4.

[5] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. 2nd ed. Springer Series in Statistics. Spinger-Verlag New York, 2009, pp. XXII, 745. ISBN: 978-0-387-84857-0.

[6] A. N. Tikhonov and V. Y. Arsenin. "Solutions of ill-posed problems" (1977).

[7] S. Mukherjee, P. Niyogi, T. Poggio, and R. Rifkin. "Learning theory: stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization". *Advances in Computational Mathematics* 25.1-3 (2006), pp. 161–193.

[8] E. J. Candès and M. B. Wakin. "An introduction to compressive sampling". *IEEE signal processing magazine* 25.2 (2008), pp. 21–30.

[9] L. Van Der Maaten, M. Chen, S. Tyree, and K. Q. Weinberger. "Learning with Marginalized Corrupted Features". *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*. Ed. by S. Dasgupta and D. McAllester. Proceedings of Machine Learning Research. Atlanta, USA: PMLR, 2013, pp. 410–418.

[10] N. Chen, J. Zhu, J. Chen, and B. Zhang. "Dropout Training for Support Vector Machines". *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*. AAAI'14. Québec City, Québec, Canada: AAAI Press, 2014, pp. 1752–1759.

[11] N. Chen, J. Zhu, J. Chen, and T. Chen. "Dropout Training for SVMs with Data Augmentation". *arXiv preprint arXiv:1508.02268* (2015).

[12] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958.

[13] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger. "Deep Networks with Stochastic Depth". *Computer Vision – ECCV 2016: 14th European Conference, Proceedings, Part IV*. Ed. by B. Leibe, J. Matas, N. Sebe, and M. Welling. Amsterdam, The Netherlands: Springer International Publishing, Oct. 2016, pp. 646–661. ISBN: 978-3-319-46493-0.

[14] S. Dieleman, K. W. Willett, and J. Dambre. "Rotation-invariant convolutional neural networks for galaxy morphology prediction". *Monthly notices of the royal astronomical society* 450.2 (2015), pp. 1441–1459.

[15] S. Dieleman, J. De Fauw, and K. Kavukcuoglu. "Exploiting Cyclic Symmetry in Convolutional Neural Networks". *Proceedings of the 33rd International Conference on Machine Learning (ICML-16)*. Ed. by M.-F. Balcan and K. Q. Weinberger. Proceedings of Machine Learning Research. Manhattan, New York, USA: PMLR, 2016, pp. 1889–1898.

[16] H. Akaike. "A new look at the statistical model identification". *Automatic Control, IEEE Transactions on* 19.6 (1974), pp. 716–723.

[17] D. Gorissen, K. Crombecq, I. Couckuyt, P. Demeester, and T. Dhaene. "A Surrogate Modeling and Adaptive Sampling Toolbox for Computer Based Design". *Journal of Machine Learning Research* 11 (2010). Available at http://sumo.intec.ugent.be, pp. 2051–2055.

[18] J. W. Tukey. "Bias and confidence in not-quite large samples". *Annals of Mathematical Statistics*. Vol. 29. 2. Inst. Matchematical Statistics. IMS Business Office-Suite 7, 3401 Investment Blvd, Hayward, CA 949545, 1958, pp. 614–614.

[19] B. Efron and C. Stein. "The jackknife estimate of variance". *The Annals of Statistics* (1981), pp. 586–596.

[20] D. Gorissen, I. Couckuyt, E. Laermans, and T. Dhaene. "Pareto-based multi-output metamodeling with active learning". *Proceedings of the 11th International Conference on Engineering Applications of Neural Networks (EANN 2009)*. London, England: Springer, Aug. 2009.

[21] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. MIT Press, 2006. ISBN: 026218253X.

[22] A. Damianou. "Deep Gaussian processes and variational propagation of uncertainty". PhD thesis. University of Sheffield, 2015.

[23] R. B. Gramacy and D. W. Apley. "Local Gaussian process approximation for large computer experiments". *Journal of Computational and Graphical Statistics* 24.2 (2015), pp. 561–578.

[24] D. R. Brillinger. "The calculation of cumulants via conditioning". *Annals of the Institute of Statistical Mathematics* 21.1 (1969), pp. 215–218.

[25] R. Garnett, M. A. Osborne, and P. Hennig. "Active learning of linear embeddings for Gaussian processes". *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence*. Ed. by Z. M.L. and T. J. Quebec, Canada: AUAI Press, 2014, pp. 230–239.

[26] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. "Design and analysis of computer experiments". *Statistical science* (1989), pp. 409–423.

[27] J. P. Kleijnen. *Design and Analysis of Simulation Experiments*. 2nd ed. New York City, USA: Springer International Publishing, 2015, p. 337. ISBN: 978-3-319-18086-1.

[28] D. C. Montgomery. *Design and analysis of experiments*. 8th ed. Hoboken, New Jersey, USA: John Wiley & Sons, 2012, p. 752. ISBN: 978-1118146927.

[29] E. R. Van Dam, B. Husslage, D. Den Hertog, and H. Melissen. "Maximin Latin hypercube designs in two dimensions". *Operations Research* 55.1 (2007), pp. 158–169.

[30] J. V. Roshan, G. Evren, B. Shan, et al. "Maximum projection designs for computer experiments". *Biometrika* 102.2 (2015), pp. 371–380.

[31] M. E. Johnson, L. M. Moore, and D. Ylvisaker. "Minimax and maximin distance designs". *Journal of statistical planning and inference* 26.2 (1990), pp. 131–148.

[32] G. E. Box and D. W. Behnken. "Some new three level designs for the study of quantitative variables". *Technometrics* 2.4 (1960), pp. 455–475.

[33] R. H. Myers, D. C. Montgomery, and C. M. Anderson-Cook. *Response Surface Methodology: Process and Product Optimization Using Designed*

Experiments. 4th ed. John Wiley & Sons, 2016, p. 856. ISBN: 978-1-118-91601-8.

[34] W. Hendrickx and T. Dhaene. "Sequential design and rational metamodelling". *Proceedings of the 38th Conference on Winter Simulation*. Ed. by M. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines. Orlando, FL, USA: Institute of Electrical and Electronics Engineers, Inc, Dec. 2005, pp. 290–298.

[35] R. Jin, W. Chen, and A. Sudjianto. "An efficient algorithm for constructing optimal design of computer experiments". *Journal of Statistical Planning and Inference* 134.1 (2005), pp. 268–287.

[36] H. Niederreiter. *Random Number Generation and quasi-Monte Carlo Methods*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1992. ISBN: 0-89871-295-5.

[37] F. A. Viana, G. Venter, and V. Balabanov. "An algorithm for fast optimal Latin hypercube design of experiments". *International journal for numerical methods in engineering* 82.2 (2010), pp. 135–156.

[38] J. Aernouts, I. Couckuyt, K. Crombecq, and J. J. Dirckx. "Elastic characterization of membranes with a complex shape using point indentation measurements and inverse modelling". *International Journal of Engineering Science* 48.6 (2010), pp. 599–611.

[39] D. Stephens, D. Gorissen, K. Crombecq, and T. Dhaene. "Surrogate based sensitivity analysis of process equipment". *Applied Mathematical Modelling* 35.4 (2011), pp. 1676–1687.

[40] S. Koziel and L. Leifsson. *Surrogate-Based Modeling and Optimization: Applications in Engineering*. SpringerLink : Bücher. Springer New York, 2013. ISBN: 9781461475514.

[41] D. Deschrijver, K. Crombecq, H. M. Nguyen, and T. Dhaene. "Adaptive Sampling Algorithm for Macromodeling of Parameterized-Parameter Responses". *Microwave Theory and Techniques, IEEE Transactions on* 59.1 (2011), pp. 39–45.

[42] D. Deschrijver, F. Vanhee, D. Pissoort, and T. Dhaene. "Automated near-field scanning algorithm for the EMC analysis of electronic devices". *Electromagnetic Compatibility, IEEE Transactions on* 54.3 (2012), pp. 502–510.

[43] S. Aerts, D. Deschrijver, W. Joseph, L. Verloock, F. Goeminne, L. Martens, and T. Dhaene. "Exposure assessment of mobile phone base station radiation

in an outdoor environment using sequential surrogate modeling". *Bioelectromagnetics* (2013).

[44] X. R. Li and Z. Zhao. "Evaluation of estimation algorithms part I: incomprehensive measures of performance". *Aerospace and Electronic Systems, IEEE Transactions on* 42.4 (2006), pp. 1340–1358.

[45] D. Gorissen, I. Couckuyt, E. Laermans, and T. Dhaene. "Multiobjective global surrogate modeling, dealing with the 5-percent problem". *Engineering with Computers* 26.1 (2010), pp. 81–98.

[46] W. J. Van Groenendaal and J. P. Kleijnen. "Deterministic versus stochastic sensitivity analysis in investment problems: an environmental case study". *European Journal of Operational Research* 141.1 (2002), pp. 8–20.

[47] T. W. Tewoldeberhan and A. Verbraeck. "Using web services and artificial intelligence techniques to develop simulation models of business networks". *Proceedings the 15th European Simulation Symposium*. Ed. by A. Verbraeck and H. Vlatka. 2003. ISBN: 3-936150-28-1.

[48] I. Dahm and J. Ziegler. "Using artificial neural networks to construct a metamodel for the evolution of gait patterns". *Proceedings of the 5th International Conference on Climbing and Walking Robots (CLAWAR 2002)*. Citeseer. 2002, p. 825.

[49] S. Xiao, B.-Z. Wang, X. Zhong, and G. Wang. "Wideband mobile antenna design based on artificial neural network models". *International Journal of RF and Microwave Computer-Aided Engineering* 13.4 (2003), pp. 316–320.

[50] S. Koziel and S. Ogurtsov. *Antenna design by simulation-driven optimization*. Springer International Publishing, 2014. ISBN: 978-3-319-04366-1.

[51] H. K. H. Lee, B. Sanso, W. Zhou, and D. M. Higdon. "Inference for a proton accelerator using convolution models". *Journal of the American Statistical Association* 103.482 (2008), pp. 604–613.

[52] J. R. Smith, D. Knight, J. Kohn, K. Rasheed, N. Weber, V. Kholodovych, and W. J. Welsh. "Using surrogate modeling in the prediction of fibrinogen adsorption onto polymer surfaces". *Journal of chemical information and computer sciences* 44.3 (2004), pp. 1088–1097.

[53] R. Yang, N. Wang, C. Tho, J. Bobineau, and B. Wang. "Metamodeling development for vehicle frontal impact simulation". *Journal of Mechanical Design* 127.5 (2005), pp. 1014–1020.

[54] E. I. Vlahogianni. "Optimization of traffic forecasting: Intelligent surrogate modeling". *Transportation Research Part C: Emerging Technologies* 55 (2015), pp. 14–23.

[55] D. R. Stockwell and A. T. Peterson. "Effects of sample size on accuracy of species distribution models". *Ecological modelling* 148.1 (2002), pp. 1–13.

[56] D. Stockwell and A. T. Peterson. "Comparison of resolution of methods used in mapping biodiversity patterns from point-occurrence data". *Ecological indicators* 3.3 (2003), pp. 213–221.

[57] S. D. Mohaghegh, A. Modavi, H. Hafez, M. Haajizadeh, and S. Guruswamy. "Development of surrogate reservoir model (SRM) for fast track analysis of a complex reservoir". *International Journal of Oil, Gas and Coal Technology* 2.1 (2009), pp. 2–23.

[58] H. Hamdi, I. Couckuyt, M. C. Sousa, and T. Dhaene. "Gaussian Processes for history-matching: application to an unconventional gas reservoir". *Computational Geosciences* (2017), pp. 1–21.

[59] C. Mensink, L. Delobbe, and A. Colles. "A policy oriented model system for the assessment of longterm effects of emission reductions on ozone". *Air Pollution Modeling and Its Application XV*. Ed. by C. Borego and G. Schayes. Springer US, 2004, pp. 3–11. ISBN: 978-0-306-47294-7.

[60] M. Edwards, S. Huet, F. Goreaud, and G. Deffuant. "Comparing an individual-based model of behaviour diffusion with its mean field aggregate approximation". *Journal of Artificial Societies and Social Simulation* 6.4 (2003).

[61] D. Hidović and J. E. Rowe. "Validating a model of colon colouration using an evolution strategy with adaptive approximations". *Genetic and Evolutionary Computation Conference*. Springer. Seattle, WA, USA, 2004, pp. 1005–1016.

[62] K. Eggensperger, M. Feurer, F. Hutter, J. Bergstra, J. Snoek, H. Hoos, and K. Leyton-Brown. "Towards an Empirical Foundation for Assessing Bayesian Optimization of Hyperparameters". *NIPS workshop on Bayesian Optimization in Theory and Practice*. Lake Tahoe, Sierra Nevada, USA, 2013.

[63] J. Snoek, H. Larochelle, and R. P. Adams. "Practical Bayesian optimization of machine learning algorithms". *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger. Lake Tahoe, Sierra Nevada, USA: Curran Associates, Inc., 2012, pp. 2951–2959.

[64] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter. "Efficient and Robust Automated Machine Learning". *Advances in Neural Information Processing Systems 28*. Ed. by C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett. Curran Associates, Inc., 2015, pp. 2962–2970.

[65] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. "Active learning with statistical models". *Journal of artificial intelligence research* 4 (1996), pp. 129–145.

[66] R. M. Castro. "Active learning and adaptive sampling for non-parametric inference". PhD thesis. University of Wisconsin at Madison, 2007.

[67] B. Settles. "Active learning literature survey". *University of Wisconsin, Madison* 52.55-66 (2010), p. 11.

[68] B. Settles. *Active learning*. Vol. 6. 1. Morgan & Claypool Publishers, 2012, pp. 1–114.

[69] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. *Tensorflow: Large-scale machine learning on heterogeneous distributed systems*. Tech. rep. 2015.

[70] H. Frohlich and A. Zell. "Efficient parameter selection for support vector machines in classification and regression via model-based global optimization". *IEEE International Joint Conference on Neural Networks, IJCNN'05*. Vol. 3. IEEE. Montreal, QC, Canada: Institute of Electrical and Electronics Engineers, Inc, 2005, pp. 1431–1436.

[71] A. Klein, K. Eggensperger, M. Feurer, and S. Falkner. *HPOlib2: hyperparameter optimization and black box optimization benchmarks*. http://github.com/automl/HPOlib2. 2016.

[72] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. "The WEKA Data Mining Software: An Update". *SIGKDD Explorations* 11.1 (2009).

[73] I. Couckuyt, T. Dhaene, and P. Demeester. "ooDACE Toolbox: A Flexible Object-Oriented Kriging Implementation". *Journal of Machine Learning Research* 15 (2014), pp. 3183–3186.

[74] B. E. Boser, I. M. Guyon, and V. N. Vapnik. "A training algorithm for optimal margin classifiers". *Proceedings of the fifth annual workshop on Computational learning theory*. ACM. ACM, 1992, pp. 144–152.

[75] C.-C. Chang and C.-J. Lin. "LIBSVM: a library for support vector machines". *ACM Transactions on Intelligent Systems and Technology (TIST)* 2.3 (2011), p. 27.

[76] J. C. Platt. "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods". *Advances in large margin classifiers*. Ed. by J. Smola, P. Barlett, B. Scholköpf, and D. Schuurmans. Citeseer. 1999.

[77] G.-B. Huang, D. H. Wang, and Y. Lan. "Extreme learning machines: a survey". *International Journal of Machine Learning and Cybernetics* 2.2 (2011), pp. 107–122.

[78] J. A. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, J. Suykens, and T. Van Gestel. *Least squares support vector machines*. Vol. 4. World Scientific, 2002.

[79] L. Breiman. "Random forests". *Machine learning* 45.1 (2001), pp. 5–32.

[80] J. Kennedy and R. Eberhart. "Particle swarm optimization". *Proceedings of the IEEE International Conference on Neural Networks*. IEEE. Perth, WA, Australia: Institute of Electrical and Electronics Engineers, Inc, 1995.

[81] D. R. Jones, M. Schonlau, and W. J. Welch. "Efficient Global Optimization of Expensive Black-Box Functions". *J. of Global Optimization* 13.4 (1998), pp. 455–492. ISSN: 0925-5001.

[82] K. Crombecq, I. Couckuyt, D. Gorissen, and T. Dhaene. "Space-filling Sequential Design Strategies for Adaptive Surrogate Modelling". *Proceedings of the First International Conference on Soft Computing Technology in Civil, Structural and Environmental Engineerin*. Ed. by B. H. Topping and Y. Tsompanakis. Vol. 92. Stirlingshire, UK: Civil-Comp Press, 2009, Paper–50.

[83] K. Crombecq, E. Laermans, and T. Dhaene. "Efficient space-filling and non-collapsing sequential design strategies for simulation-based modeling". *European Journal of Operational Research* 214.3 (2011), pp. 683–696.

[84] K. Crombecq, D. Gorissen, D. Deschrijver, and T. Dhaene. "A Novel Hybrid Sequential Design Strategy for Global Surrogate Modelling of Computer Experiments". *SIAM Journal of Scientific Computing* 33.4 (2010), pp. 1948–1974.

"For all these distributions [...], the moment generating functions are precomputed, so you don't have to do anything. You just plug that into that equation... and this also means it has a mean and variance."

— Kilian Q. Weinberger

3

Surrogate Modelling with Classifiers and Sequential Design

J. van der Herten, I. Couckuyt, D. Deschrijver, and T. Dhaene.
***Advances in Engineering Software* 99 (2016), pp. 137–146**

3.1 Introduction

Supervised learning algorithms learn the relation between an input space and a corresponding output space based on multiple examples (*samples*). After learning, the predictor can be used to predict the output(s) of unseen data points. In case an output varies continuously, this task is referred to as *regression*. When only a distinct number of discrete outcomes are possible (*labels*), the term *classification* is used. In literature, classification algorithms usually label large data sets. To limit the massive computational requirements of the learning process, the data is often sub-sampled to obtain a smaller representative set of training data.

Sometimes, obtaining the label for a sample is a very *expensive* task: it might be the result of a lengthy computer simulation or a (possibly dangerous) real-life experiment. Assuming there are *budget constraints* limiting the total amount of labels that can be acquired, obtaining the labels for all samples in the data set might not be possible. Although budget constraints also include applications where time and money is required for instance preparation [2], this work focuses on labels obtained through evaluation of complex physics-based (deterministic) simulators. These are used frequently in Computer Aided Design (CAD) and Computer Aided Engineering (CAE) to avoid building and testing several prototypes of new products. As these simulations have become significantly more accurate over the years, their computational requirements have also become more expensive.

A surrogate model is traditionally a cheap-to-evaluate mathematical regression model mimicking the response of computationally intensive simulators with continuous response range, and are trained from a small set of (sequentially) well-chosen evaluations. This chapter demonstrates the applicability of the state-of-the-art SUMO toolbox for surrogate modelling [3] using sequential design (as introduced in Section 2.9) for classification tasks. The concept of sequential design is discussed in detail in Section 3.2, and the sequential sampling step for classification is discussed in more detail in Section 3.3. The integrated SUMO platform is then illustrated on a number of use cases in Section 3.4.

3.2 Sequential design

Prior to applying classifiers in the context of surrogate modelling, first the concept of sequential design is discussed in more detail, as well as an overview of recent methods.

3.2.1 Methodology

Sequential design turns the one-shot approaches discussed in Section 2.5.1 into an iterative process [4, 5]. The acquired data and the constructed models from previous iterations are analysed in order to intelligently select locations for new data points (sequential sampling). Next, the labels for these additional data points are obtained and new models can be trained or existing models can be updated (in case online learning methods are used to update existing models with additional data [6]). First of all, this means there is no risk of over- or undersampling as the process can be halted when the desired accuracy is reached (or if the computational budget is exceeded). A second major advantage is that information provided by the consecutive labels and intermediate models can guide the selection to obtain

optimal locations for new data points. This allows the data distribution to be adapted and refined to the problem at hand as more knowledge becomes available, which means the sampling is no longer only guided by space-filling designs/methods. In surrogate modelling, the concept of sequential design has been applied in several successful applications [4, 7–11].

Experimental design with sequential sampling is related to the field of active learning [12–15]. Under its original formulation, active learning picks some data points from a set of unlabeled candidate points for evaluation, after which one or multiple classifiers are trained on the labeled instances. After a performance evaluation, the process may be repeated to label more training instances to improve the accuracy. Often, an active learning algorithm provides a ranking of possible data points [16, 17]. However, over the years active learning has grown into a broad field which now includes intelligent selection of data points in a continuous space (which implies no predefined set of unlabeled data points is available), and also allows continuous output ranges, rather than a discrete set of labels. Examples include an approach presented by [18] which sequentially learns a decision function, and optimal learning for infinitely many armed bandits problems [19], in which the learner can either sample an arm (or a distribution) that has been already been observed in the past, or sample a new arm with a known mean. There is a reward associated with each arm (each potential sample), and the objective is to maximise the reward and minimise the regret (expected difference between collected reward and the reward associated with the optimal arm). For an excellent overview and mathematical treatment of sequential sampling techniques in active learning, we refer to [20]. The notion of reward or regret is not considered in this work.

The typical modelling process with sequential design is illustrated in Figure 3.1: it is initiated by generating a small set of initial data points (referred to as *initial design*) which are simulated. The process then initiates a loop: a model is trained and its *hyperparameters* are optimized with respect to a pre-set quality criterion (discussed in Section 2.9). When improvement can no longer be realized and the quality of the model is not sufficient, the sequential sampling routine is started. Based on all available information one or possibly more new data points are chosen by this co-routine, of which the labels are acquired. When the labels are available a new model is trained and optimized. This process continues until the stopping criterion is satisfied: either the regression model or classifier is sufficiently accurate, or the budget constraints (maximum number of evaluations or a time limit) are reached.

The sampling and modelling steps of the process are independent (with the exception of modelling-based sampling strategies (Section 3.3), which means construction of intermediate models is not an absolute requirement (it is possible to immediately select new samples after evaluation as represented by the dashed line in Figure 3.1):

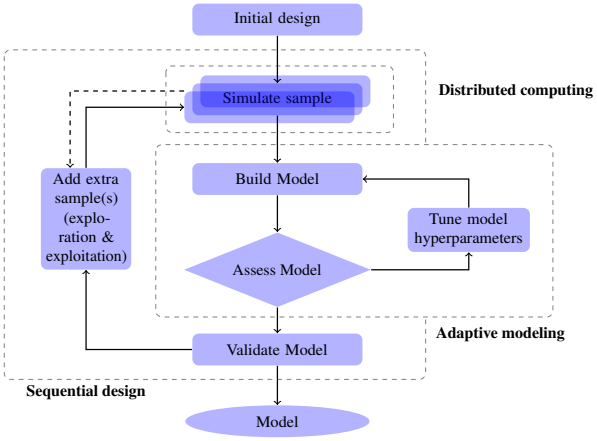


Figure 3.1: Sequential design: a typical flow. After evaluating an initial set of samples an (intermediate) model is created and tuned. A feedback loop allows for the evaluation of additional samples to improve the quality of the model. When the sample selection strategy is not model-based, sample selection may also proceed without training the intermediate models each iteration as represented by the dashed arrow.

possible scenarios include sequential selection and simulation of samples and construction of a model only when the computational budget is consumed, or selection of samples in batches as opposed to one-by-one.

Continuous outputs can be approximated with regression techniques such as Kriging [21], Artificial Neural Network (ANN), Radial Basis Function (RBF) models etc. Sequential sampling algorithms typically discover “difficult" regions in the design space and sample them densely as these regions tend to result in high model uncertainty. For regression applications sequential design usually focuses on regions of the design space that are undersampled and where additional samples are needed to discover the response behaviour (input-based exploration), or highly non-linear regions requiring additional information in order to be modeled accurately. The latter requires knowledge on the responses of earlier samples (output-based exploitation). When discrete outputs are encountered, techniques such as Random Forests (RF) [22], Support Vector Machine (SVM) [23–25] or Naive Bayes [26] are used to classify the data points in the adaptive modelling step. For these problems, the model uncertainty is usually situated in regions which have been undersampled (exploration), or near the classification boundaries (exploitation).

3.2.2 Existing sequential sampling methods

Fundamentally all sequential design approaches involve two competing objectives:

1. **Exploration**: sampling regions of the design space where proportionally only little information has been acquired.
2. **Exploitation**: sampling promising (w.r.t to the goal) regions of the design space.

These terms are widespread, and are also present in literature on active learning and reinforcement learning. It is clear, however, that a good strategy strikes a balance between these goals as both are required to obtain satisfactory results. A brief summary is given below, however a more in-depth discussion can be found in Chapter 4.

Exploration-based algorithms are typically less involved with the goal of the process and aim to reduce model uncertainty by covering the design space uniformly. They are crucial to assure that no relevant parts of the response surface are completely missed. Space-filling sequential experimental design usually relate to the same criteria used to construct one-shot designs and involves distance to neighbouring points, e.g., the maximin / minimax criteria, potentially complemented with projective properties [27]. Some explicitly expand a one-shot design such as nested Latin Hypercube Design (LHD), Monte Carlo (MC) or Quasi-Monte Carlo (Q-MC).

On the other hand, exploitation methods clearly pursue the goal of the process and often require an intermediate surrogate model(s) to be built for analysis (e.g., Expected Improvement (EI) requires Kriging or Gaussian Process (GP) models). In case a global accurate model is required, a very effective approach is raising the information density in regions with non-linear response behaviour such as LOLA-Voronoi [28] and FLOLA-Voronoi [29]. These approaches are an exception to this, as these approaches do not require intermediate surrogate models models (they operate on local linear interpolations). For optimisation purposes, the type of sampling depends on the task and the requirements. For single-objective optimisation examples of such sampling methods include Constrained Optimisation using Response Surfaces (CORS) [30] and Bayesian optimisation acquisition functions such as Expected Improvement [31] (combined with Kriging models this corresponds to the well-known Efficient Global Optimisation (EGO) approach [32]), the knowledge-gradient [33] and Predictive Entropy Search (PES) [34]. Many of these methods for optimisation can also be used in combination with a method which learns about the feasibility of input regions of the design space: during the iterative process an additional model learns the feasibility from the samples (as reflected

by the simulation thereof). This information is then used during the selection of new samples with specific criteria such as the Probability of Feasibility (PoF) [35, 36]. Surrogate-Based Optimisation (SBO) has also been extended to problems with two or more (potentially conflicting) objectives. The goal of this type of multi-objective optimisation is the identification of a Pareto front of solutions which presents the trade-off between those objectives. Pioneer work is the well-known approach known as ParEGO [37], which was later improved using different interpretations of improvement in multi-objective setting: this includes hypervolume-based methods such as the Hypervolume Probability of Improvement (HvPoI) [38, 39], the Hypervolume Expected Improvement (HvEI) [40] or the recent multi-objective generalisation of PES [41].

Some methods target model uncertainty explicitly. This type of methods, however, creates a dependency between the sampling and modelling steps (and often comes at a non-negligible extra computational cost). Bayesian model types represent the former type of models, e.g., the prediction variance of Kriging and Gaussian Process models, which can be applied directly for maximum variance sampling [42, 43] or maximum entropy designs [44]. For these kind of models a better way is expressing the uncertainty on the model hyperparameters resulting in approaches to reduce this uncertainty and hence, enhancing the overall model confidence [45]. Model uncertainty can also be inferred by training several models and compare their responses. Areas with most disagreements are then marked for additional samples (*query by committee* methods [46–48]). This can be a very effective approach in combination with ensemble modelling.

An overview of some popular adaptive sampling methods for modelling with sequential design, for both regression and classification outputs is given in Table 3.3, together with some of their relevant properties and goals. Methods are categorised as input-, output- and model-based. Roughly, input methods only consider the geometry within the input space, output-based methods analyse the observed responses whereas model-based approaches use the intermediate surrogate model(s). Note that this distinction is not very strict as the latter type of methods imply the existence of observations in order to train models, whereas that output-based methods usually build a simple model representation for their analysis. In addition, some methods are a combination of several algorithms, explaining several checks.

Method	Response type		Type				Purpose	References
	Regression	Classification	Input	Output	Model	Model types		
Low discrepancy sequences	✓	✓	✓				Space-filling	[49–51]
Sequentially Nested LHD	✓	✓	✓				Space-filling	[52–54]
Minimax/Maximin	✓	✓	✓				Space-filling	[55]
Fast Flexible Filling Designs	✓	✓	✓				Constrained space-filling	[56]
Voronoi-based	✓	✓	✓				Space-filling	[28]
Random sampling (MC)	✓	✓	✓				Any	
LOLA-Voronoi	✓		✓	✓			Non-stationarity	[28]
FLOLA-Voronoi	✓		✓	✓			Non-stationarity	[29]
Neighbourhood-Voronoi		✓	✓	✓			Boundary sampling	[57]
DHASD	✓		✓	✓	✓	Any	Non-stationarity	[58]
ASK	✓				✓	Trees	Non-stationarity	[59]
CV-Voronoi	✓	✓	✓		✓	Any	Space-filling/model uncertainty	[60]
Minimum Energy Design	✓		✓	✓			Non-stationarity	[61]

Table 3.1: Adaptive sampling algorithms: overview.

Method	Response type		Type				Remarks	References
	Regression	Classification	Input	Output	Model	Model types		
Model error sampling	✓				✓	Any	Reduce model error	[62]
EDSD		✓			✓	SVM	Boundary sampling	[63]
SEED	✓	✓			✓	Any	Reduce model disagreement	[46]
Global/Local disagreement	✓				✓	Any	Reduce model error	[64]
Maximum variance	✓	✓			✓	Bayesian	Reduce model uncertainty	[42, 43]
Maximum entropy design	✓				✓	GP	Reduce model uncertainty	[44]
BALD	✓				✓	GP	Learn hyperparameters	[45]
EI	✓				✓	Bayesian	Optimisation	[31, 32, 65]
q-EI	✓				✓	GP	Batch optimisation	[66–68]
Constrained EI	✓				✓	GP	Constrained optimisation	[36]
Knowledge-gradient	✓				✓	Bayesian	(Discrete) Optimisation	[33, 69, 70]
Parallel Knowledge-gradient	✓				✓	GP	Batch optimisation	[71]
PoF	✓	✓			✓	GP	Constraint satisfaction	[35]

Table 3.2: Adaptive sampling algorithms: overview.

Method	Response type		Type				Remarks	References
	Regression	Classification	Input	Output	Model	Model types		
PoI	✓				✓	GP,TP	Optimisation	
Generalized PoF	✓				✓	GP	Inverse modelling	[72, 73]
Copeland EI					✓	GP	Tournament-based optimisation	[74]
CORS	✓				✓	RBF	Optimisation	[30]
HvPoI	✓				✓	GP, TP	Multi-objective optimisation	[38, 39, 73]
HvEI	✓				✓	GP	Multi-objective optimisation	[40]
PES	✓				✓	GP	(Multi-objective) optimisation	[34, 41]
MRS	✓				✓	GP	(Multi-objective) optimisation	[75]
ALP	✓				✓	GP	Multi-objective optimisation	[76]
CBMOO	✓				✓	CART	Multi-objective optimisation	[77]
Estimating Probability of Faillure	✓				✓	GP	Estimate size excursion set	[78, 79]

Table 3.3: Adaptive sampling algorithms: overview.

3.3 Sequential sampling for classification

We now review some aspects of sequential design for classification problems approximating a black-box simulator with discrete outputs. Two sequential sampling approaches are described in more detail: Neighbourhood-Voronoi (N-V) [57] (an output-based approach) and Probability of Feasibility [35] (a model-based approach). These methods are also applied to the test cases in Section 3.4.

3.3.1 Neighbourhood-Voronoi

By default, the SUMO Toolbox offers the N-V algorithm [57, 80] for classification, a sequential sampling strategy combining exploration and exploitation for the construction of accurate classifiers. This algorithm is a modification of the LOLA-Voronoi [28] sequential sampling algorithm used in surrogate modelling. The N-V algorithm is based on the Voronoi tesselation of the search space and focuses on two distinct goals:

- Discover the class regions: the input space should be explored to find the (sub-)regions of the different classes. When nothing is known about the problem at hand, the choice of new data points should be influenced by the possible existence of undiscovered regions. As iterations evolve and all (possibly disconnected) regions of all classes have at least one data sample the *exploration* can be halted. Depending on the problem, this knowledge may be available or not.
- Refine the boundaries: when two or more distinct regions have been identified, new data points should be chosen such that the location of the boundary between the regions can be identified. This *exploitation* component greatly enhances the accuracy of the classifier.

For each data point $\boldsymbol{x}_r$ of a set of data samples X, the N-V algorithm first selects a set of indices to nearby points $N(\boldsymbol{x}_r)$ known as the *neighbourhood*. The choice of neighbouring points is guided by two principles: the cohesion (defined as the average distance of the points in $N(\boldsymbol{x}_r)$ and $\boldsymbol{x}_r$) and adhesion (the average minimum distance of points in $N(\boldsymbol{x}_r)$ from each other). A valuable neighbourhood has high cohesions (corresponding to a low average distance to $\boldsymbol{x}$)) and low adhesion (high average minimum distance amongst the points in $N(\boldsymbol{x}_r)$). Clearly, these two principles conflict as a higher cohesion implies higher adhesion as well. When the size of the neighbourhood equals twice the dimensionality of the data samples, the optimal configuration is known as the *cross-polytope*. A candidate neighbourhood is first assigned a score which indicates how much it resembles to a

cross-polytope:

$$R(N(\boldsymbol{x}_r)) = \frac{A(N(\boldsymbol{x}_r))}{\sqrt{2}C(N(\boldsymbol{x}_r))}.$$

To obtain the neighbourhood score which is used to guide the search amongst all possible neighbourhood candidates, R is divided by C to prefer neighbourhoods with low cohesion if two candidates are found which resemble the cross-polytope configuration equally:

$$S(N(\boldsymbol{x}_r)) = \frac{R(N(\boldsymbol{x}_r))}{C(N(\boldsymbol{x}_r))}.$$

Once the neighbourhood candidate with optimal S has been selected, the labels of the points in $N(\boldsymbol{x}_r)$ are compared: when no disagreement is found the Voronoi cell defined by $\boldsymbol{x}_r$ is considered to contain no class boundary. The size of all Voronoi cells is computed and serves as a basic score. If a disagreement is found in the labels of the points in $N(\boldsymbol{x}_r)$, the score is increased. New samples are then selected within the Voronoi cells with highest score: this could be because a disagreement was found and the cell is large compared to other cells with disagreements (*exploitation*) or because the cell became very large compared to all other cells and should be sampled, even if a disagreement has not yet been found (*exploration*). A full description of the Neighbourhood-Vornoi algorithm can be found in [57].

The N-V algorithm is an excellent choice for sequential selection of data points in the context of black-box classification problems. The method can define all data points upfront, independently of classifiers to be trained in a later step: N-V does not query the classifier for regions of uncertainty. The benefit of having a sampling strategy independent from the intermediate classifiers is significant when only a small number of data points have already been evaluated: at this point the classifier is still unstable because it lacks information which might influence the sample selection undesirably. Furthermore, the N-V algorithm automatically balances exploration and exploitation which allows discovery of previously undiscovered class regions. The latter property distinguishes N-V from other methods such as Explicit Design Space Decomposition (EDSD) [63], which assumes the initial set of points covers all regions. A downside of the Neighbourhood-Voronoi algorithm is its increasing computational complexity as the dimensionality of the input space grows (similar to the problems encountered with the LOLA-Voronoi algorithm). However, this issue could be tackled by applying a faster method to select $N(\boldsymbol{x}_r)$. These complexity issues as well as faster neighbourhood selection is further explored in Chapter 4, together with a more in depth description of the (F)LOLA-Voronoi algorithms.

3.3.2 Probability of feasibility

A model-based method for sequential design is the PoF [35]. This criterion picks new data points in underexplored areas which have a high probability of remaining below a certain threshold $f_{\min}$, exploiting information of the predictive distribution of a Bayesian model.

Definition 3 (Probability of Feasibility)**.** *Given the model belief represented by random variable $\tilde{F} \sim \mathcal{N}\left(\mu(\boldsymbol{x}), s^2(\boldsymbol{x})\right)$, with μ and s^2 representing the mean and variance of the predictive distribution. For GP models these quantities were specified in Equations (2.6a) and (2.6b) but they can also be computed for other models such as Kriging (see Section 5.2.1). For a predefined threshold $f_{\min}$, the PoF criterion is defined as*

$$\begin{aligned}\alpha_n^{PoF}(\boldsymbol{x}) &= p\left(\tilde{F} < f_{\min}\right) \\ &= \Phi\left(\frac{f_{\min} - \mu(\boldsymbol{x})}{s^2(\boldsymbol{x})}\right).\end{aligned}$$

As more data becomes available, the underlying model becomes more certain and the PoF becomes an accurate representation of the actual constraint. This knowledge is then typically combined with other criteria to guide constrained expensive black-box optimisation problems.

For classification problems, the PoF can be interpreted as the probability estimate of a probabilistic classifier, extending the compatibility of the PoF to probabilistic classification models such as the probabilistic SVM. This approach is very suitable for modelling constraints when the output of the constraints is discrete (feasible/infeasible).

3.4 Test cases

3.4.1 Spiral

As a first illustration, the two spiral example [81], a widely used binary classification benchmark, is modeled with sequential sampling. A small Latin Hypercube of 10 points generated by the translational propagation algorithm [82] was used as initial design. This method is able to generate near-optimal LHDs requiring only a fraction of the computational cost required for optimizing LHDs using traditional optimization techniques. Each iteration, a SVM classifier with RBF kernel is trained and its hyperparameters (kernel bandwidth and regularization constant) are optimized using a multi-objective quality criterion computed over a validation set

specifically chosen to evaluate the correctness of the class boundaries as shown in Figure 3.2. Next, an additional sample is added to the design using the N-V algorithm.

Although the amount of the input space assigned to one of the classes is approximately 50%, during the early stages of the process when little data is available the sequential selection of data points may cause class imbalance. Several options exist to counter class imbalance problems such as the F_1 score or geometric average of precision and recall (as used in the following test cases). Here, we make use of the multi-objective hyperparameter optimization features of the SUMO Toolbox to optimize the position of the classifiers in the Receiver Operating Characteristic (ROC) space. Let A denote the set of data points with label A, and $\mathrm{Pr}(A)$ the set of data points labeled A by a classifier. We define the set of *true positives* as $\mathrm{TP}(A) = A \cap \mathrm{Pr}(A)$, and the set of *false positives* as $\mathrm{FP}(A) = \mathrm{Pr}(A) \setminus A$. Dividing TP and FP by the total amount of data points labeled A or not A respectively, we obtain the True Positive Rate (TPR) and False Positive Rate (FPR). As we are essentially confronted with two objectives, we apply the NSGA-II [83] and optimize the hyperparameters to improve the trade-off of both objectives.

Figure 3.3 shows the ROC space for the final population of the genetic algorithm for certain iterations. As the number of samples grows, the quality of the classifiers increases. The modelling run terminates when 200 samples have been evaluated: at this point the population consists of approximately ideal classifiers (true positive rate of 1 and false positive rate of 0, with respect to the validation set).

3.4.2 Stanford bunny

In this illustration, a classifier is trained for the Stanford Bunny 3D model [84] consisting of 69451 polygons. The input space is three-dimensional (x, y, z coordinates) and the output is binary: a zero indicates the point is outside of the model, a one indicates the point is inside. The resulting class boundary is the contour of the object. In fact, checking if a point is inside or outside of an object is not a very computationally complex task: in this chapter it is only used to illustrate the capabilities of the sequential approach, as well as the toolbox.

The toolbox was configured with an initial LHD generated by the Translational Propagation algorithm [82] ($N_{\text{init}} = 50$). For input dimensionality lower than 6, this algorithm was proven to be generating the optimal LHD with high probability. Each iteration of the sequential design, 10 additional points were selected by the N-V algorithm. The process was terminated when 1000 samples were evaluated. Given the shape of the 3D object, this is quite a sparse data set (the size corresponds

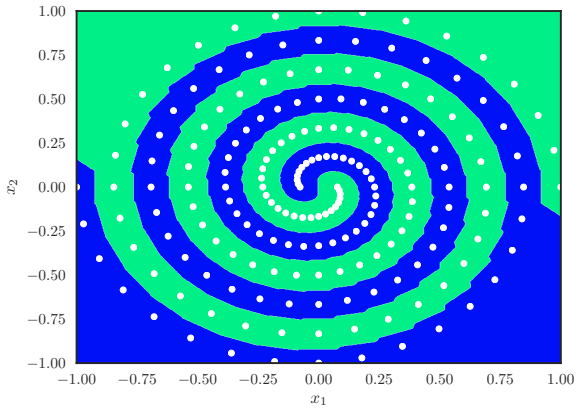


Figure 3.2: Spiral: illustration of the class boundaries (color) and the validation set (dots) for evaluation of the model quality.

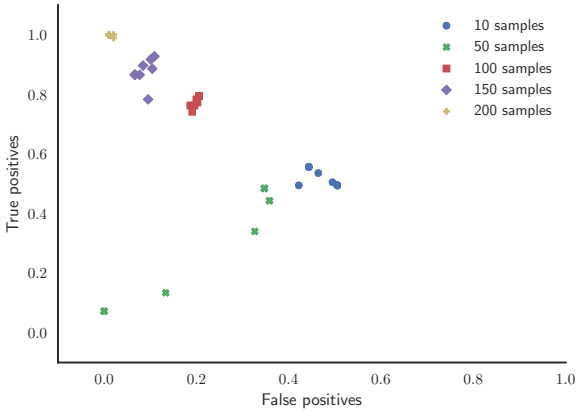


Figure 3.3: Spiral: final population of the genetic algorithm situated in the ROC space. As the number of samples increases, the fitness of the classifiers approaches towards the optimal result.

to a $10 \times 10 \times 10$ grid). The growth and evolution of the dataset[1] is illustrated in Figure 3.4.

For each iteration, several classifier types (SVM, ANN, RF and Naive Bayes) were trained concurrently in several threads to evaluate the performance of each classifier for this application. For the SVM, the Dividing Rectangles (DIRECT) algorithm [85] was used to optimize the kernel parameter (Radial Basis Function kernel) and the regularisation parameter. For ANN, a Genetic Algorithm (10 generations of 15 individuals) was used to optimize the network architecture and initial weights. Each individual network was trained with Levenberg-Marquard backpropagation with Bayesian regularisation (300 epochs) [86]. Random Forest (fixed number of 500 trees) and Naive Bayes had no parameters to be optimized.

For hyperparameter optimisation, 5-fold cross-validation was used as performance measure. In addition, the classifiers were also validated on a dense validation set to estimate their true error. A common problem specific to quality estimation of classifiers is caused by *class imbalance*. If a class is underrepresented, a straightforward error function such as the misclassification rate will favour predicting the majority class and hence pushes the hyperparameter optimization to classifiers discriminating the minority class. In extreme cases, the minority classes will be completely ignored. To avoid this, the geometric average of the *precision* and *recall* of both classes (out, in), represented by $p_{\text{out}}, p_{\text{in}}$ and $r_{\text{out}}, r_{\text{in}}$ respectively, is used as error function for this experiment:

$$G = \sqrt[4]{p_{\text{out}} p_{\text{in}} r_{\text{out}} r_{\text{in}}}.$$

The recall of the classifier for the class A is the ratio of true positives and the number of data points with label A:

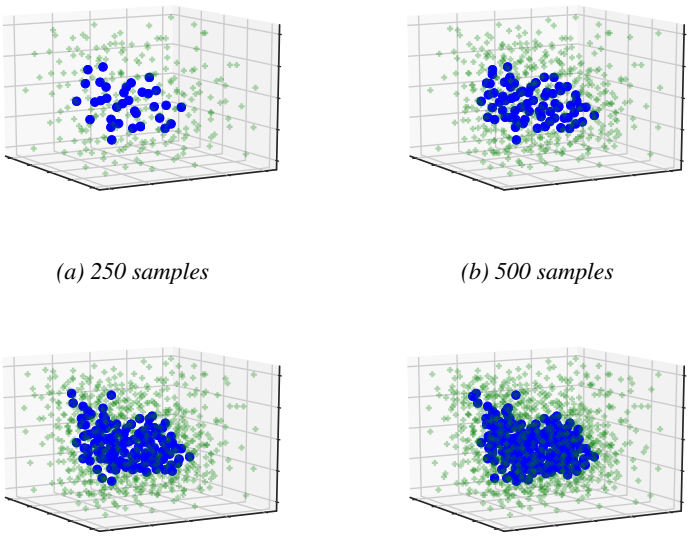
$$p_A = \frac{|\text{TP}(A)|}{|A|},$$

whereas the precision of A is defined as the ratio of the true positives and the number of predicted cases of A:

$$r_A = \frac{|\text{TP}(A)|}{|\text{Pr}(A)|}.$$

A score of $G = 1$ represents a perfect classifier as it implies all precision and recall terms have a value of 1. This means the classifier labels all data points correctly, a score of $G = 0$ represents a misclassification of every data point [87]. The G-measure is related to the $F1$-score, the latter is similar but is based on a harmonic average instead.

[1] A short movie of the sequential selection of data points can be seen on https://www.youtube.com/watch?v=EcvfbaSUMOw



(a) 250 samples *(b) 500 samples*

(c) 750 samples *(d) 1000 samples*

Figure 3.4: Stanford bunny: evolution of the experimental design used to train the classifiers, as constructed by the N-V algorithm. Blue dots are inside the 3D object, green crosses are outside.

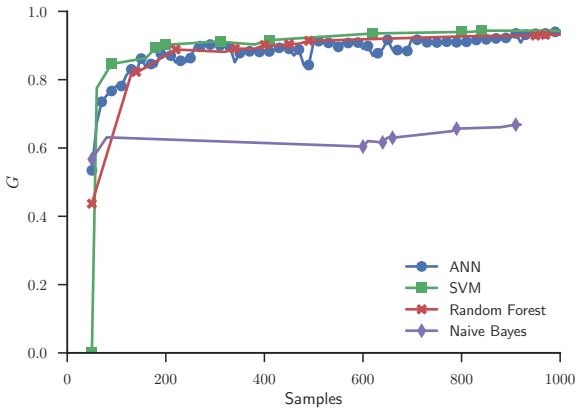


Figure 3.5: Stanford Bunny: evolution of the geometric mean of the precision and recall of both classes on the validation set for all classifier types as more samples are evaluated (up to 1000).

(a) *(b)* *(c)*

Figure 3.6: Stanford bunny: a grid of 10^7 points was classified by the final SVM model of the Bunny based on 1000 data samples. Iso-surface techniques were use to plot a volume using the resulting labels. Clearly, the SVM manages to fit the contour of the model very accurately.

Figure 3.5 shows the obtained G-score on the validation set as more samples are evaluated, and classifiers are retrained. The classifier accuracy improves as the number of samples increases for all methods included in this illustration: RF, SVM and ANN are performing very similar, but SVM always seem to be slightly better. The results for the ANN show most fluctuation: closer inspection reveals the optimisation of the network architecture sometimes gets stuck in a solution which scores well for cross-validation, but performs worse on the validation set: when new data points are added, the cross-validation score drops and the network architecture needs to be altered. This causes the bumpy behaviour of the ANN performance. Naive Bayes clearly is not suited to model the boundary of the 3D model: its score G score is stuck around 0.6 and is barely increasing as additional data points are added. Of all methods it performs worst.

Considering $G = 0.9$ corresponds to a very satisfying classifier for this application, SVM obtains the score after 200 evaluated samples. In comparison, an SVM trained on a one-shot maximin LHD of 200 points generated by the Translational Propagation algorithm [82] obtains a score of only $G = 0.85$. The final best SVM Model was evaluated on a dense grid and the obtained labels were used to generate an iso-surface of the Stanford Bunny which is shown in Figure 3.6.

3.4.3 Bended microstrip

This section describes the use of adaptive classification in the field of Electromagnetic Compatibility (EMC) [57]: a Near-Field (NF) pattern of a double bended microstrip line, measured using a scanning system as illustrated in Figure 3.7. The Printed Circuit Board (PCB) comprises a microstrip on a substrate. The microstrip was excited with a generator set and the amplitude of a field component, e.g. $|H_y|$, was measured with a NF scanner. The head of the scanner can be moved automatically in two dimensions at a fixed height of 2mm above the Device Under

Table 3.4: Bended microstrip: partitioning of the NF Range in three different classes.

Class Label	NF Range (dB μV)	Color
Low	[0 - 30]	Blue
Elevated	[30 - 35]	Green
High	[35 - inf[	Red

Test (DUT) to perform the measurements.

The NF pattern is a continuous output. However, we would like to identify radiation hotspots, regions with elevated radiation, and areas with low radiation near the board. Table 3.4 indicates how the output range was mapped onto these three labels. A small LHD of $N_{\text{init}} = 30$ generated by the Translational Propagation algorithm [82] was used as initial design. The input space consists of the (x, y)-coordinate on the PCB. Each iteration the N-V algorithm selects a new sample. After evaluation, an SVM classifier (RBF kernel) with two parameters (kernel bandwidth and regularisation parameter) optimized by the DIRECT algorithm [85] was trained on the dataset, the performance of the classifiers was estimated by cross-validation. A simple error function such as the miss-classification rate results in a premature end of the process due to class imbalance. To counter this issue, the geometric average of precision and recall for all three classes was used.

When 264 measurements were evaluated, the desired accuracy of $G = 0.90$ (a score of 1 represents a perfect classifier with perfect precision and recall for all classes) was obtained and the process was halted. Figure 3.8 shows a plot of the distribution of labels of the final classifier, and all measurements as chosen by the sequential design strategy. A strong focus is on the region containing the hotspot: it is surrounded by a thin region with elevated radiation which requires high sampling density to obtain sufficient information on the class boundaries. This concentration effect did not cause the central region to be oversampled. The exploration part of N-V has explored the design space to avoid missing out a class region: if any region was missed it is no larger than the size of the largest Voronoi cell.

3.4.4 Cyclone optimisation

The adaptive classification strategy can also be used to model computationally expensive black-box constraints in optimisation problems. In this section a 7D constrained Computational Fluid Dynamics (CFD) design problem is studied. Multi-Objective Surrogate Based (Bayesian) Optimisation (MOSBO) [88] is used to find pareto-optimal solutions. Gas cyclones are widely used in air pollution control, gas-solid separation for aerosol sampling and industrial applications when large

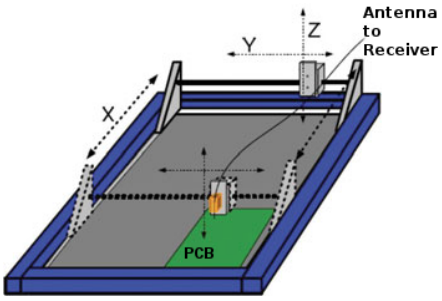


Figure 3.7: Bended microstrip: near-field scanner setup.

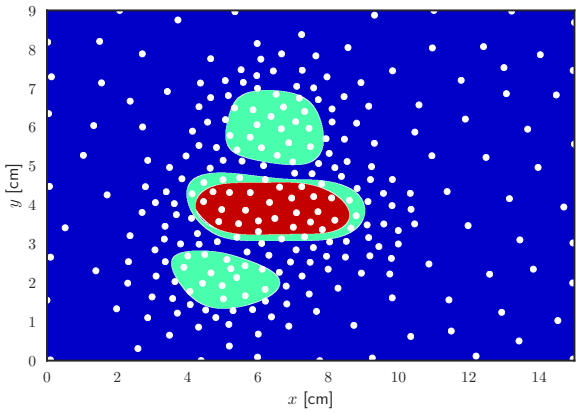


Figure 3.8: Bended microstrip: contour plot of the final SVM classifier based on 264 measurements (dots). The colors of the classes correspond to the last column in Table 3.4. The focus of the sampling algorithm is on the class boundaries near the hotspot region.

particles are to be caught. In cyclone separators, a strongly swirling turbulent flow is used to separate phases with different densities. A tangential inlet generates a complex swirling motion of the gas stream, which forces particles toward the outer wall where they spiral in the downward direction. Eventually, the particles are collected in the dustbin (or flow out through a dipleg) located at the bottom of the conical section of the cyclone body. The cleaned gas leaves through the exit pipe at the top. The cyclone geometry [89] is described by seven geometrical parameters: the inlet height a, width b, the vortex finder diameter D_x, and length S, cylinder height h, cyclone total height H_t and cone-tip diameter B_c. Modifying these parameters has an impact on the gas cyclone itself. Two aspects of the cyclone must be optimized: the pressure loss (represented by the Euler number) and the cut-off diameter. The latter is represented by the Stokes number

$$\text{Stk} = \frac{t_r\, u}{l},$$

with t_r representing the particle relaxation time, u the velocity of the fluid away from the obstacle, and l the diameter of the obstacle. The particle relaxation time corresponds to the time constant of the exponential decay of its velocity due to drag.

In addition to both objectives, evaluating the simulator also yields four binary values representing black-box constraints on the geometry (denoted as f_{c1}, f_{c2}, f_{c3} and f_{c4}). Each constraint corresponds to internal checks regarding the feasibility of the configuration specified by the user. As each evaluation is computationally demanding this additional knowledge should be included in order to maximise the probability of selecting feasible solutions. Therefore, the constraints should be modeled and included in the optimisation process. As the output of the constraints in this example is discrete (feasible/infeasible), we could map both classes to a number (0/1) and apply regression and PoF. However, this would essentially be a non-stationary problem (the smoothness of the response surface varies greatly at the boundary) which can lead to problems with GP and Kriging models [90]. Instead of using a probabilistic classification algorithm, we can model the discrete constraint responses and still use the PoF criterion.

To handle this complex 7D multi-objective constrained design problem, the SUMO toolbox is configured to model the Euler and Stokes objectives with Least-Squares Support Vector Machine (LS-SVM) [91]. The hyperparameters (RBF kernel bandwidth and regularisation parameter) are optimized with the DIRECT algorithm [85]. The sequential design strategy is a combination of two criteria: the HvPoI [88], a sequential sampling criterion for regression to guide the multi-objective optimisation and the PoF [35] to guide the optimisation towards feasible regions. The combined criterion becomes

$$\alpha(\boldsymbol{x}) = \alpha_n^{\text{HvPoI}}\left(\boldsymbol{x} \,\middle|\, \tilde{f}_{\text{Euler}}, \tilde{f}_{\text{StK}}\right) \prod_{c\in\{\,c1,c2,c3,c4\,\}} \alpha_n^{\text{PoF}}\left(\boldsymbol{x} \,\middle|\, \tilde{f}_{\text{c}}\right). \tag{3.1}$$

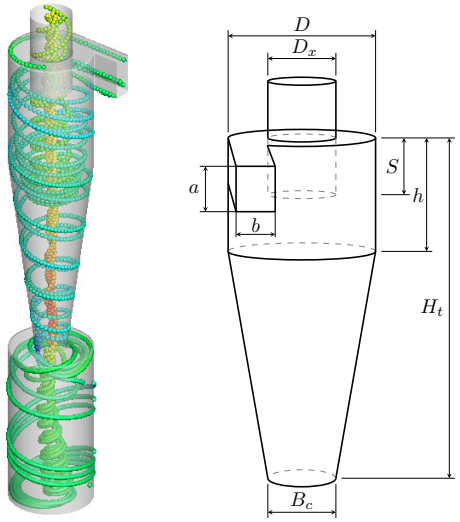
The next data point for evaluation is selected by optimizing α numerically. To compute the PoF, each constraint is modeled with a probabilistic SVM (RBF Kernel) optimized with the DIRECT algorithm [85]. The quality of the constraint models is assessed by cross-validation, with the F_1-score of the positive class used as error function. The constraints are modeled using the same samples used for training of the surrogate model for the optimisation: as the process evolves, the optimisation learns the feasibility of the current samples. Inevitably, some samples that violate the constraints will be evaluated while the process evolves. The initial design is a LHD of $N_{\text{init}} = 50$ generated by the Translational Propagation algorithm [82]. Each iteration 5 samples are selected by the sequential design strategy until the sample budget is consumed ($N_{\text{max}} = 120$).

Figure 3.10 shows the scores for all evaluated samples for both objectives. The red and green samples form the Pareto front. As the constraints were black-box and were learned throughout the process, many samples have been evaluated that do not satisfy the constraints (as these were not known at that time): only 8% of all 120 samples satisfy the constraints. Fortunately, 4 of them are Pareto optimal and represent valid optimal configurations. The exact optimal Pareto front is unknown, however in order to provide a comparison NSGA-II [83] was applied directly on the CFD simulations for a total of 10000 evaluations: the results are shown in Figure 3.10. It is clear that the Pareto optimal solutions found by our approach form a similar front to the front found by NSGA-II. However, our approach was able to identify these solutions with significantly fewer evaluations. Hence, the Pareto front of Figure 3.10 is a very good approximation given the budget constraint of 120 evaluations.

3.5 Conclusion

The SUMO Toolbox, a state-of-the-art MATLAB Toolbox developed for Surrogate Modelling with Sequential Design has been successfully extended to support adaptive training of classifiers, next to its wide variety of regression models. This chapter illustrates how the SUMO toolbox can be applied to efficiently solve computational expensive design applications involving classification and optimisation problems.

By default, the toolbox uses the sequential design methodology. We discussed the applicability to classification problems with labels resulting from expensive computer experiments. Sequentially, new data samples can be selected to improve



(a) Swirl simulation *(b) Geometry parameters*

Figure 3.9: Cyclone: illustration of a cyclone separator.

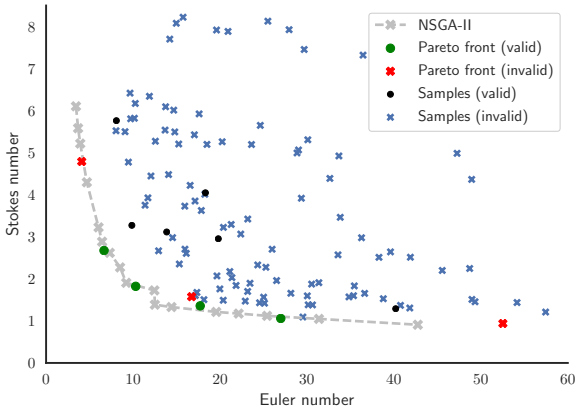


Figure 3.10: Cyclone: scores for all 120 evaluated samples for the multi-objective cyclone optimisation problem. Pareto front points that satisfy the constraints are shown in green, red crosses are Pareto optimal points that do not satisfy the constraints. Black-points are not Pareto optimal, but satisfy the constraints whereas blue crosses are invalid. For comparison, we included the Pareto front obtained by applying NSGA-II on the simulator for 10000 evaluations.

the accuracy of the classifier. These new samples are chosen based on what is already known about the application at that point (intermediate classifier, obtained labels, space-fillingness,...).

Improving the sequential sampling algorithms for classification problems (including incorporating existing methodologies from active learning) is subject of further work. In this chapter, N-V and PoF are two strategies used for sequential sampling of the class boundaries, both are available in the SUMO Toolbox. We highlighted the benefits of the N-V approach, but depending on the classifier and the problem at hand (constraints in optimisation, global accurate classifier,...) more optimal strategies can be developed.

3.6 References

[1] J. van der Herten, I. Couckuyt, D. Deschrijver, and T. Dhaene. "Adaptive classification under computational budget constraints using sequential data gathering". *Advances in Engineering Software* 99 (2016), pp. 137–146.

[2] F. J. Ruiz, N. Agell, and C. Angulo. "SVM-based learning method for improving colour adjustment in automotive basecoat manufacturing". *ESANN 2009, 17th European Symposium on Artificial Neural Networks*. Bruges, Belgium, Apr. 2009.

[3] D. Gorissen. "Grid-enabled adaptive surrogate modeling for computer aided engineering". PhD thesis. Ghent University, 2010, XIV, 362 [in multiple pagination]. ISBN: 9789085783503.

[4] D. Deschrijver, K. Crombecq, H. M. Nguyen, and T. Dhaene. "Adaptive Sampling Algorithm for Macromodeling of Parameterized-Parameter Responses". *Microwave Theory and Techniques, IEEE Transactions on* 59.1 (2011), pp. 39–45.

[5] D. Gorissen, K. Crombecq, I. Couckuyt, P. Demeester, and T. Dhaene. "A Surrogate Modeling and Adaptive Sampling Toolbox for Computer Based Design". *Journal of Machine Learning Research* 11 (2010). Available at http://sumo.intec.ugent.be, pp. 2051–2055.

[6] T. Anderson. *The Theory and Practice of Online Learning*. 2nd. Canada: Athabasca University Press, 2008. ISBN: 978-1-897425-08-4.

[7] J. Aernouts, I. Couckuyt, K. Crombecq, and J. J. Dirckx. "Elastic characterization of membranes with a complex shape using point indentation measurements and inverse modelling". *International Journal of Engineering Science* 48.6 (2010), pp. 599–611.

[8] D. Stephens, D. Gorissen, K. Crombecq, and T. Dhaene. "Surrogate based sensitivity analysis of process equipment". *Applied Mathematical Modelling* 35.4 (2011), pp. 1676–1687.

[9] S. Koziel and L. Leifsson. *Surrogate-Based Modeling and Optimization: Applications in Engineering*. SpringerLink : Bücher. Springer New York, 2013. ISBN: 9781461475514.

[10] D. Deschrijver, F. Vanhee, D. Pissoort, and T. Dhaene. "Automated near-field scanning algorithm for the EMC analysis of electronic devices". *Electromagnetic Compatibility, IEEE Transactions on* 54.3 (2012), pp. 502–510.

[11] S. Aerts, D. Deschrijver, W. Joseph, L. Verloock, F. Goeminne, L. Martens, and T. Dhaene. "Exposure assessment of mobile phone base station radiation in an outdoor environment using sequential surrogate modeling". *Bioelectromagnetics* (2013).

[12] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. "Active learning with statistical models". *Journal of artificial intelligence research* 4 (1996), pp. 129–145.

[13] R. M. Castro. "Active learning and adaptive sampling for non-parametric inference". PhD thesis. University of Wisconsin at Madison, 2007.

[14] B. Settles. "Active learning literature survey". *University of Wisconsin, Madison* 52.55-66 (2010), p. 11.

[15] B. Settles. *Active learning*. Vol. 6. 1. Morgan & Claypool Publishers, 2012, pp. 1–114.

[16] N. Ailon. "Active Learning Ranking from Pairwise Preferences with Almost Optimal Query Complexity". *Advances in Neural Information Processing Systems 24*. Ed. by J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger. Curran Associates, Inc., 2011, pp. 810–818.

[17] N. Ailon. "An active learning algorithm for ranking from pairwise preferences with an almost optimal query complexity". *Journal of Machine Learning Research* 13.Jan (2012), pp. 137–164.

[18] K. Trapeznikov and V. Saligrama. "Supervised sequential classification under budget constraints". *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*. 2013, pp. 581–589.

[19] A. Carpentier and M. Valko. "Simple regret for infinitely many armed bandits." *Proceedings of the 32rd International Conference on Machine Learning (ICML-15)*. Ed. by F. R. Bach and D. M. Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, 2015, pp. 1133–1141.

[20] K. Jamieson. "The Analysis of Adaptive Data Collection Methods for Machine Learning". PhD thesis. UW-Madison, 2014.

[21] I. Couckuyt, T. Dhaene, and P. Demeester. "ooDACE Toolbox: A Flexible Object-Oriented Kriging Implementation". *Journal of Machine Learning Research* 15 (2014), pp. 3183–3186.

[22] L. Breiman. "Random forests". *Machine learning* 45.1 (2001), pp. 5–32.

[23] B. E. Boser, I. M. Guyon, and V. N. Vapnik. "A training algorithm for optimal margin classifiers". *Proceedings of the fifth annual workshop on Computational learning theory*. ACM. ACM, 1992, pp. 144–152.

[24] C.-C. Chang and C.-J. Lin. "LIBSVM: a library for support vector machines". *ACM Transactions on Intelligent Systems and Technology (TIST)* 2.3 (2011), p. 27.

[25] J. C. Platt. "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods". *Advances in large margin classifiers*. Ed. by J. Smola, P. Barlett, B. Scholköpf, and D. Schuurmans. Citeseer. 1999.

[26] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. 2nd ed. Pearson Education, 2003. ISBN: 0137903952.

[27] K. Crombecq, E. Laermans, and T. Dhaene. "Efficient space-filling and non-collapsing sequential design strategies for simulation-based modeling". *European Journal of Operational Research* 214.3 (2011), pp. 683–696.

[28] K. Crombecq, D. Gorissen, D. Deschrijver, and T. Dhaene. "A Novel Hybrid Sequential Design Strategy for Global Surrogate Modelling of Computer Experiments". *SIAM Journal of Scientific Computing* 33.4 (2010), pp. 1948–1974.

[29] J. van der Herten, I. Couckuyt, D. Deschrijver, and T. Dhaene. "A Fuzzy Hybrid Sequential Design Strategy for Global Surrogate Modeling of High-Dimensional Computer Experiments". *SIAM Journal on Scientific Computing* 37.2 (2015), A1020–A1039.

[30] R. G. Regis and C. A. Shoemaker. "Constrained global optimization of expensive black box functions using radial basis functions". *Journal of Global optimization* 31.1 (2005), pp. 153–171.

[31] J. Močkus. "On Bayesian methods for seeking the extremum". *Optimization Techniques IFIP Technical Conference*. Ed. by G. Marchuk. Springer. Novosibirsk: Springer Berlin Heidelberg, July 1975, pp. 400–404. ISBN: 978-3-540-37497-8.

[32] D. R. Jones, M. Schonlau, and W. J. Welch. "Efficient Global Optimization of Expensive Black-Box Functions". *J. of Global Optimization* 13.4 (1998), pp. 455–492. ISSN: 0925-5001.

[33] P. Frazier, W. Powell, and S. Dayanik. "The knowledge-gradient policy for correlated normal beliefs". *INFORMS journal on Computing* 21.4 (2009), pp. 599–613.

[34] J. M. Hernández-Lobato, M. W. Hoffman, and Z. Ghahramani. "Predictive entropy search for efficient global optimization of black-box functions". *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger. Montreal, Canada: Curran Associates, Inc., 2014, pp. 918–926.

[35] A. I. Forrester and A. J. Keane. "Recent advances in surrogate-based optimization". *Progress in Aerospace Sciences* 45.1 (2009), pp. 50–79.

[36] J. Gardner, M. Kusner, K. Q. Weinberger, J. Cunningham, and Z. Xu. "Bayesian Optimization with Inequality Constraints". *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. Ed. by T. Jebara and E. P. Xing. Proceedings of Machine Learning Research. Beijing, China: PMLR, 2014, pp. 937–945.

[37] J. Knowles. "ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems". *Evolutionary Computation, IEEE Transactions on* 10.1 (2006), pp. 50–66.

[38] M. T. M. Emmerich, K. C. Giannakoglou, and B. Naujoks. "Single- and multiobjective evolutionary optimization assisted by Gaussian random field metamodels." *Evolutionary Computation, IEEE Transactions on* 10.4 (2006), pp. 421–439. ISSN: 1089-778X.

[39] I. Couckuyt, D. Deschrijver, and T. Dhaene. "Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization". *Journal of Global Optimization* 60.3 (2014), pp. 575–594. ISSN: 0925-5001.

[40] M. T. M. Emmerich, A. H. Deutz, and J. W. Klinkenberg. "Hypervolume-based expected improvement: Monotonicity properties and exact computation". *Congress on Evolutionary Computation (CEC)*. Ed. by M. T. M. Emmerich and P. Hingston. IEEE. New Orleans, LA, USA: Institute of Electrical and Electronics Engineers, Inc., 2011, pp. 2147–2154. ISBN: 978-1-4244-7834-7.

[41] D. Hernández-Lobato, J. M. Hernández-Lobato, A. Shah, and R. P. Adams. "Predictive Entropy Search for Multi-objective Bayesian Optimization". *Proceedings of the 33rd International Conference on Machine Learning (ICML-16)*. Ed. by M.-F. Balcan and K. Q. Weinberger. Vol. 48. Proceedings of

Machine Learning Research. Manhattan, New York, USA: PMLR, 2016, pp. 1492–1501.

[42] J. P. Kleijnen and W. C. Van Beers. "Application-driven sequential designs for simulation experiments: Kriging metamodelling". *Journal of the Operational Research Society* 55.8 (2004), pp. 876–883.

[43] M. J. Sasena. "Flexibility and efficiency enhancements for constrained global design optimization with kriging approximations". PhD thesis. General Motors, 2002.

[44] A. Farhang-Mehr and S. Azarm. "Bayesian meta-modelling of engineering design simulations: a sequential approach with adaptation to irregularities in the response behaviour". *International Journal for Numerical Methods in Engineering* 62.15 (2005), pp. 2104–2126.

[45] R. Garnett, M. A. Osborne, and P. Hennig. "Active learning of linear embeddings for Gaussian processes". *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence*. Ed. by Z. M.L. and T. J. Quebec, Canada: AUAI Press, 2014, pp. 230–239.

[46] Y. Lin. "An Efficient Robust Concept Exploration Method and Sequential Exploratory Experimental Design". PhD thesis. Georgia Institute of Technology, 2004.

[47] Y. Zhao, C. Xu, and Y. Cao. "Research on query-by-committee method of active learning and application". *Advanced Data Mining and Applications*. Springer, 2006, pp. 985–991.

[48] R. Burbidge, J. J. Rowland, and R. D. King. "Active learning for regression based on query by committee". *Proceedings of the 8th International Conference on Intelligent Data Engineering and Automated Learning*. IDEAL'07. Birmingham, UK: Springer-Verlag, 2007, pp. 209–218. ISBN: 978-3-540-77225-5.

[49] F. Hickernell. "A generalized discrepancy and quadrature error bound". *Mathematics of Computation of the American Mathematical Society* 67.221 (1998), pp. 299–322.

[50] R. Jin, W. Chen, and A. Sudjianto. "An efficient algorithm for constructing optimal design of computer experiments". *Journal of Statistical Planning and Inference* 134.1 (2005), pp. 268–287.

[51] H. Niederreiter. *Random Number Generation and quasi-Monte Carlo Methods*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1992. ISBN: 0-89871-295-5.

[52] B. G. M. Husslage et al. *Maximin designs for computer experiments*. Tech. rep. Tilburg University, 2006.

[53] P. Z. G. Qian. "Nested Latin hypercube designs". *Biometrika* 96.4 (2009), pp. 957–970.

[54] E. R. Van Dam, B. Husslage, D. Den Hertog, and H. Melissen. "Maximin Latin hypercube designs in two dimensions". *Operations Research* 55.1 (2007), pp. 158–169.

[55] M. E. Johnson, L. M. Moore, and D. Ylvisaker. "Minimax and maximin distance designs". *Journal of statistical planning and inference* 26.2 (1990), pp. 131–148.

[56] R. Lekivetz and B. Jones. "Fast Flexible Space-Filling Designs for Nonrectangular Regions". *Quality and Reliability Engineering International* 31.5 (2015), pp. 829–837.

[57] P. Singh, D. Deschrijver, D. Pissoort, and T. Dhaene. "Adaptive classification algorithm for EMC-compliance testing of electronic devices". *Electronics Letters* 49.24 (2013), pp. 1526–1528.

[58] A. Ajdari and H. Mahlooji. "An adaptive exploration-exploitation algorithm for constructing metamodels in random simulation using a novel sequential experimental design". *Communications in Statistics-Simulation and Computation* 43.5 (2014), pp. 947–968.

[59] P. Castro, E. Petit, A. Farjallah, and W. Jalby. "Adaptive sampling for performance characterization of application kernels". *Concurrency and Computation: Practice and Experience* 25.17 (2013), pp. 2345–2362.

[60] S. Xu, H. Liu, X. Wang, and X. Jiang. "A Robust Error-Pursuing Sequential Sampling Approach for Global Metamodeling Based on Voronoi Diagram and Cross Validation". *Journal of Mechanical Design* 136.7 (2014), p. 071009.

[61] V. R. Joseph, T. Dasgupta, R. Tuo, and C. J. Wu. "Sequential exploration of complex surfaces using minimum energy designs". *Technometrics* 57.1 (2015), pp. 64–74.

[62] W. Hendrickx and T. Dhaene. "Sequential design and rational metamodelling". *Proceedings of the 38th Conference on Winter Simulation*. Ed. by M. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines. Orlando, FL, USA: Institute of Electrical and Electronics Engineers, Inc, Dec. 2005, pp. 290–298.

[63] A. Basudhar, S. Missoum, and A. H. Sanchez. "Limit state function identification using support vector machines for discontinuous responses and disjoint failure domains". *Probabilistic Engineering Mechanics* 23.1 (2008), pp. 1–11.

[64] K. Boopathy and M. P. Rumpfkeil. "Unified framework for training point selection and error estimation for surrogate models". *AIAA Journal* 53.1 (2014), pp. 215–234.

[65] V. Picheny, T. Wagner, and D. Ginsbourger. "A benchmark of kriging-based infill criteria for noisy optimization". *Structural and Multidisciplinary Optimization* 48.3 (2013), pp. 607–626.

[66] C. Chevalier and D. Ginsbourger. "Fast computation of the multi-points expected improvement with applications in batch selection". *Revised Selected Papers of the 7th International Conference on Learning and Intelligent Optimization*. Catania, Italy: Springer-Verlag New York, Inc., 2013, pp. 59–69.

[67] S. Marmin, C. Chevalier, and D. Ginsbourger. "Differentiating the multipoint Expected Improvement for optimal batch design". *International Workshop on Machine Learning, Optimization and Big Data*. Springer. 2015, pp. 37–48.

[68] Z. Wang, F. Hutter, M. Zoghi, D. Matheson, and N. de Feitas. "Bayesian optimization in a billion dimensions via random embeddings". *Journal of Artificial Intelligence Research* 55 (2016), pp. 361–387.

[69] W. Scott, P. Frazier, and W. Powell. "The correlated knowledge gradient for simulation optimization of continuous parameters using gaussian process regression". *SIAM Journal on Optimization* 21.3 (2011), pp. 996–1026.

[70] J. van der Herten, I. Couckuyt, D. Deschrijver, and T. Dhaene. "Fast Calculation of the Knowledge Gradient for Optimization of Deterministic Engineering Simulations". *Submitted to the Journal of Machine Learning Research (JMLR)* (2017).

[71] J. Wu and P. Frazier. "The Parallel Knowledge Gradient Method for Batch Bayesian Optimization". *Advances in Neural Information Processing Sys-*

tems 29. Ed. by D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett. Curran Associates, Inc., 2016, pp. 3126–3134.

[72] I. Couckuyt, J. Aernouts, D. Deschrijver, F. D. Turck, and T. Dhaene. "Identification of quasi-optimal regions in the design space using surrogate modeling". *Engineering with Computers* 29.2 (2013), pp. 127–138.

[73] J. van der Herten, I. Couckuyt, and T. Dhaene. "Hypervolume-based Multi-objective Bayesian Optimization with Student-t Processes". *BayesOpt 2016 Workshop at the 30th Conference on Neural Information Processing Systems*. Barcelona, Spain, 2016.

[74] J. González, Z. Dai, A. Damianou, and N. Lawrence. "Bayesian Optimisation with Pairwise Preferential Returns". *BayesOpt 2016 Workshop at the 30th Conference on Neural Information Processing Systems*. Barcelona, Spain, 2016.

[75] J. H. Metzen. "Minimum Regret Search for Single- and Multi-Task Optimization". *Proceedings of the 33rd International Conference on Machine Learning (ICML-16)*. Ed. by M. F. Balcan and K. Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 192–200.

[76] P. Campigotto, A. Passerini, and R. Battiti. "Active learning of Pareto fronts". *Neural networks and learning systems, IEEE transactions on* 25.3 (2014), pp. 506–519.

[77] L. Shi, Y. Fu, R.-J. Yang, B.-P. Wang, and P. Zhu. "Selection of initial designs for multi-objective optimization using classification and regression tree". *Structural and Multidisciplinary Optimization* 48.6 (2013), pp. 1057–1073.

[78] J. Bect, D. Ginsbourger, L. Li, V. Picheny, and E. Vazquez. "Sequential design of computer experiments for the estimation of a probability of failure". *Statistics and Computing* 22.3 (2012), pp. 773–793.

[79] C. Chevalier, J. Bect, D. Ginsbourger, E. Vazquez, V. Picheny, and Y. Richet. "Fast parallel kriging-based stepwise uncertainty reduction with application to the identification of an excursion set". *Technometrics* 56.4 (2014), pp. 455–465.

[80] P. Singh, J. van der Herten, D. Deschrijver, I. Couckuyt, and T. Dhaene. "A sequential sampling strategy for adaptive classification of computationally expensive data". *Structural and Multidisciplinary Optimization* (2016), pp. 1–14.

[81] K. J. Lang. "Learning to tell two spirals apart". *Proc. of 1988 Connectionist Models Summer School*. 1988.

[82] F. A. Viana, G. Venter, and V. Balabanov. "An algorithm for fast optimal Latin hypercube design of experiments". *International journal for numerical methods in engineering* 82.2 (2010), pp. 135–156.

[83] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. "A fast and elitist multiobjective genetic algorithm: NSGA-II". *Evolutionary Computation, IEEE Transactions on* 6.2 (2002), pp. 182–197.

[84] G. Turk and M. Levoy. "Zippered polygon meshes from range images". *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. ACM. 1994, pp. 311–318.

[85] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. "Lipschitzian optimization without the Lipschitz constant". *Journal of Optimization Theory and Applications* 79.1 (1993), pp. 157–181.

[86] M. T. Hagan and M. B. Menhaj. "Training feedforward networks with the Marquardt algorithm". *Neural Networks, IEEE Transactions on* 5.6 (1994), pp. 989–993.

[87] R. Houthooft, J. Ruyssinck, J. van der Herten, S. Stijven, I. Couckuyt, B. Gadeyne, F. Ongenae, K. Colpaert, J. Decruyenaere, T. Dhaene, et al. "Predictive modelling of survival and length of stay in critically ill patients using sequential organ failure scores". *Artificial intelligence in medicine* 63.3 (2015), pp. 191–207.

[88] I. Couckuyt, D. Deschrijver, and T. Dhaene. "Fast calculation of multiobjective probability of improvement and expected improvement criteria for pareto optimization". *Journal of Global Optimization* 60.3 (2013), pp. 575–594.

[89] K. Elsayed. "Optimization of the cyclone separator geometry for minimum pressure drop using Co-Kriging". *Powder Technology* 269 (2015), pp. 409–424.

[90] Y. Xiong, W. Chen, D. Apley, and X. Ding. "A non-stationary covariance-based Kriging method for metamodelling in engineering design". *International Journal for Numerical Methods in Engineering* 71.6 (2007), pp. 733–756.

[91] J. A. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, J. Suykens, and T. Van Gestel. *Least squares support vector machines*. Vol. 4. World Scientific, 2002.

"Non-technical questions sometimes don't have an answer at all."

— Linus Torvalds

4

Fuzzy Local Linear Approximations

J. van der Herten, I. Couckuyt, D. Deschrijver, and T. Dhaene.
***SIAM Journal on Scientific Computing* 37.2 (2015), A1020–A1039**

4.1 Introduction

To avoid many real-life experiments and countless prototypes, modern engineering problems rely heavily on highly accurate computer simulations to reduce costs, time and (potentially) risks. The simulations are used to help the engineer understand the relation between inputs and the outputs of the system, and to identify interesting regions in the design space.

The downside of using high-accuracy simulations is that one simulation of a complex system with several inputs (commonly referred to as *variables*), and outputs (also called *responses*) can be very expensive in terms of computation time [2, 3]. These lengthy or expensive computations often make it impractical to use simulations directly for design exploration and gaining insight in the complex system

behaviour. Most optimisation algorithms require many simulations in the search space which makes optimisation a computationally expensive task.

An extra abstraction layer can be used to expedite the process. The simulator (which approximates the real world) is approximated by *surrogate models* (also known as response surface models or meta-models). These computationally cheap replacement models can be used to analyse or optimise the complex system while minimising the required number of expensive simulations. For this study, we make two assumptions: the simulator is deterministic which means that running the simulation twice with the same input parameters always produces the same results. Secondly, the complex system is treated as a gray or black box (little or nothing is known about the inner working of the system).

Surrogate models can be used for optimisation: in this context a *local* surrogate model is constructed to guide an optimisation algorithm towards an optimum. Afterwards, the model is no longer of use and discarded. This is not the case in *global* surrogate modelling, which aims to construct a model that approximates the behaviour of the system over the entire domain. This surrogate model can afterwards be used instead of the expensive simulator.

The simulator can be defined as an unknown function $f : \mathcal{X} \rightarrow \mathbb{C}$ with $\mathcal{X} \subset \mathbb{R}^d$, which maps a d-dimensional input vector of real inputs to a possibly complex output. This function is sampled at a discrete set of data n points: $X = \{ \boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n \}$. These data points (called the experimental design) are evaluated by the simulator and their responses are denoted as $F = \{ f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), ..., f(\boldsymbol{x}_n) \}$. Based on this information a surrogate model $\tilde{f}$ is chosen from a set of candidate approximation functions. This choice is usually guided by predefined quality criteria (such as cross-validation). Since acquiring the responses is computationally expensive but necessary to build an accurate surrogate model, the goal is to keep the set X as small as possible while still obtaining good accuracy. The choice of the data points in X is of crucial importance for constructing an accurate surrogate model with a reduced amount of points. Intuitively, the data points should be spread over the domain in such a way they capture a maximum amount of information on the behaviour of f. Since f is considered to be a black box, this is a difficult task.

The LOLA-Voronoi algorithm [4, 5], an earlier proposed hybrid iterative scheme that distributes the points to cover the design space such that the data density is distributed proportional to the non-linearity of f has proven to be very useful in several studies in several fields: [6–12]. Non-linear regions are more difficult to model, so the additional data points in these regions greatly help the search for a good approximation $\tilde{f}$. The algorithm combines an approach that estimates the gradient in the data points based on Local Linear Approximation (LOLA), and a Voronoi space-filling approach. The downside of the LOLA algorithm is that

it becomes very computationally demanding for high-dimensional design spaces. In this chapter we propose a new fuzzy based approach to overcome this issue. This approach can replace LOLA without further modifications to the concept of LOLA-Voronoi.

4.2 Sequential design

The selection of data points can be determined by means of a *one-shot* approach: all points are chosen at once and simulated. This data is given to the modelling algorithm and a surrogate model is constructed. The location of the points in the design space is called the Design of Experiments (DoE). One-shot designs of computer experiments are usually space-filling meaning they try to cover the domain as equally as possible. Examples are (maximin) Latin Hypercubes [13] and fractional designs [14].

Sequential designs turn the one-shot approach into an iterative process. The data acquired and/or the constructed models from previous iterations are analysed in order to intelligently select locations for new data points. These additional points are evaluated and usually new models are constructed. Sequential design has two important benefits over one-shot designs: first of all, it is impossible to have too few or too many points: the iterative process it halted when the objectives are reached (i.e., the surrogate model meets the predefined accuracy goals [15]). In a one-shot setting too few points means restarting the process, whereas too many points means wasting time due to evaluating an expensive simulator more than required. Secondly, the information provided by the intermediate simulator responses and constructed models can be used to identify regions that are difficult to model. This allows the sampling distribution to be guided towards these regions.

4.2.1 Exploration and exploitation

Any sequential design method faces the trade-off between *exploration* and *exploitation*. Exploration involves exploring the complete design space for key regions such as discontinuities, steep areas, optima and stable regions that have not yet been identified. Usually exploration does not look at system responses and focusses on filling the design space as uniform as possible. Undersampling and oversampling no longer occurs when exploring the design space sequentially. Examples of sequential exploration methods can be found in [16]. Exploitation on the other hand analyses simulator responses and/or constructed models to sample regions that have been identified as interesting. One could sample near optima or discontinuities to capture the complex behaviour, or sample in regions where intermediate surrogate models

make large errors. Examples of methods that involve exploitation of the available experimental design information can be found in [17, 18].

These two concepts conflict with each other: exploration aims to look away from regions we already visited and focus on unexplored areas, whereas exploitation does the opposite and gathers more information about irregularities that have been spotted previously. If a sequential design only focusses on exploitation, certain key regions are potentially missed as the sequential design strategy is stuck sampling a region that was identified previously. To reduce this risk we could specify a large initial space-filling design, but this might result in oversampling the design space. Only focussing on exploration disallows the sampling distribution to be modified towards interesting regions as we end up with a sequential space-filling design. Finding a balance between exploration and exploitation can be done in many different ways, and can also be application dependent.

4.3 Exploitation using local approximations

Exploitation includes the responses from previous points to guide the sequential design process to interesting regions in the design spaces. The definition of interesting regions depends entirely on the context of the surrogate modelling process: for instance for optimisation interesting regions are those (possibly) containing optima. In the context of accurate global surrogate modelling this means distributing a minimal amount of points to find a model which accurately represents the systems response over the entire design space.

Previously, the LOLA algorithm was introduced to guide the sampling process towards regions in the domain that may be more difficult to approximate [4, 5]. Often, systems have a very linear response in a large part of the design space, but have one or more regions that behave very non-linearly. Sampling more densely in these "difficult" regions has proven to be a successful approach for global surrogate modelling. The LOLA algorithm first estimates the gradient in each point, which is the best local linear approximation of the system response. This approximation is compared to the true simulator responses for nearby points. To compute the gradient approximation $\boldsymbol{g}$ in a point $\boldsymbol{x}_r$, a set of point indices is defined known as the neighbourhood: $N(\boldsymbol{x}_r) = \{ u_1, \ldots, u_v \}, \forall p < n, p \neq r$. This set is used to construct the following least squares problem for gradient estimation:

$$\begin{bmatrix} x_{u_1,1} - x_{r,1} & \ldots & x_{u_1,d} - x_{r,d} \\ x_{u_2,1} - x_{r,1} & \ldots & x_{u_2,d} - x_{r,d} \\ \vdots & \ddots & \vdots \\ x_{u_v,1} - x_{r,1} & \ldots & x_{u_v,d} - x_{r,d} \end{bmatrix} \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_d \end{bmatrix} = \begin{bmatrix} f(\boldsymbol{x}_{u_1}) - f(\boldsymbol{x}_r) \\ f(\boldsymbol{x}_{u_2}) - f(\boldsymbol{x}_r) \\ \vdots \\ f(\boldsymbol{x}_{u_v}) - f(\boldsymbol{x}_r) \end{bmatrix}. \tag{4.1}$$

LOLA requires $v \geq 2d$, so this system is never underdetermined. Using this gradient estimate to predict the values of other points in the neighbourhood, we can compute the error between the linear approximation and the true simulator response at these points:

$$E(\boldsymbol{x}_r) = \sum_{i \in N(\boldsymbol{x}_r)} |f(\boldsymbol{x}_i) - (f(\boldsymbol{x}_r) + \boldsymbol{g}(\boldsymbol{x}_i - \boldsymbol{x}_r))|. \tag{4.2}$$

The error E is referred to as the *non-linearity score*. The region surrounding $\boldsymbol{x}_r$ is non-linear if the error is large, as a linear prediction will be insufficient to capture the system response locally. Theoretically, LOLA chooses new points in locations near points with high non-linearity scores. In practice, however, LOLA is combined with a Voronoi exploration based component (see Section 4.5.1).

4.3.1 How to determine the neighbourhood

A key issue in LOLA is how to determine the neighbourhood $N(\boldsymbol{x}_r)$ of a point $\boldsymbol{x}_r$, which is referred to as the *reference point*. Determining this set is essentially a multi-objective optimisation problem which optimises two criteria:

1. *Cohesion*: A neighbour should be as close to the reference point as possible, as we are constructing a local approximation.
2. *Adhesion*: The neighbours should be as far away from each other as possible, in order to cover the space surrounding the reference point.

Clearly, it is impossible to maximise both. If the neighbours are very close to the reference point (high cohesion), they are close to each other as well (high adhesion). Points further away can have better adhesion, but can result in a bad local approximation. Unfortunately, there is no known general solution to place an arbitrary number of points in an ideal configuration on a (hyper)sphere [19].

The original LOLA algorithm [5] solves this optimisation problem by comparing a neighbourhood with an optimal configuration known as the cross-polytope. This configuration always has $2d$ points (which explains the constraint of $v \geq 2d$). This configuration is intuitive: for one-dimensional problems this results in one neighbour on each side of the reference point, for two dimensions this is a square, etc. For each $\boldsymbol{x}_r$, all possible subsets of $X \setminus \boldsymbol{x}_r$ of v points are constructed and compared to the cross-polytope. The set which resembles the cross-polytope is chosen as $N(\boldsymbol{x}_r)$. When new points are available, each point in the neighbourhood is removed and replaced by a new point. If this results in a better configuration, then the neighbourhood is updated. This solution is very elegant and leads to quasi-optimal configurations in terms of cohesion and adhesion. Because in an ideal cross-polytope configuration the vectors $\boldsymbol{x}_{u_i} - \boldsymbol{x}_r$, $i = 1, ..., 2d$ are orthogonal and

the method generates neighbourhoods resembling a cross-polytope, this results in a well conditioned system for Equation (4.1).

However, the downside of this approach is its complexity: $\mathcal{O}\left(2^{2d} n N_{\text{new}}\right)$ (N_{new} represents the amount of new samples the algorithm proposes for evaluation). Two optimisations to the algorithm were proposed, affecting mostly the n and N_{new} component. The "too far" heuristic excludes certain points from addition to the neighbourhood as they are mathematically unable to improve the neighbourhood. This makes the algorithm very powerful and usable for low-dimensional problems. Issues appear, however, when using LOLA to build global surrogate models for problems of higher dimensionality. As d becomes larger the neighbourhood size increases, which causes each new point to result in many new candidate neighbourhoods that need to be evaluated. Additionally, due to the *curse of dimensionality* more points will be required to obtain sufficient information to construct an accurate surrogate model. Problems of four dimensions and higher will spend a very long time on determining where to choose new samples when using the LOLA algorithm for exploitation.

4.3.2 Novel approach to determine the neighbourhood

Surrogate modelling of high-dimensional systems can be computationally very demanding as a lot of (expensive) data points are required to construct accurate models. The complexity of many modelling types, such as Kriging and Radial Basis Function (RBF) models, scales badly with sample size and design space dimensionality. Having a sequential sampling algorithm that adds to the computational burden is undesirable. In this section a new approach to determine $N(\boldsymbol{x}_r)$ is introduced. This approach requires computing weights to include information about cohesion and adhesion. The weight computation is covered in Section 4.4.

The original neighbourhood selection procedure [5] reviewed for the Neighbourhood-Voronoi algorithm in Section 3.3 is selective: no matter how many points surround the reference point, a fixed amount of neighbours (v) is selected which means that, in some cases, valuable information is neglected. The new algorithm therefore includes all points within a certain range of the reference point given by ρ:

$$N(\boldsymbol{x}_r) = \{\, i \mid \boldsymbol{x}_i \in X_r, \|\boldsymbol{x}_i - \boldsymbol{x}_r\| < \rho(\boldsymbol{x}_r) \,\}, \tag{4.3}$$

with $X_r = X \setminus \boldsymbol{x}_r$. We assume that each parameter was scaled to compatible ranges, and an appropriate distance metric is used. In Section 4.5.2, a brief discussion on distances in high-dimensional spaces is given. The regulatory distance given by the ρ function in Equation (4.3) controls the part of the input space that is included in the gradient estimation. It defines the notion "local" for $\boldsymbol{x}_r$. It can be proportional

to the average distance between points, or it can be time controlled. In this chapter the following heuristic is used:

$$\rho(\boldsymbol{x}_r) = \begin{cases} \frac{2}{K}\sum_{j=1}^{K} \|\boldsymbol{n}_j - \boldsymbol{x}_r\| & \text{if } |N(\boldsymbol{x}_r)| > d \\ \|\boldsymbol{n}_d - \boldsymbol{x}_r\| & \text{otherwise.} \end{cases} \tag{4.4}$$

with $\boldsymbol{n}_j$ the j-th nearest neighbour of $\boldsymbol{x}_r$ in the input space. By default, the heuristic represents twice the average distance to the K nearest neighbours. The parameter K is chosen in function of the input dimensionality to assure sufficient points are included to avoid an underdetermined system in Equation (4.1). For all experiments in this chapter it was chosen to be $4d$.

When a point is very isolated, ρ will be large to include sufficient points in the gradient estimation to avoid an underdetermined system. In a dense region, a smaller ρ will only include points that are sufficiently close to obtain an accurate gradient. If points that are distant would be included, they could smooth out the gradient in case of subtle non-linearities. Unfortunately, in case of a very isolated point Equation (4.4) still can result in $|N(\boldsymbol{x}_r)| < d$, which turns Equation (4.1) into an underdetermined system. In this situation, ρ is raised to $\|\boldsymbol{n}_d - \boldsymbol{x}_r\|$ to include the d nearest neighbours.

Note how this definition of $N(\boldsymbol{x}_r)$ no longer select points based on adhesion and cohesion as defined above. However, we still require to include this information into our gradient estimation. This issue is covered by assigning weights to each neighbour. In the next section we come up with a strategy to assign the weights.

As we are no longer chasing the cross-polytope, we risk instability when solving Equation (4.1). However, due to nature of experimental design, points will still be spread out over the design space as much as possible, leading to a surrounding configuration. Because of this property most of the vectors $\boldsymbol{x}_{u_i} - \boldsymbol{x}_r$, $i = 1, ..., v$ have different directions, which results in a well-conditioned matrix.

4.4 Determining the neighbour weights

Attaching proper weights to each neighbour of $\boldsymbol{x}_r$ and solving Equation (4.1) as a weighted least square problem reintroduces the concept of cohesion and adhesion. The weights reflect how much influence each point in $N(\boldsymbol{x}_r)$ has in the gradient estimation. Points with high cohesion and low adhesion are preferred and are assigned high weights, while low cohesion and/or high adhesion result in low weights.

First, we mathematically define cohesion and adhesion $\forall i \in N(\boldsymbol{x}_r)$:

$$C(\boldsymbol{x}_r, \boldsymbol{x}_i) = \|\boldsymbol{x}_i - \boldsymbol{x}_r\|, \tag{4.5}$$

$$A(\boldsymbol{x}_r, \boldsymbol{x}_i) = \min_{\boldsymbol{q} \in X_r} \|\boldsymbol{q} - \boldsymbol{x}_i\|. \tag{4.6}$$

We refer to high cohesion if the neighbouring points being very close (which corresponds to lower values for C) in comparison to other points in $N(\boldsymbol{x}_r)$. On the other hand, low adhesion corresponds to large values for A. For simplicity, $C(\boldsymbol{x}_r)$ and $A(\boldsymbol{x}_r)$ are vectors which represent cohesion and adhesion values for all neighbours of $\boldsymbol{x}_r$. In Section 4.4.2 a system based on Fuzzy logic is defined to determine the neighbour weights. The concept of a Mamdani Fuzzy Inference System (FIS) is first explained in the next section.

4.4.1 Mamdani fuzzy inference system

A FIS maps inputs to outputs, using fuzzy set theory. Common types include the Mamdani and the Sugeno FIS. In this section we briefly explain the concept of the former.

Fuzzy sets and concepts were introduced as a way to represent data imprecisely. An example is the weight of a person: we can express the weight numerically using a number with a unit, but we can also treat a persons weight as a *linguistic variable* with *linguistic values*. Someone can be skinny, normal, or heavy. Typical for this type of statements is the lack of clear boundaries: when is someone no longer skinny but normal? Usually there is a gray zone between the linguistic values. Mathematically this is expressed by means of fuzzy sets. A crisp[1] set A has a simple membership function $\chi_A : A \to \{0, 1\}$: an element is either member or not. For a fuzzy set, the membership function is less strict and takes the form of $\gamma_A : A \to [0, 1]$. Usually a membership value of zero indicates complete non-membership, whereas one represents complete membership. Values in between indicate intermediate degrees of membership.

Built on the theory of fuzzy sets, a FIS consists of a fuzzifier, an inference engine, and a defuzzifier. The fuzzifier maps crisp inputs of linguistic variables to fuzzy set memberships, using provided membership functions. These membership degrees are fed into a rule-based inference engine, which processes rules of the form "if-then". To process the rules, we need to be able to process operations such as AND and OR within the rules. In fuzzy logic, these operations are known as fuzzy combinations. Many possible operations have been proposed, in this chapter the minimum t-norm and maximum t-conorm are used.

The output of these rules (of which some might not be activated, depending on the input) is combined (usually by applying a fuzzy OR) and defuzzified. A popular method for defuzzification is the centroid method. For more information about

[1] A traditional set.

Mamdani FIS, the reader is referred to [20]. More information on fuzzy logic and the t-(co)norms can be found in [21].

4.4.2 Fuzzy-based neighbour weight assignment

Although we defined crisp values for cohesion and adhesion with Equation (4.5) and Equation (4.6), it is clear that handling these two quantities as linguistic variables is much more convenient. In fact, reasoning with crisp values for cohesion and adhesion makes the problem complex. In what follows, a FIS $\mathcal{S}$ is proposed to assign weights to each point in $N(\boldsymbol{x}_r)$. The system has two input parameters: cohesion and adhesion, and produces a weight as output.

For the cohesion input parameter, one fuzzy set referred to as "high" with membership function $\mathcal{C}_{\text{high}}$ is defined:

$$\begin{aligned} \mathcal{C}_{\text{high}}: \quad & [0,\rho] \to [0,1], \\ & x \mapsto \tfrac{1}{1+\exp\left(-\zeta_c\left(\frac{x}{\rho}-0.5\right)\right)}. \end{aligned}$$

Points that have a small distance to the reference point $\boldsymbol{x}_r$ have a high membership degree (corresponding to high cohesion) of the fuzzy set, points that are far away do not.

For adhesion, two fuzzy sets with membership functions $\mathcal{A}_{\text{low}}$ and $\mathcal{A}_{\text{high}}$ exist:

$$\begin{aligned} \mathcal{A}_{\text{low}}: \quad & [0, A_{\max}] \to [0,1], \\ & x \mapsto \exp\left(\tfrac{-(x-A_{\max})^2}{2(A_{\max}\zeta_{al})^2}\right), \\ \mathcal{A}_{\text{high}}: \quad & [0, A_{\max}] \to [0,1], \\ & x \mapsto \exp\left(\tfrac{-x^2}{2(A_{\max}\zeta_{ah})^2}\right). \end{aligned}$$

$A_{\max}$ is the maximum adhesion value for $A(\boldsymbol{x}_r)$. ζ_c, ζ_{al} and ζ_{ah} are the hyperparameters of the membership functions. Larger values for the ζ_a values result in wider Gaussian membership functions, which means higher membership values to the sets. It is possible to define a single adhesion membership function and define the other as the negation (as for the cohesion), but two membership functions allow more control for the adhesion parameter. An illustration of the membership functions is given in Figure 4.1.

For the output, three triangular membership functions are defined (low, average and high) as shown in Figure 4.2a. To complete the definition of the FIS, the rules listed in Table 4.1 are included. The system processes the cohesion and adhesion for each $\boldsymbol{x} \in N(\boldsymbol{x}_r)$ and computes the membership degree for the Fuzzy sets $\mathcal{C}_{\text{high}}$, $\mathcal{A}_{\text{low}}$ and $\mathcal{A}_{\text{high}}$ (this step is reffered to as *fuzzification*). Next, all rules are evaluated

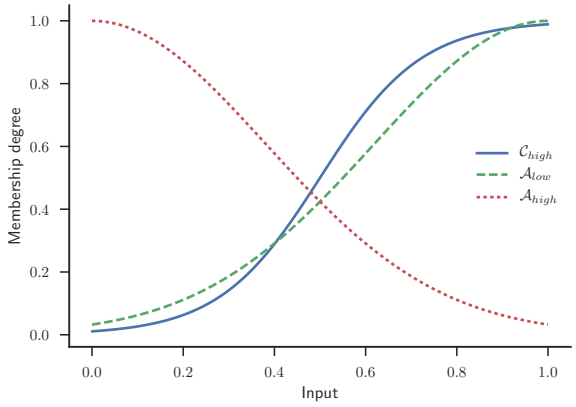


Figure 4.1: Neighbour weight assignment: Membership functions for the fuzzy sets for $\rho = 1$, $A_{max} = 1$, $\zeta_c = 9$, $\zeta_{al} = 0.27$ *and* $\zeta_{ah} = 0.3$.

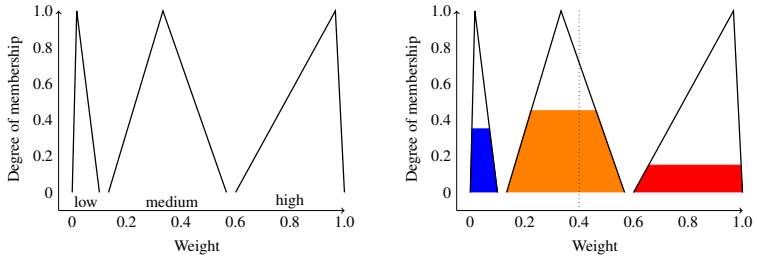


Figure 4.2: Neighbour weight assignment: Illustration of the weight determination with the FIS. (a) The output membership functions low, average and high for $\mathcal{S}$. *(b) Clipped functions for the example with input membership degrees of 0.5 in* $\mathcal{C}_{high}$, *0.15 in* $\mathcal{A}_{low}$ *and 0.45 in* $\mathcal{A}_{high}$ *are shown. The vertical line indicates the weight obtained by defuzzification of the resulting output membership distribution by the centroid method.*

Table 4.1: Neighbour weight assignment: FIS rules to infer the membership for the weight linguistic variable.

	Cohesion	Adhesion	Weight
1	high	low	high
2	high	high	average
3	not high	low	average
4	not high	high	low

to assign a degree of membership to the triangular output member functions. The output degree of membership corresponds to the result of the evaluation of the rule expressions. For example, after fuzzification a point has a membership degree of 0.5 in $\mathcal{C}_{\text{high}}$, 0.15 in $\mathcal{A}_{\text{low}}$ and 0.45 in $\mathcal{A}_{\text{high}}$. According to the first rule, this point has a degree of membership of 0.15 in the fuzzy set high[2]. The output membership function is then clipped by the obtained degree of membership (y-axis in Figure 4.2). By applying a fuzzy OR over the obtained output membership values, the final output membership distribution is obtained (Figure 4.2b illustrates this for the example). The centroid defuzzification method is then used to convert the result into a crisp value for the weight (x-axis in Figure 4.2).

Figure 4.3 shows the response surface of $\mathcal{S}$. Highly cohesive points with low adhesion are preferred, as opposed to low cohesive points with high adhesion. It is possible to use different membership functions (for example sigmoid instead of Gaussian membership functions), or to define more fuzzy sets both for inputs as well as the output. Throughout the rest of this chapter, we use the FIS defined above.

An illustration of how weights are assigned by $\mathcal{S}$ is shown in Figure 4.4. The ideal configuration of the points in two dimensions (the cross-polytope) is illustrated in Figure 4.4a. Each point is assigned an equal weight, which is not surprising as all cohesion and adhesion values are identical. Figure 4.4b shows a more complex situation. Instead of one sample at $(-1, 0)$ we now have three points: two close to each other, and a third one somewhat further away. In this case, the weight is divided amongst the two neighbouring points. If the weight of these points would be added, they would roughly add up to the weights of the previous case, which means that more points are contributing to the gradient estimation. The third point on the left at $(-0.7, 0.6)$ is further away and provides information about a different direction. Therefore it is assigned a higher weight, although it doesn't have the same impact as the other standalone points. In fact, it has taken over some weight from the two points at $(-1, 0)$ and the point at $(0, 1)$.

[2]Due to the choice of minimum as t-norm and maximum as t-conorm. For more information on norms and fuzzy logic, the reader is referred to [21]

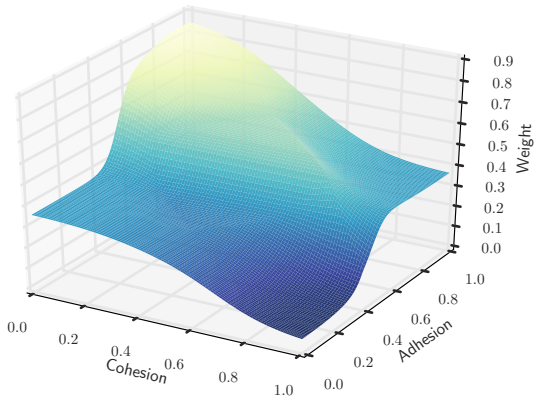
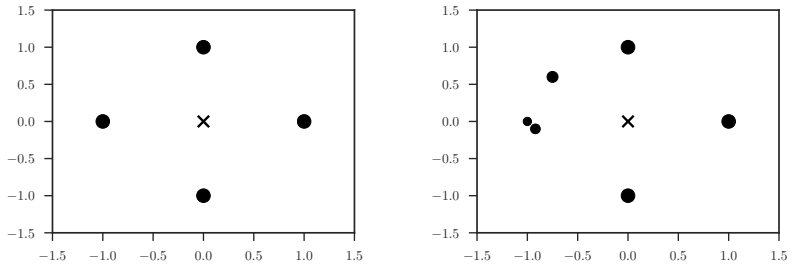


Figure 4.3: Neighbour weight assignment: example of the response surface of the FIS $\mathcal{S}$ for $\rho = 1$, $A_{max} = 1$, $\zeta_c = 9$, $\zeta_{al} = 0.27$ and $\zeta_{ah} = 0.3$. The cohesion and adhesion on the axes correspond to Equations (4.5) and (4.6), not the response of the membership functions.



(a) *(b)*

Figure 4.4: Neighbour weight assignment: illustration of weight assigments with $\mathcal{S}$ using the same constants as Figure 4.3. The cross indicates the reference point $\boldsymbol{x}_r$, the size of each neighbour the weight. (a) The ideal (cross-polytope) configuration: all weights are equal. (b) The left side has three points: the weights have been distributed amongst them.

4.5 Hybrid sequential design method

In Section 4.3, a previously introduced exploitation method was discussed. A key complexity issue for high-dimensional problems was identified and a new approach to construct the neighbourhoods was introduced. Section 4.4 introduced a fuzzy-based mechanism to assign weights to each neighbour, based on cohesion and adhesion. All these concepts are now brought together into a new approach that can take the place of the LOLA algorithm in LOLA-Voronoi.

4.5.1 Fuzzy local linear approximation

Algorithm 1 Fuzzy LOLA (FLOLA): this exploitation algorithm computes a score $\forall \boldsymbol{x} \in X$, indicating the non-linearity of the region surrounding $\boldsymbol{x}$. New samples are chosen in the neighbourhood of the N_{new} highest ranked samples.

Require: X, F, ζ_c, ζ_{al}, ζ_{ah}, N_{new}
  initiate $\mathcal{S}$ (Section 4.4.2)
  Calculate distance matrix for $\boldsymbol{x}$
  for all $\boldsymbol{x}_r \in X$ **do**
    Compute $\rho(\boldsymbol{x}_r)$ (Equation (4.4))
    Initialise $N(\boldsymbol{x}_r)$ (Equation (4.3))
    Determine $C(\boldsymbol{x}_r)$ and $A(\boldsymbol{x}_r)$ (Equations (4.5) and (4.6))
    Compute weights $\boldsymbol{\omega}$ by evaluating $\mathcal{S}$
    Estimate $\boldsymbol{g}$ (Equation (4.1)), given $\boldsymbol{\omega}$
    Calculate error on gradient estimation (Equation (4.2))
  end for
  Pick N_{new} samples with highest non-linearity score
  X_{new} = new samples in the neighbourhood of these samples
  $X = X \cup X_{\text{new}}$

The weights computed by $\mathcal{S}$ can be used to solve Equation (4.1) as a Weighted Least Squares[3] problem to estimate the gradient. After obtaining $\boldsymbol{g}$, we can compute the non-linearity score. An overview of this new approach, known as *Fuzzy Local Linear Approximation* (FLOLA) is given in Algorithm 1. Because the size of the system Equation (4.1) is not dependent on the size of the set X, only the for loop contributes to the complexity of the algorithm: this results in a complexity of $\mathcal{O}(N)$ which is a massive improvement compared to LOLA. Furthermore, the for-loop allows parallel computation since each iteration is independent. The biggest cost are many distance calculations to determine $\rho(\boldsymbol{x}_r)$, $A(\boldsymbol{x}_r)$ and $C(\boldsymbol{x}_r)$. This is solved by computing a distance matrix once prior to the for-loop: this matrix contains all

[3] Note that the weights are computed for each point $\boldsymbol{x}$ separately. This essentially means we turned the gradient estimation into a Moving Least Squares problem.

required information for computations inside the loop. Furthermore, for the next iteration of the sequential design process this distance matrix may be expanded by adding rows and columns avoiding a quadratic complexity. Distances matrices tend to occupy a lot of memory in case of many points, which is unlikely for this algorithm in the context of surrogate modelling as each evaluation is expensive. Due to the limited size of the set X, the size of the matrix remains manageable.

4.5.2 Including an exploration metric

Similar to LOLA, the exploitation based algorithm Fuzzy Local Linear Approximation (FLOLA) can be complemented with a Voronoi approximation based exploration component and form *FLOLA-Voronoi*. For each point $\boldsymbol{x}_r$, the non-linearity score E_{fuzzy} is complemented with a measure V indicating an approximation of the relative Voronoi cell size of the reference point. For more information on approximating the size of a Voronoi cell, the reader is referred to [5]. The value of V is in the range $[0, 1]$ so E_{fuzzy} is first normalised and then added to V:

$$H_{\text{fuzzy}}(\boldsymbol{x}_r) = V(\boldsymbol{x}_r) + \frac{E_{\text{fuzzy}}(\boldsymbol{x}_r)}{\sum_{i=1}^{N} E_{\text{fuzzy}}(\boldsymbol{x}_i)}. \tag{4.7}$$

For clarity, the pseudocode of FLOLA-Voronoi is shown in Algorithm 2. The only difference with LOLA-Voronoi is the algorithm used to calculate $E_{\text{fuzzy}}(\boldsymbol{x}_r)$. The hybrid score H is then used to rank all currently available points according to the non-linearity and the sample density of the surrounding region. The N_{new} highest ranked reference points are selected to assign new points in the next iteration. The position of the point is determined by considering local space-fillingness. Usually the position maximising the minimum distance from both the reference point as well as its neighbours is chosen.

The combination of both criteria guarantees we do not get stuck in one region of the design space and no large areas are left unexplored. However, the exploitation score pushes the strategy to sample non-linear regions much denser when they are discovered. When these regions are sampled dense enough, the FLOLA score will lower, and exploration will take over. This additional information on non-linear regions helps the surrogate model to capture the non-linear behaviour accurately as more information is provided on irregularities. In Equation (4.7), the exploration and exploitation component contribute equally. It is possible to use a different balance, or even change the balance dynamically as more samples become available. For more information, the reader is referred to [22].

Setting $N_{\text{new}} = 1$ is optimal, as each sampling decision can be made with the latest information at hand. This means that when a new non-linear region is discovered

it is exploited immediately. However, adding add more samples each iteration does not lead to undesired clusters or a bad design, since only a single additional point can be placed in each Voronoi cell during one iteration. For high-dimensional problems this is recommended as fitting a surrogate model may be expensive.

Algorithm 2 FLOLA-Voronoi: hybrid sequential strategy. Combines an exploitation and an exploration score (FLOLA and Voronoi respectively) and selects a new candidate samples in the neighbourhood of the N_{new} highest ranked samples.

Require: X, F, ζ_c, ζ_{al}, ζ_{ah}, N_{new}
  for all $\boldsymbol{x}_r \in X$ **do**
    Calculate $E_{\text{fuzzy}}(\boldsymbol{x}_r)$ (Equation (4.2))
    Calculate $V(\boldsymbol{x}_r)$ (See [5])
    Compute $H_{\text{fuzzy}}(\boldsymbol{x}_r)$ (Equation (4.7))
  end for
  Sort X by H_{fuzzy}
  for $i = 1$ **to** N_{new} **do**
    $\boldsymbol{x}_{\text{new}} \leftarrow$ location near $\boldsymbol{x}_i$
    $P_{\text{new}} \leftarrow X_{new} \cup \boldsymbol{x}_{\text{new}}$
  end for

4.6 Remarks

4.6.1 Distances in high-dimensional space

Throughout this entire chapter, distance between vectors $a, b \in \mathbb{R}^d$ was indicated as $\|a - b\|$, without specifying the actual distance metric. The most commonly used distance metric is the well-known Euclidean distance which is essentially a Minkowski distance (Equation (4.8)) for $p = 2$.

$$l_p = \|a - b\|_p = \left(\sum_{i=1}^{d} (|a_i - b_i|)^p \right)^{1/p}. \tag{4.8}$$

However, in high-dimensional spaces the Euclidean distance can fail to provide a meaningful notion to the concept of proximity. This is known as the concentration of norms, and affects all Minkowski distances for $p \geq 1$ [23] in reverse order (i.e., the l_∞ max norm is more likely to encounter the concentration effect than the l_2 Euclidean distance, and the l_1 Manhattan distance is even less likely to produce concentrated distances). Later it was highlighted that this is effect is more likely to occur with uncorrelated data [24]. A statistical test was introduced to compute a

bound on the probability of distance concentration given a dissimilarity function (such as Minkowski distances) and a set of data. The existence of structure and correlation in data typically indicates the existence of some lower dimensional manifold, and reduces the concentration effect. Unfortunately, the concept of generating designs typically does not involve the generation of a correlated set of inputs (in fact space-filling designs try to avoid this).

As a solution, it is often suggested to consider fractional distances. In fact this can be interpreted as a Minkowski distance with $p \in]0, 1]$ [4]. The latter does not magically solve the concentration effect but as the dimensionality increases, choosing a value of p closer to zero reduces the probability of concentration effects. In our experimental setup, we consider fractional distances for the high-dimensional problems with $p = \frac{1}{d}$, to test if they result in better designs. Unfortunately computing these distances comes at a high computational cost as the approximation of p-th root is very expensive, and this computation is now required along each dimension rather than the sum as is the case for Minkowski distances with $p \geq 1$. For problems of very high dimensionality it is often suggested to $p = \frac{1}{\lfloor \log(d) \rfloor + 1}$ to gain some speed.

4.6.2 Impact of output uncertainty

So far, we have only considered the case with a directly observable function f. This implies evaluating the same points again will result in an identical response (*deterministic response*). We now analyse the behaviour of the FLOLA-Voronoi method for responses observed with noise (*stochastic response*). We assume the observed value for a point is subject to Normal distributed uncertainty:

$$y_i \sim \mathcal{N}\left(f(\boldsymbol{x}_i), \sigma_n^2\right).$$

As the exploration algorithm (Voronoi cell size estimation) does not consider the outputs of the simulation and only considers the d-dimensional input points P, it is insensitive to σ_n^2. However, the non-linearity error (Equation (4.2)) sums the differences between the linear prediction and the response obtained from the simulator. This score depends on the output and therefore is sensitive to σ_n^2. We now show how much the uncertainty impacts E, under our assumption of normal distributed noise. This can be rewritten as

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i \qquad \text{with } \epsilon_i \sim \mathcal{N}(0, \sigma_n^2).$$

[4] Strictly speaking these dissimilarity functions are no longer distances as they do not satisfy the triangular inequality

By expanding the terms and applying the triangular inequality:

$$\tilde{E}(\boldsymbol{x}_r) = \sum_{i \in N(\boldsymbol{x}_r)} |\eta_i| \tag{4.9a}$$

$$= \sum_{i \in N(\boldsymbol{x}_r)} |f(\boldsymbol{x}_i) + \epsilon_i - (f(\boldsymbol{x}_r) + \epsilon_r + \boldsymbol{g}(\boldsymbol{x}_i - \boldsymbol{x}_r))|, \tag{4.9b}$$

$$\leq \sum_{i \in N(\boldsymbol{x}_r)} |f(\boldsymbol{x}_i) - (f(\boldsymbol{x}_r) + \boldsymbol{g}(\boldsymbol{x}_i - \boldsymbol{x}_r))| + \sum_{i \in N(\boldsymbol{x}_r)} |\epsilon_i - \epsilon_r|,$$

$$= E(\boldsymbol{x}_r) + \sum_{i \in N(\boldsymbol{x}_r)} |\epsilon_i - \epsilon_r|. \tag{4.9c}$$

Clearly the score for the deterministic response complemented with a sum of error terms forms an upper bound for the score. The computation of $\boldsymbol{g}$ now depends on $\boldsymbol{y}$ instead. Rewriting Equation (4.2):

$$\underbrace{\begin{bmatrix} \boldsymbol{x}_{u_1} - \boldsymbol{x}_r \\ \boldsymbol{x}_{u_2} - \boldsymbol{x}_r \\ \vdots \\ \boldsymbol{x}_{u_v} - \boldsymbol{x}_r \end{bmatrix}}_{\boldsymbol{A}} \boldsymbol{g} = \boldsymbol{y} - y_r.$$

Following the principles of Bayesian linear regression assuming a normal prior on $\boldsymbol{g} \sim \mathcal{N}(0, b^{-1}\boldsymbol{I}), b > 0$ and likelihood

$$\boldsymbol{y} - y_r | \boldsymbol{A}, \boldsymbol{g}, \sigma_n^2 \sim \mathcal{N}\left(\boldsymbol{f} - f(\boldsymbol{x}_r), 2\sigma_n^2 \boldsymbol{I}\right),$$

the posterior over $\boldsymbol{g}$ is a Multivariate Normal distribution (MVN) which can be computed analytically. It is given by

$$\boldsymbol{g} | \boldsymbol{y} - y_r, \boldsymbol{A}, \sigma_n^2, b \sim \mathcal{N}\left(\frac{1}{2\sigma_n^2} \Upsilon^{-1} \boldsymbol{A}^T (\boldsymbol{y} - y_r), \Upsilon^{-1}\right),$$

with

$$\Upsilon = \frac{1}{2\sigma_n^2} \boldsymbol{A}^T \boldsymbol{A} + b\boldsymbol{I}.$$

Hence, propagating this MVN through the dot product yields a univariate normal for the terms η_i. The variance is augmented twice by σ_n^2 due to the ϵ terms in Equation (4.9c). The absolute value folds part of the density along the probability

density axis. This is known as the folded normal distribution:

$$\mathbb{E}\left[\eta_i\right] = f(\boldsymbol{x}_i) - (f(\boldsymbol{x}_r) + \mathbb{E}\left[\boldsymbol{g}|\boldsymbol{y}, \boldsymbol{A}, \sigma_n^2, b\right](\boldsymbol{x}_i - \boldsymbol{x}_r)^T), \tag{4.10a}$$

$$\text{Var}\left[\eta_i\right] = (\boldsymbol{x}_i - \boldsymbol{x}_r)^T \Upsilon^{-1} (\boldsymbol{x}_i - \boldsymbol{x}_r) + 2\sigma_n^2, \tag{4.10b}$$

$$\mathbb{E}\left[|\eta_i|\right] = \sqrt{\frac{2\text{Var}\left[\eta_i\right]}{\pi}} \exp\left(\frac{\mathbb{E}\left[\eta_i\right]^2}{2\text{Var}\left[\eta_i\right]}\right) + \mathbb{E}\left[\eta_i\right]\left(1 - 2\Phi\left(\frac{-\mathbb{E}\left[\eta_i\right]}{\sqrt{\text{Var}\left[\eta_i\right]}}\right)\right)$$

$$|\eta_i| \sim \mathcal{FN}\left(\mathbb{E}\left[|\eta_i|\right], \mathbb{E}\left[\eta_i\right] + \mathbb{E}\left[|\eta_i|\right] + \text{Var}\left[\eta_i\right]\right).$$

Finally, $\tilde{E}(\boldsymbol{x}_r)$ corresponds to the sum of the v random variables $|\eta_i|$ (Equation (4.9a)). Unfortunately, the sum of an arbitrary number of folded normal distributions is not a known distribution, and its probability density function can no longer be computed easily, mainly because both the mean and variance of a folded normal distribution are coupled with the mean and variance of the unfolded normally distributed random variable. Hence, we aim to characterize the distribution by its moments. The average magnitude of the scores has no impact on the final ranking by FLOLA. However, if the second moment of the distribution on $\tilde{E}(\boldsymbol{x}_r)$ increases due to the observation noise, this can have an impact on the sample ranking and by extent, sampling decisions. Defining following quantities:

$$\begin{aligned}
\boldsymbol{\mu} &= \left[\mathbb{E}\left[\eta_{u_1}\right], \ldots, \mathbb{E}\left[\eta_{u_v}\right]\right]^T, \\
\boldsymbol{\sigma}^2 &= \left[\text{Var}\left[\eta_{u_1}\right], \ldots, \text{Var}\left[\eta_{u_v}\right]\right]^T, \\
\boldsymbol{\delta}_a &= \left[\mu_1 \,\text{erf}\left(\tfrac{\mu_1}{\sigma_1}\right) \quad \ldots \quad \mu_v \,\text{erf}\left(\tfrac{\mu_v}{\sigma_v}\right)\right]^T, \\
\boldsymbol{\delta}_b &= -4(\boldsymbol{\sigma}^3)^T \phi\left(\boldsymbol{\mu} \,\middle|\, 0, \text{diag}(\boldsymbol{\sigma}^2)\right) - \boldsymbol{\delta}_a,
\end{aligned}$$

the variance of the distribution resulting from the sum of folded normal-distributed random variables can be computed analytically

$$\text{Var}\left[\tilde{E}(\boldsymbol{x}_r)\right] = \boldsymbol{\mu}^T\boldsymbol{\mu} + \boldsymbol{\sigma}^T\boldsymbol{\sigma} + \frac{2}{\sqrt{\pi}}(\boldsymbol{\sigma}^3)^T \phi\left(\boldsymbol{\mu} \,\middle|\, 0, \frac{1}{2}\text{diag}(\boldsymbol{\sigma}^2)\right)^2 + \boldsymbol{\delta}_a^T\boldsymbol{\delta}_b. \tag{4.11}$$

Here ϕ returns a column vector of the probability density function of a MVN for the argument. The higher order moments are not considered further. Besides being mathematically complicated, the expression for the variance shares similarities with the variance of the folded normal distribution. The first two terms correspond directly, whereas the last two terms correspond to decreasing the variance due to folded segments of the distribution of the terms. Figure 5.1 visualises Equation (4.11), by varying the parameters mean and variance of one η term while keeping the others fixed. Hence it can be observed that higher σ_n^2 and more terms in the neighbourhood (as expressed by v) cause higher variance on the error term.

Using the neighbourhood selection mechanism of FLOLA-Voronoi the latter can be countered by including only the most significant neighbours in Equation (4.9b) (instead of all).

A direct consequence is that cells situated near linear output behaviour might be ranked higher due to the output uncertainty. This automatically triggers additional exploration which is a desirable property for stochastic applications. This is further illustrated by selecting 120 data points from the same function (*Peaks*) twice, once with output noise and once without. The density estimate of the final sampling distribution is shown in Figure 4.5. Clearly, the perturbed non-linearity scores cause additional exploration which is a desirable property when output is subject to uncertainty.

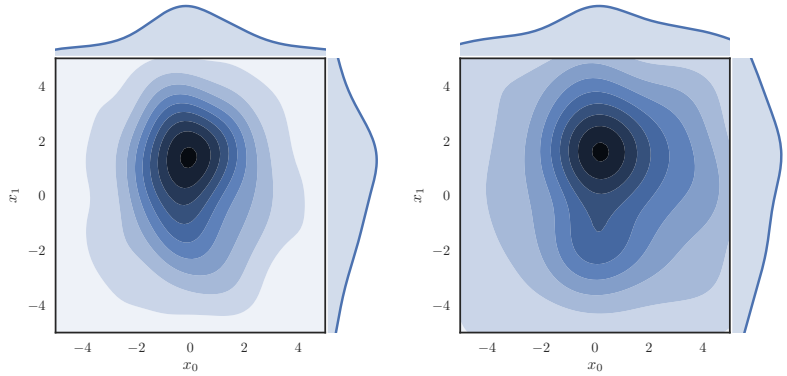
4.6.3 FLOLA for classification

Previously, Section 3.3 described how to use a modification of the LOLA-Voronoi algorithm to increase sampling density near class boundaries. Under this setting, the LOLA algorithm assigns a 1 to each cell for which the neighbourhood has a label disagreement, all others are assigned 0. This can be interpreted as significantly boosting all cells with a label disagreement amongst the neighbours. The tie-braker is then the Voronoi cell size.

The same approach applied with the FLOLA neighbourhoods can loose a lot of its exploitation properties. Because neighbourhoods can contain a lot of points, this would increase the scores of a lot of cells as many of them will have at least one sample with a different label. However, the only change required to overcome this difference and obtain FLOLA-Voronoi for classification is redefining Equation (4.2):

$$E(\boldsymbol{x}_r) = \max\left(\{\,\omega_i^{(r)} \mid i \in N(\boldsymbol{x}_r), f(\boldsymbol{x}_i) \neq f(\boldsymbol{x}_r)\,\} \cup \{\,0\,\}\right).$$

Unlike the LOLA algorithm, a score of 1 is not assigned as soon as any neighbour sample has a label disagreement. Instead, the highest weight amongst the neighbours with different labels is assigned. Following this scheme, samples with disagreements amongst the neighbours are only assigned elevated scores when the disagreements occur within the most important neighbours.



(a) $\sigma_n^2 = 0$ *(b)* $\sigma_n^2 = 1$

Figure 4.5: Output uncertainty: kernel density estimate of the sampling distribution obtained by applying FLOLA-Voronoi to the Peaks problem, with and without noise noise corruption. Clearly, the noiseless case is more focussed on the non-linear area whereas the noisy case results in more exploration.

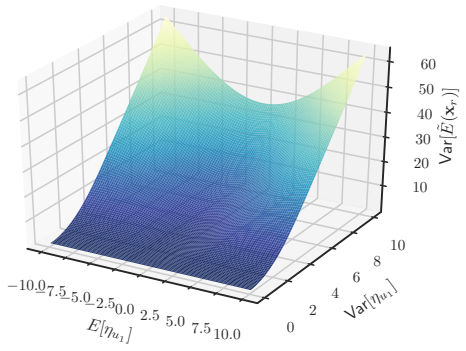


Figure 4.6: Output uncertainty: variance of the distribution of $\tilde{E}(\boldsymbol{x}_r)$*, as a function of mean and variance of one error term* η_{u_1} *(Equations (4.10a) and (4.10b)). All other elements of* $\boldsymbol{\mu}$ *and* $\boldsymbol{\sigma}^2$ *were kept constant.*

4.7 Experiments

In previous studies [6–12] in several research fields, LOLA-Voronoi has proven to be an excellent algorithm to building sequential designs. The sampling distribution is modified to focus on non-linear regions at the expense of a small computational cost for low-dimensional problems. For high-dimensional problems this cost quickly magnifies, which can be countered by using FLOLA-Voronoi. Throughout all experiments, the hyperparameters ζ_c, ζ_{al} and ζ_{ah} of the membership functions of FLOLA are fixed to 9, 0.27 and 0.3 respectively. These values were computed by minimising the Root-Mean-Square Error (RMSE) between the gradients computed by the LOLA and FLOLA algorithms on hundreds of randomly generated gradient estimation problems with varying dimensionality.

In this section, we first show by means of simple two-dimensional problems that the new algorithm performs very similarly to LOLA-Voronoi and has the same desirable properties. Next, higher dimensional problems are modeled to illustrate the performance gain of the new algorithm. Next to FLOLA-Voronoi and LOLA-Voronoi, four more sequential design strategies are tested: the first is a solely Voronoi-based sequential design which is a pure exploration based method. More recently, a new algorithm known as Delaunay-Hybrid Adaptive Sequential Design (DHASD) was proposed [25]. This method combines an exploitation and exploration metric based on a Delaunay triangulation, and dynamically balances between the two. The balancing strategy relies on several predefined parameters to avoid clustering. These parameters heavily influence the performance of the sampling strategy but need to be chosen by an expert or in function of the design space and the complexity of the problem at hand. This is a disadvantage for practical applications where nothing is known in advance. For our test cases, the parameters were chosen by trial-and-error.

An exploitation based Model Error (ME) strategy was included as well: this strategy evaluates the best models of previous generations on a dense grid. The outputs are compared and new samples are chosen in regions with largest differences. Since evaluating surrogate models is cheap, evaluating a dense grid is not computationally demanding. Furthermore, this method requires the construction of the intermediate surrogate models, which can have a considerable cost. All other methods do not require this, i.e. FLOLA-Voronoi only needs the simulator responses. To conclude, also random sampling has been included in the experiments.

All problems start with a small initial design. Sequentially, samples are added while intermediate models are constructed to evaluate the accuracy that can be obtained with the current set of samples. This iterative process continues until a target accuracy is of 0.05 is reached for the Root-Relative-Square Error (RRSE) (as

defined in Section 2.6) on a dense pre-evaluated validation set. The same metric is used to optimise the hyperparameters of the models. Note that this configuration is atypical for a real surrogate modelling application: usually, no dense validation set is available and a quality estimator such as cross-validation is used. In these experiments, however, we wish to assess which algorithm is fastest to provide the model with a set of highly relevant training inputs, in case we are able to optimally determine the hyperparameters. This does not imply worse performance is expected when using for instance cross-validation: in fact, the additional focus on non-linear areas implies those areas will gain impact during the hyperparameter optimisation as the training data of these areas are more likely to appear in the cross-validation folds. This causes the optimisation to be pushed towards settings which provide good performance in these difficult regions.

The model type for each problem was chosen based on prior knowledge about the test cases. Often this information is not available, in these cases automatic model type selection approaches can be used as described in [26].

4.7.1 Low-dimensional test cases

The goal of the low-dimensional experiments is to illustrate the equivalence of LOLA and FLOLA. As our goal is to evaluate the performance of sequential design strategies, a minimal initial design consisting of a Latin hypercube of 10 points combined with a 2-level factorial design was used, hence $N_{\text{init}} = 14$. Each iteration, a single point is added to this set ($N_{\text{new}} = 1$). For all test cases, each experiment was repeated ten times to reduce noise by random factors in the SUMO toolbox (for example randomisation in the hyperparameter optimisation process). A visual representation of each test case is given in Figure 4.7.

4.7.1.1 Case 1: Peaks

The first test case is a two-dimensional problem known as Peaks. The surface is flat, with a few Gaussian distributions in the center of the domain. This function is very useful to illustrate the concept of LOLA- and FLOLA-Voronoi: as a large part of the input domain is flat, an increased focus on the non-linear region will result in fewer samples required to reach the target accuracy. Kriging with Gaussian correlation function was used as surrogate model type: due to the nature of this model type it is very suitable to model the Gaussian distributions.

Three cases on different domains are considered: $[-3, 3]^2$, $[-5, 5]^2$ and $[-8, 8]^2$. The first case is zoomed in on the non-linear region. As the input range grows, the

quasi-flat surface surrounding the non-linear central region grows, and LOLA- and FLOLA-Voronoi are expected to be more efficient.

4.7.1.2 Case 2: Ackley

Ackley's path, a function well known from optimisation is used as a second test case. For a d-dimensional problem, it is defined as:

$$f(\boldsymbol{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right) + 20 + e$$

with $x_i \in [-2, 2]$. The function is modeled in two dimensions with RBF.

4.7.1.3 Case 3: Low-noise amplifier

This test case consists of a real world problem from electronics. A Low-noise amplifier (LNA), which is a simple Radio-Frequency Circuit (RFC), is the typical first stage of a receiver, providing the gain to suppress noise of subsequent stages. The performance of an LNA can be determined by means of computer simulations where the underlying physical behaviour is taken into account. For this experiment we chose to model the input noise-current $\sqrt{\overline{i_{in}^2}}$, in function of two (normalised) parameters: the inductance L_{sn} and the Metal–Oxide–Semiconductor Field-Effect Transistor (MOSFET) width W_n. The relation to the real parameters is defined as:

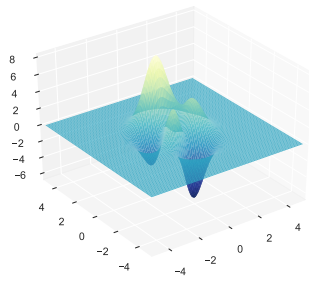
$$W = 100 \cdot 10^{-6} \cdot 10^{W_n} \text{ m}$$

$$L_s = 0.1 \cdot 10^{-9} \cdot 10^{L_{sn}} \text{ H}$$

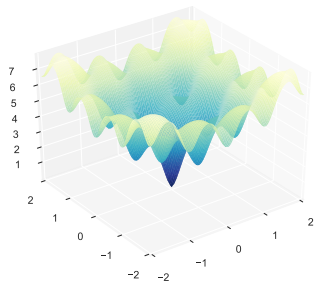
The input domain of this test case is smooth with a steep ridge for $W_n = 0$. A full description of the LNA problem can be found in [27]. The chosen model type for this problem is Artificial Neural Network (ANN), trained with Levenberg-Marquardt backpropagation with Bayesian regularisation (300 epochs). The network topology and initial weights are optimised by a Genetic Algorithm (GA).

4.7.1.4 Results

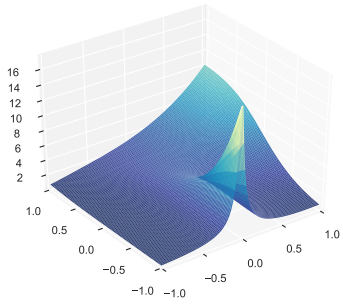
Results of the low-dimensional test cases are shown in Table 4.2 and Figure 4.8, and a detailed overview of the evolution of the accuracy is illustrated in Figure 4.9. For the Peaks test cases, the results confirm the expectation: FLOLA- and LOLA-Voronoi clearly perform better compared to the other methods in all three cases. As the quasi-flat region surrounding the non-linear central region grows, the advantage



(a) Peaks



(b) Ackley (Zoom)



(c) LNA

Figure 4.7: Low-dimensional test cases: Illustrations

Table 4.2: Low-dimensional test cases: summary of the results for the two-dimensional test cases. Each problem was modeled with different sampling strategies until a RRSE of 0.05 over a pre-evaluated validation set was reached. Each experiment ran ten times to cancel out noise by random factors. The 95% confidence intervals are shown between brackets.

	Peaks $[-3, 3]$	Peaks $[-5, 5]$	Peaks $[-8, 8]$	Ackley	LNA
FLOLA-Voronoi	71 (67, 73)	99 (93, 105)	**145** (140, 149)	259 (244, 273)	63 (58, 68)
LOLA-Voronoi	**65** (61, 68)	**97** (87, 105)	147 (142, 153)	262 (244, 278)	**62** (56, 69)
DHASD	79 (75, 83)	115 (106, 123)	192 (179, 205)	352 (326, 379)	88 (79, 97)
Voronoi	96 (93, 98)	229 (219, 238)	643 (589, 698)	**243** (233, 251)	103 (95, 112)
Model Error	104 (98, 110)	281 (253, 308)	851 (768, 934)	262 (250, 273)	118 (105, 132)
Random	105 (94, 116)	270 (240, 300)	1042 (842, 1242)	430 (400, 458)	165 (136, 194)

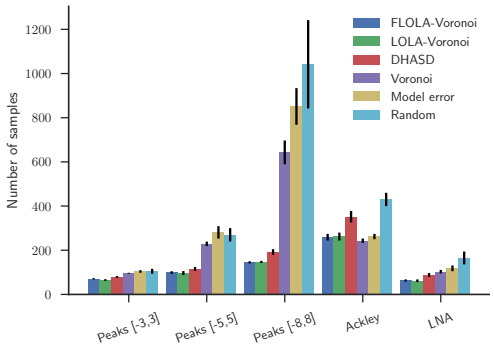


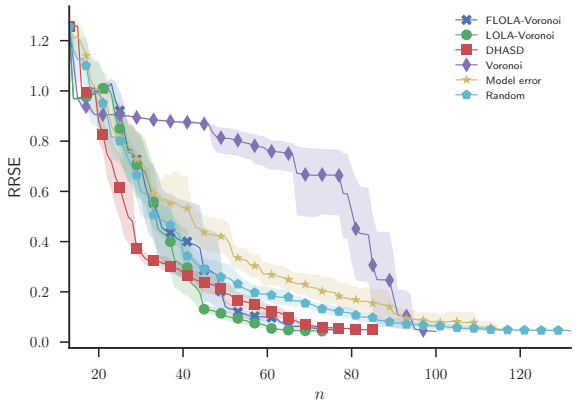
Figure 4.8: Low-dimensional test cases: visual summary of the results listed in Table 4.2.

over the other method increases. This observation is confirmed by the LNA test case: the steep ridge is sampled much more densely by both methods, which leads to a satisfying model with 40% less samples than required for the next best method.

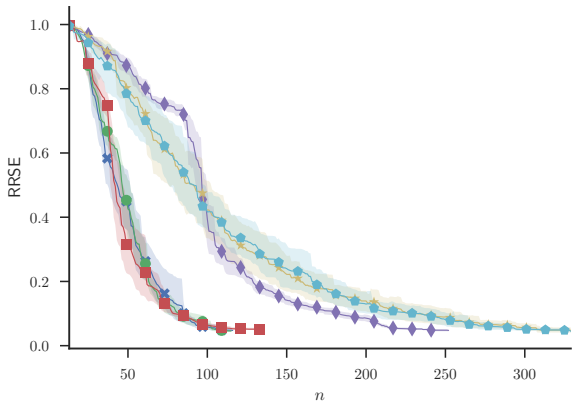
Only the Ackley test case behaves somewhat differently: FLOLA- and LOLA-Voronoi and Model Error result in very comparable results, but the pure exploration method (Voronoi) performs slightly better for this test case. This is not unexpected: the Ackley function is non-linear over the entire interval. There is no benefit of balancing between exploration and exploitation as the sample density should be more or less the same. The exploitation scores do not provide an advantage and sometimes influence the strategy to pick a sample which is not in the largest Voronoi cell. The negative impact of the exploitation score for this test case is quite limited, and can be further reduced by improving the selection of the new candidate in the highest ranked cell.

DHASD is a recent approach, and was presented as alternative for LOLA-Voronoi with the ability to generate better designs. In our study the method performs quite average. Possibly, better results can be obtained by adjusting the parameters of the method to result in better balancing between exploration and exploitation: since there is no automatic way to do this it is a serious disadvantage of the method, especially when nothing is known about the system in advance. Another possible cause for the performance of DHASD could be related to the combination with different surrogate model types: the method has only been tested in combination with Kriging [25]. However, the Peaks problem was modeled with Kriging and DHASD requires more samples.

Although the FLOLA algorithm is less complex and lifts the strict constraints on neighbourhoods of the LOLA algorithm, the low-dimensional experiments indicate

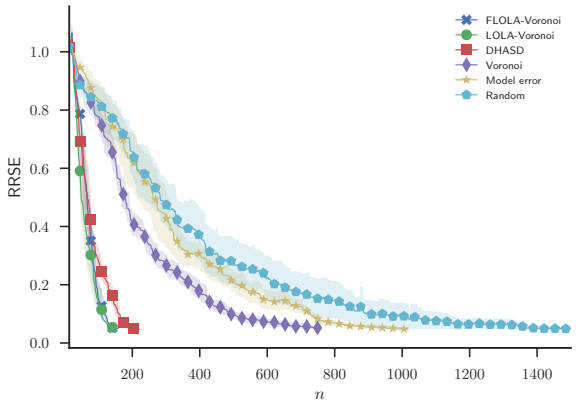


(a) Peaks $[-3, 3]$

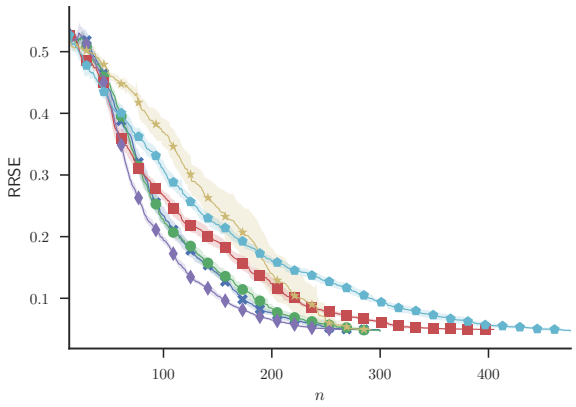


(b) Peaks $[-5, 5]$

Figure 4.9: Low-dimensional test cases: evolution of the RRSE.

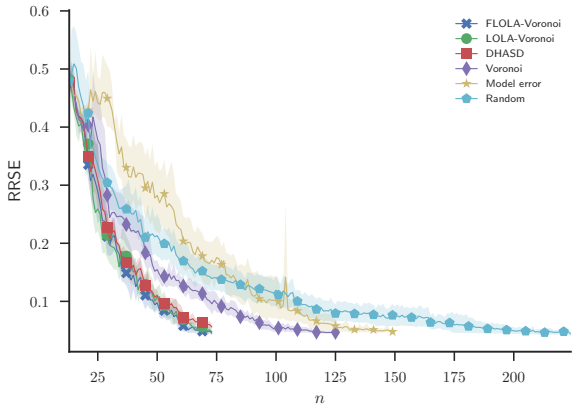


(c) Peaks $[-8, 8]$



(d) Ackley

Figure 4.9: Low-dimensional test cases: evolution of the RRSE.



(e) LNA

Figure 4.9: Low-dimensional test cases: evolution of the RRSE.

the capability of the new algorithm to produce comparable results in terms of number of samples required to reach a predefined target accuracy.

4.7.2 High-dimensional test cases

To illustrate the speed of the new algorithm, two high-dimensional test cases are considered. Global surrogate modelling of high-dimensional problems is not easy, due to the curse of dimensionality. Adding samples one by one and reconstructing the models would be a very lengthy process. To avoid this, samples are added in batches. After each modelling iteration, the sample selection strategy is run once, and a batch of new candidates is selected for evaluation ($N_{\text{new}} > 1$). Many model types are not able to reach a very strict accuracy for high-dimensional problems: at some point their accuracy will not improve much, and the exact location of the samples does not have a big impact. Futhermore, due to the maximum walltime for jobs on the UGent High Performance Computing (HPC) infrastructure, each run was given a time limit of 72 hours which may be too short for some strategies to obtain the target accuracy (RRSE of 0.05). To obtain a better comparison between the sequential design strategies, the amount of samples required for a RRSE of 0.1 is included as well. To evaluate the performance, the running time of the the LOLA, FLOLA and Voronoi components are recorded separately each time the respective algorithm is run. The DHASD method was excluded from the high-dimensional test cases; Delaunay triangulation of high-dimensional datasets with many points is an infeasible lengthy process (as opposed to the Monte Carlo (MC)

approach to estimate the size of the Voronoi cells). Additionally, figuring out the parameters of DHASD by trial-and-error is difficult. Because the input spaces are high-dimensional, the experiments with FLOLA-Voronoi were performed with both Euclidean (l_2) and fractional ($l_{1/d}$) distance to study the impact of the distance metrics. The LOLA-Voronoi algorithm was not modified to use fractional distances, as many optimisations of the LOLA algorithm rely on the use of Euclidean distance. As for the low-dimensional test-cases, each experiment was repeated ten times to reduce noise by random factors in the SUMO toolbox.

4.7.2.1 Case 1: Hartmann

As a first test case, a six-dimensional Hartmann function was chosen. The function is not very complex so it can be modeled in reasonable time, but it does feature some areas that are more difficult to model. The initial design was generated using a Monte Carlo approach as described in [16] with $N_{\text{init}} = 400$. The sample selection batch size was set to 50 samples. As model type, Least-Squares Support Vector Machine (LS-SVM) [28] were chosen as they have a fixed number of 2 parameters: one kernel parameter and the amount of noise. These parameters were optimised with the Dividing Rectangles (DIRECT) algorithm [29].

4.7.2.2 Case 2: Styblinski-Tang

The Styblinski-Tang function is a test function from optimisation. In d dimensions, it is defined as

$$f(\boldsymbol{x}) = \frac{\sum_{i=1}^{d} x_i^4 - 16x_i^2 + 5x_i}{2}$$

for $-5 \leq x_i \leq 5$. The central region is quite flat, but towards the bounds of the interval the function is suddenly steep. It is expected usage of LOLA- and FLOLA-Voronoi will be advantageous as the bounds will be sampled denser. A 2-level factorial design complemented with a Latin hypercube of 244 points generated by the Translational Propagation Latin Hypercube Design (TPLHD) algorithm [30] was used as initial design of size $N_{\text{init}} = 500$. Each iteration 50 samples are added by the sequential design strategy. The chosen model type was again LS-SVM optimised with the DIRECT algorithm.

4.7.2.3 Results

The number of samples required to reach the target accuracy is shown in Table 4.3 and Figure 4.10, and a detailed overview of the accuracy evolution is shown in Figure 4.12. For the Hartmann test case, FLOLA- and LOLA-Voronoi clearly

Table 4.3: High-dimensional test cases: summary of the required sample size to reach two different target accuracies. Each experiment ran ten times to cancel out noise by random factors. The 95% confidence intervals are shown between brackets. In (b), some runs did not finish due to a time constraint of 72 hours. In those cases, the amount of selected samples is shown and the average accuracy at that point.

(a) RRSE of 0.1 on Validation Set

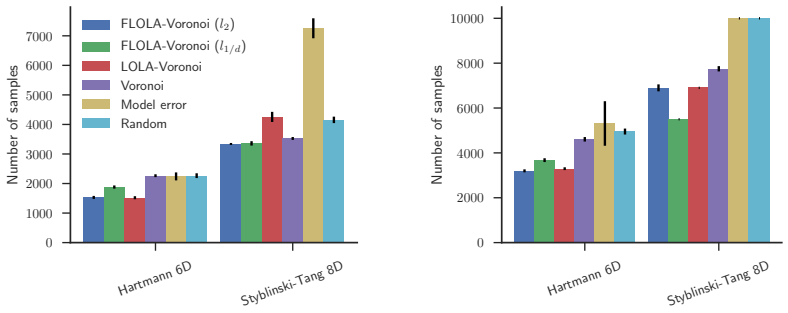
	Hartmann 6D	Styblinski-Tang 8D
FLOLA-Voronoi (l_2)	1530 (1479, 1582)	**3340** (3307, 3374)
FLOLA-Voronoi ($l_{1/d}$)	1881 (1827, 1935)	3359 (3285, 3433)
LOLA-Voronoi	**1520** (1469, 1572)	4255 (4084, 4426)
Voronoi (l_2)	2265 (2217, 2313)	3525 (3476, 3573)
Model error	2241 (2106, 2376)	7255 (6917, 7593)
Random	2266 (2185, 2347)	4155 (4046, 4264)

(b) RRSE of 0.05 on Validation Set

	Hartmann 6D	Styblinski-Tang 8D
FLOLA-Voronoi (l_2)	**3200** (3134, 3268)	**6899** (6744, 7054)
FLOLA-Voronoi ($l_{1/d}$)	3676 (3591, 3761)	> 5500(RRSE ≈ 0.08)
LOLA-Voronoi	3296 (3234, 3358)	> 6900(RRSE ≈ 0.07)
Voronoi (l_2)	4606 (4506, 4706)	7750 (7632, 7868)
Model error	5311 (4318, 6304)	> 10000(RRSE ≈ 0.07)
Random	4951 (4818, 5084)	> 10000(RRSE ≈ 0.06)

outperform the other methods. This confirms that the results of the low-dimensional experiments hold for higher dimensional problems. Surprisingly, Model Error sampling performs worse than random sampling for this test case. Figure 4.11a indicates the average runtime for each of the LOLA, FLOLA and Voronoi components over the 10 runs for both distance metrics, in function of the evaluated samples available before the sampling iteration. For Euclidean distance, the new algorithm is a lot faster compared to LOLA. When selecting new samples with 3000 evaluated samples, computing the scores with LOLA takes 15 minutes, compared to only a few seconds with FLOLA. This is also reflected in the total runtime of the experiment: on average, a run with FLOLA-Voronoi takes 3 hours to complete, compared to 13 hours with LOLA-Voronoi! When using fractional distances, computing distance matrices becomes a lot more expensive because of the n-th root. The impact on FLOLA is limited, but the performance of the Voronoi approximation is heavily affected when using fractional distances and becomes very slow.

For the eight-dimensional Styblinski-Tang function not a single run completed using the LOLA-Voronoi, Model Error or random sampling algorithms. For these algorithms, the strict target accuracy (RRSE = 0.05) was never reached within the time constraint. Runs with LOLA-Voronoi were ended after 72 hours with approximately 6900 samples selected, a large part of the time was spent on sample selection. The average accuracy of runs with LOLA-Voronoi at this point was 0.07. Model Error managed to select up to 10000 points. Unfortunately, this was not

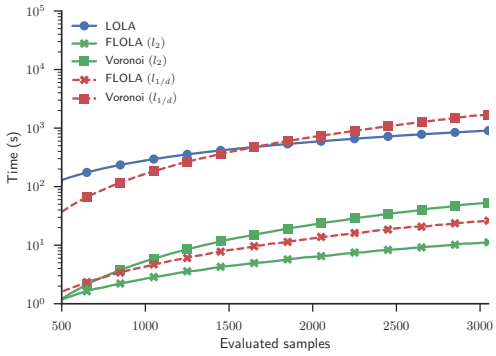


(a) RRSE of 0.1 on Validation Set *(b) RRSE of 0.05 on Validation Set*

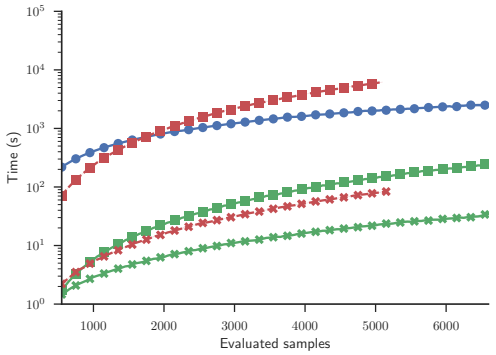
Figure 4.10: High-dimensional test cases: visual summary of the results listed in Table 4.3. The incomplete runs for the Stylinski-Tang 8D experiments due to the time limit are represented by striped bars.

sufficient to reach the target accuracy. FLOLA-Voronoi with fractional distances also failed to reach the target accuracy, mainly due to the poor performance of the Voronoi component with fractional distances. Only (Euclidean) Voronoi sampling and FLOLA-Voronoi managed to reach the target accuracy in time, the latter using 10 percent less samples.

FLOLA-Voronoi with Euclidean distance is the most efficient method to reach the target accuracy for both problems. It is a lot faster compared to LOLA-Voronoi, and requires less samples compared to all other methods. The usage of fractional distance slows the algorithm down and does not seem to provide a benefit. However, the fractional distance does seem to have a slight impact when modelling the eight-dimensional test case. Figure 4.12 shows the evolution of the RRSE as more samples are added each iteration. Clearly, FLOLA-Voronoi with fractional distance brings down the error faster during the beginning of the runs for Styblinski-Tang as shown in Figure 4.12b, which means initially the space is covered better. However, when about 2000 samples have been selected, the version based on Euclidean distance has caught up and as the process evolves, it obtains higher accuracy faster. At this point, the input space has been saturated up to a level which prevents fractional distances from being better at covering the input space. This effect is likely to be more present for problems of higher dimensionality: in this case the usage of FLOLA-Voronoi with fractional distances may be appropriate to obtain a qualitative model faster when samples are very expensive and the additional runtime of the algorithm is not an issue. Furthermore, the computation of the fractional distance matrix can benefit greatly from Graphical Processing Unit (GPU) computation. An implementation in CUDA [31] runs approximately 5x faster than the optimised Central Processing Unit (CPU) implementations used for these experiments.

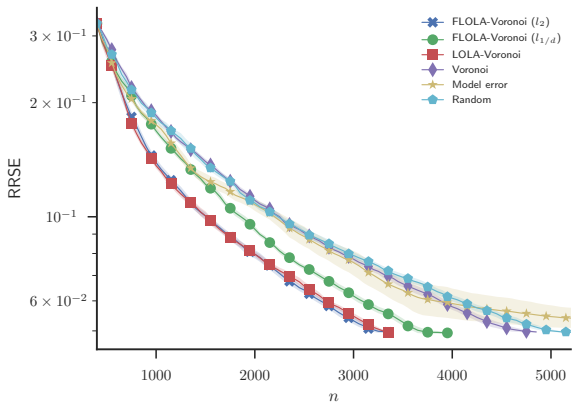


(a) Hartmann 6D

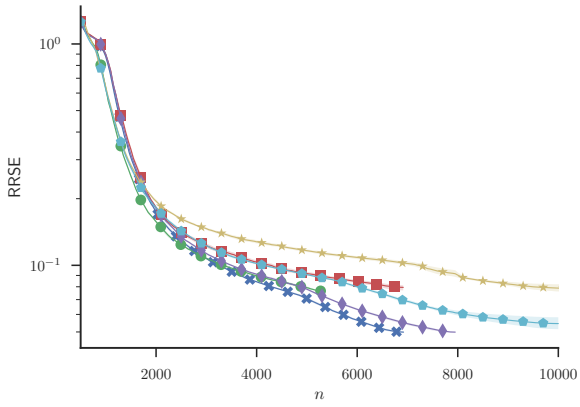


(b) Styblinski-Tang 8D

Figure 4.11: High-dimensional test cases: runtime of LOLA, FLOLA and Voronoi for the high-dimensional test cases. For the Styblinski-Tang function, the experiments of FLOLA-Voronoi with fractional distances were interrupted because the time limit was reached. Confidence intervals are included in the plot but very small.



(a) Hartmann 6D



(b) Styblinski-Tang 8D

Figure 4.12: High-dimensional test cases: evolution of the RRSE. Confidence intervals are included in the plot but very small.

4.8 Conclusion

The LOLA-Voronoi method has proven previously to outperform other sequential design methods for several model types and problems. It does not require intermediate models to make sampling decisions and has been applied to multiple real-world test cases from different problem domains by users of the SUMO Toolbox in several studies [6–12]. The performance of this method comes at the cost of computational complexity, which grows rapidly as the dimensionality of the problem increases.

This chapter presents a novel approach which replaces the computationally complex LOLA algorithm with a Fuzzy variant: FLOLA. Experiments show similar results indicating the new approach has the benefits of the original algorithm, but as it has a complexity of $\mathcal{O}(N)$ the overall time to build a global surrogate model of a high-dimensional problem reduces dramatically. Recent developments such as adaptive balancing of both components [22] are also applicable to this new algorithm.

Currently, new candidate points (samples) are chosen in the design space near the highest ranked samples, based on the maximin distance to existing samples. Better options to improve this local space-fillingness will be investigated in further work. The reduced complexity allows for the construction of global surrogate models of problems with higher dimensionality. In this study, the use of the fractional distances did not offer a lot of advantages. Only in an eight-dimensional problem, a slight advantage in the beginning of the modelling process was noticed. A more thorough study of the impact of high-dimensional spaces on aspects of the surrogate modelling process based on Euclidean distance (model types and measures) is subject of further research.

4.9 References

[1] J. van der Herten, I. Couckuyt, D. Deschrijver, and T. Dhaene. "A Fuzzy Hybrid Sequential Design Strategy for Global Surrogate Modeling of High-Dimensional Computer Experiments". *SIAM Journal on Scientific Computing* 37.2 (2015), A1020–A1039.

[2] D. Gorissen, W. Hendrickx, K. Crombecq, and T. Dhaene. "Adaptive Distributed Metamodeling". *Proceedings of 7th International Meeting on High Performance Computing for Computational Science (VECPAR 2006), Lecture Notes in Computer Science*. Ed. by M. Dayde et al. Vol. 4395. Rio de Janeiro, Brazil: Springer, June 2007, pp. 579–588.

[3] K. Goethals, I. Couckuyt, T. Dhaene, and A. Janssens. "Sensitivity of night cooling performance to room/system design: Surrogate models based on CFD". *Building and Environment* 58 (2012), pp. 23–36.

[4] K. Crombecq, D. Gorissen, L. D. Tommasi, and T. Dhaene. "A Novel Sequential Design Strategy for Global Surrogate Modeling". *Proceedings of the 42nd Conference on Winter Simulation*. Ed. by M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls. Austin, TX, USA: Institute of Electrical and Electronics Engineers, Inc, Dec. 2009, pp. 731–742.

[5] K. Crombecq, D. Gorissen, D. Deschrijver, and T. Dhaene. "A Novel Hybrid Sequential Design Strategy for Global Surrogate Modelling of Computer Experiments". *SIAM Journal of Scientific Computing* 33.4 (2010), pp. 1948–1974.

[6] J. Aernouts, I. Couckuyt, K. Crombecq, and J. J. Dirckx. "Elastic characterization of membranes with a complex shape using point indentation measurements and inverse modelling". *International Journal of Engineering Science* 48.6 (2010), pp. 599–611.

[7] D. Deschrijver, K. Crombecq, H. M. Nguyen, and T. Dhaene. "Adaptive Sampling Algorithm for Macromodeling of Parameterized-Parameter Responses". *Microwave Theory and Techniques, IEEE Transactions on* 59.1 (2011), pp. 39–45.

[8] D. Stephens, D. Gorissen, K. Crombecq, and T. Dhaene. "Surrogate based sensitivity analysis of process equipment". *Applied Mathematical Modelling* 35.4 (2011), pp. 1676–1687.

[9] B. Rosenbaum and V. Schulz. "Comparing sampling strategies for aerodynamic Kriging surrogate models". *ZAMM-Journal of Applied Mathematics*

and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik 92.11-12 (2012), pp. 852–868.

[10] S. Aerts, D. Deschrijver, W. Joseph, L. Verloock, F. Goeminne, L. Martens, and T. Dhaene. "Exposure assessment of mobile phone base station radiation in an outdoor environment using sequential surrogate modeling". *Bioelectromagnetics* (2013).

[11] D. Deschrijver, F. Vanhee, D. Pissoort, and T. Dhaene. "Automated near-field scanning algorithm for the EMC analysis of electronic devices". *Electromagnetic Compatibility, IEEE Transactions on* 54.3 (2012), pp. 502–510.

[12] P. Singh, D. Deschrijver, D. Pissoort, and T. Dhaene. "Adaptive classification algorithm for EMC-compliance testing of electronic devices". *Electronics Letters* 49.24 (2013), pp. 1526–1528.

[13] T. Simpson, D. Lin, and W. Chen. "Sampling Strategies for Computer Experiments: Design and Analysis". *International Journal of Reliability and Application* 2.3 (2002), pp. 209–240.

[14] T. Simpson, J. D. Poplinski, P. N. Koch, and J. K. Allen. "Metamodels for Computer-based Engineering Design: Survey and recommendations." *Eng. Comput. (Lond.)* 17.2 (2001), pp. 129–150.

[15] D. Gorissen, I. Couckuyt, E. Laermans, and T. Dhaene. "Multiobjective global surrogate modeling, dealing with the 5-percent problem". *Engineering with Computers* 26.1 (Jan. 2010), pp. 81–89.

[16] K. Crombecq, I. Couckuyt, D. Gorissen, and T. Dhaene. "Space-filling Sequential Design Strategies for Adaptive Surrogate Modelling". *Proceedings of the First International Conference on Soft Computing Technology in Civil, Structural and Environmental Engineerin*. Ed. by B. H. Topping and Y. Tsompanakis. Vol. 92. Stirlingshire, UK: Civil-Comp Press, 2009, Paper–50.

[17] P. de Oliveira Castro, E. Petit, J. C. Beyler, and W. Jalby. "ASK: adaptive sampling kit for performance characterization". *Euro-Par 2012 Parallel Processing*. Springer, 2012, pp. 89–101.

[18] Y. Lin. "An Efficient Robust Concept Exploration Method and Sequential Exploratory Experimental Design". PhD thesis. Georgia Institute of Technology, 2004.

[19] H. T. Croft, K. J. Falconer, and R. K. Guy. *Unsolved problems in geometry*. Springer-Verlag New York, 1991. ISBN: 978-1-4612-0963-8.

[20] E. H. Mamdani and S. Assilian. "An experiment in linguistic synthesis with a fuzzy logic controller". *International journal of man-machine studies* 7.1 (1975), pp. 1–13.

[21] R. Fullér. *Neural fuzzy systems*. Åbo Akademi, 1995. ISBN: 951-650-624-0.

[22] P. Singh, D. Deschrijver, and T. Dhaene. "Balanced Sequential Design Strategy for Global Surrogate Modeling". *Proceedings of the 46th Conference on Winter Simulation*. Austin, TX, USA: Institute of Electrical and Electronics Engineers, Inc, Dec. 2013.

[23] D. Francois, V. Wertz, and M. Verleysen. "The concentration of fractional distances". *Knowledge and Data Engineering, IEEE Transactions on* 19.7 (2007), pp. 873–886.

[24] A. Kabán. "Non-parametric detection of meaningless distances in high dimensional data". *Statistics and Computing* 22.2 (2012), pp. 375–385.

[25] A. Ajdari and H. Mahlooji. "An adaptive exploration-exploitation algorithm for constructing metamodels in random simulation using a novel sequential experimental design". *Communications in Statistics-Simulation and Computation* 43.5 (2014), pp. 947–968.

[26] I. Couckuyt, D. Gorissen, F. De Turck, and T. Dhaene. "Automatic surrogate model type selection during the optimization of expensive black-box problems". *Proceedings of the 44th Conference on Winter Simulation*. Phoenix, AZ, USA: Institute of Electrical and Electronics Engineers, Inc, Dec. 2011, pp. 4274–4284.

[27] D. Gorissen, L. De Tommasi, K. Crombecq, and T. Dhaene. "Sequential Modeling of a Low Noise Amplifier with Neural Networks and Active Learning". *Neural Computing and Applications* 18.5 (June 2009), pp. 485–494.

[28] J. A. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, J. Suykens, and T. Van Gestel. *Least squares support vector machines*. Vol. 4. World Scientific, 2002.

[29] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. "Lipschitzian optimization without the Lipschitz constant". *Journal of Optimization Theory and Applications* 79.1 (1993), pp. 157–181.

[30] F. A. Viana, G. Venter, and V. Balabanov. "An algorithm for fast optimal Latin hypercube design of experiments". *International journal for numerical methods in engineering* 82.2 (2010), pp. 135–156.

[31] C. Nvidia. *Programming guide*. 2008.

"That's the old passage to Ravenholm. We don't go there anymore. Believe me, that tunnel is sealed for a reason."

— Alyx Vance

5 Deterministic Knowledge-Gradient

5.1 Introduction

As building many prototypes and performing real-life experiments is costly, engineers have adopted the concepts of virtual prototyping and Computer Aided Design (CAD) since long. After specifying a set of values for the input parameters (design of experiments), multiple experiments on the complex input-output systems are performed virtually by means of computer simulations resulting in cost savings and a shorter time-to-market.

Over the years, the accuracy of the available simulation software has improved significantly allowing simulation of systems at a finer level of detail. This evolution opens up usage of simulations for increasingly complex problems, but also increases the associated computational cost tremendously. Some high-fidelity simulations are known to require days or even weeks of runtime for a single evaluation [1]. This makes their use infeasible for evaluation-intensive analysis such as parameter exploration, sensitivity analysis or optimisation. This sparked the development of surrogate modelling or metamodelling: essentially these are predictive models used specifically to approximate the behavioural response of engineering systems. The simulator responses should be approximated accurately using a minimum number

of evaluations, while still keeping the computational cost acceptable [2].

Surrogate models can either be used as a global approximation that can replace the simulator, but can also be used to guide an optimisation process. Most known is the Efficient Global Optimisation (EGO) methodology [3, 4] which sequentially picks the next evaluation by optimizing the Expected Improvement (EI) policy using Kriging models under the assumption that the response of the simulation is deterministic: the output of the simulation is considered to be noise-free, the only form of noise encountered is negligibly small, e.g., in the order of the machine epsilon. The EI policy essentially observes the difference between the expectancy on the prediction mean of the next intermediate model given an arbitrary unobserved evaluation and the current best observed value. It has been shown that the concept can be extended to multi-objective optimisation problems by aggregating the responses of several objectives [5, 6] or by allowing different interpretations of improvement [7, 8]. This approach has been applied successfully for optimisation of several engineering applications and quickly leads to satisfying results. Because the objective function is expensive to compute, spending some computation time to decide the next evaluation intelligently is justified. The EI policy, amongst others, can be combined with different (Bayesian) models to optimize any real-valued (typically expensive) objective function: this is often referred to as Bayesian Optimisation (BO) which has lately become increasingly popular for hyperparameter optimisation [9–11].

Different policies for Bayesian optimisation exist, such as the Upper Confidence Bound (UCB) proposed by Cox and John [12]. Although the concept of the UCB policy is quite straightforward, it features some strong theoretical guarantees [13]. A more recent information-based policy introduced by Hernández-Lobato et al. [14] for Bayesian optimisation under the presence of uncertainty is Predictive Entropy Search (PES). On a set of problems, it was shown to perform comparable or better than standard EI. A different policy for Bayesian optimisation with a discrete input domain in the presence of uncertainty on the obtained response is the Knowledge-Gradient with Correlated Beliefs (KGCB) [15]. An approximation known as the Knowledge-Gradient for Continuous Parameters (KGCP) by Scott et al. [16] enables application of the knowledge-gradient to problems with a continuous input domain, without the need to discretize the input domain first. The relationship to, and differences with EI have been discussed in [17], although sometimes the two policies are mixed up [18]. Results obtained on problems involving uncertain responses (i.e., noise) pointed out an advantage of Knowledge-Gradient for Continuous Parameters (KGCP) over EGO and Sequential Kriging Optimisation (SKO) [19, 20] which both involve the EI policy, the latter including a correction term to account for the belief that the unknown next point to be evaluated also has noise associated with it. However for deterministic problems (as often encountered in physics-based

engineering simulation) the additional complexity of computing the KGCP has been disproportional to the advantage in terms of evaluations.

In short, this chapter addresses the following:

- A novel closed form for computation of the KGCP for deterministic problems is derived, and it is shown that the KGCP has now similar (computational) complexity as EI.
- The relationship between EI and KGCP is studied. From the formulation it can be observed the KGCP has more confidence in the underlying intermediate model, in comparison to EI.
- The KGCP is compared with EI, UCB and PES on several deterministic functions and a real-life 10D structural dynamics optimisation problem from engineering.
- The use of slice sampling and Maximum Likelihood Estimation (MLE) for selecting the hyperparameters of the underlying Kriging model is compared empirically on all test problems.

Although most recent research on Bayesian optimisation focusses on stochastic optimisation, deterministic problems arise frequently in various research application domains and are frequently solved efficiently by (Bayesian) machine learning methods. Machine learning-based engineering is a prime example. Here expensive deterministic simulations are optimized using machine learning methods. The EI policy has long been a popular choice for this task. Optimisation of deterministic engineering problems is the point of view of this chapter, though our approach to compute the KGCP is general and can be applied to any (expensive) deterministic optimisation problem.

Section 5.2 addresses the optimisation problem formally and introduces the model used by both policies in this contribution. Section 5.3 reviews the knowledge-gradient and develops a closed form to compute the KGCP for deterministic problems. In Section 5.4 the KGCP is compared to EI, UCB and PES on several deterministic problems.

5.2 Formalism

Given the following global optimisation problem:

$$\arg\max_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}), \tag{5.1}$$

for an unknown function $f : \mathcal{X} \to \mathbb{R}$, with $\mathcal{X} \subset \mathbb{R}^d$ corresponding to the simulator. The vector $\boldsymbol{x}$ is referred to as *decision* and is part of the set of feasible decisions $\mathcal{X} \subset \mathbb{R}^d$. We can obtain the observation y_i by evaluating f for a decision $\boldsymbol{x}_i$. Assuming the observation is not exact and subject to uncertainty, the distribution of y_i is centred around the true response with variance given by σ_n^2. We focus on deterministic problems, which implies $\sigma_n^2 = 0$ and denote the observations by f_i hereafter. The goal is to decide on a sequential set of decisions $\boldsymbol{x}_i$ using a sampling policy to identify an optimal solution of Equation (5.1) using only a limited number of N_{max} observations. Because evaluating f is expensive, additional computational effort to determine these decisions is justified.

5.2.1 Kriging interpolation

We proceed by modelling the obtained information on f with a Kriging model [21]. Given $n < N_{\text{max}}$ observations f_i corresponding to decisions $\{\boldsymbol{x}_0, ..., \boldsymbol{x}_{n-1}\}$, we construct a model with prediction mean and variance μ_n and s_n^2 respectively, which combines a regression model and a Gaussian Process (GP) with $\gamma(\boldsymbol{x}) = 0, \forall \boldsymbol{x}$, variance σ^2 and correlation matrix $\boldsymbol{R}$ interpolating the residual.

Given a set of basis functions $\{ b_1, ..., b_p \}$ and a correlation function r, $\boldsymbol{B}$ and $\boldsymbol{R}$ represent the regression and correlation matrix respectively:

$$\boldsymbol{B} = \begin{bmatrix} b_1(\boldsymbol{x}_0) & \dots & b_p(\boldsymbol{x}_0) \\ \vdots & \ddots & \vdots \\ b_1(\boldsymbol{x}_{n-1}) & \dots & b_p(\boldsymbol{x}_{n-1}) \end{bmatrix},$$

$$\boldsymbol{R} = \begin{bmatrix} r(\boldsymbol{x}_0, \boldsymbol{x}_0) & \dots & r(\boldsymbol{x}_0, \boldsymbol{x}_{n-1}) \\ \vdots & \ddots & \vdots \\ r(\boldsymbol{x}_{n-1}, \boldsymbol{x}_0) & \dots & r(\boldsymbol{x}_{n-1}, \boldsymbol{x}_{n-1}) \end{bmatrix}.$$

Denoting the regression matrix for an arbitrary point $\boldsymbol{x}_\star$ as $\boldsymbol{B}_\star = [b_1(\boldsymbol{x}_\star), ..., b_p(\boldsymbol{x}_\star)]$, whereas $\boldsymbol{R}_{\star f} = \boldsymbol{R}_{f\star}^T = [r(\boldsymbol{x}_\star, \boldsymbol{x}_1), ..., r(\boldsymbol{x}_\star, \boldsymbol{x}_n)]$. The regression coefficients $\boldsymbol{\alpha} \in \mathbb{R}^{n\times 1}$ can be obtained by solving the Generalized Least Squares problem (corresponding to the Moore-Penrose pseudoinverse):

$$\boldsymbol{\alpha} = \left(\boldsymbol{B}^T \boldsymbol{R}^{-1} \boldsymbol{B}\right)^{-1} \boldsymbol{B}^T \boldsymbol{R}^{-1} \boldsymbol{f}.$$

In contrast to GPs, the process variance is computed analytically:

$$\sigma^2 = \frac{1}{n}(\boldsymbol{f} - \boldsymbol{B}\boldsymbol{\alpha})^T \boldsymbol{R}^{-1}(\boldsymbol{f} - \boldsymbol{B}\boldsymbol{\alpha}).$$

For this model, the prediction mean and variance are given by

$$\mu_n(\boldsymbol{x}_\star) = \boldsymbol{B}_\star \boldsymbol{\alpha} + \boldsymbol{R}_{\star f}\boldsymbol{R}^{-1}(\boldsymbol{f} - \boldsymbol{B}\boldsymbol{\alpha}), \tag{5.2a}$$

$$s_n^2(\boldsymbol{x}_\star) = \sigma^2 \left(1 - \boldsymbol{R}_{\star f}\boldsymbol{R}^{-1}\boldsymbol{R}_{f\star} \frac{1 - \boldsymbol{B}^T\boldsymbol{R}^{-1}\boldsymbol{R}_{f\star}}{\boldsymbol{B}^T\boldsymbol{R}^{-1}\boldsymbol{B}}\right). \tag{5.2b}$$

The gradients for prediction mean and variance for the model can also be computed analytically.

Under this formulation the model interpolates the training data. By adding a constant to the diagonal of $\boldsymbol{R}$ the model becomes a regression model and is referred to as Stochastic Kriging in operational research. Note that in machine learning, policies are most often based on a GP with noise. Kriging can be considered a special case of a GP where the regression function coefficients are estimated by Generalized Least Squares and the data interpolated. We chose the Kriging formulation, because it is a common choice in engineering simulation optimisation (as it is the underlying model used in the EGO formulation of Jones et al. [4]).

5.2.2 Basis and correlation functions

Ordinary Kriging is the common choice and includes only the constant regression function: $b_0(\boldsymbol{x}) = 1$. Specifying a correlation function r defines the correlation matrix $\boldsymbol{R}$. We selected the Matérn 5/2 correlation function [22] for engineering problems, because the popular Gaussian correlation function assumes an unrealistic smoothness of the underlying response [10]. The Matérn 5/2 correlation function requires only twice differentiability which fits many engineering problems, including the application tackled in this work. Assuming two decisions $\boldsymbol{x}$ and $\boldsymbol{x}'$ we get

$$r(\boldsymbol{x}, \boldsymbol{x}') = \left(1 + \sqrt{5}l + \frac{5l^2}{3}\right) \exp\left(-5\sqrt{l}\right)$$

$$l = \sqrt{(\boldsymbol{x} - \boldsymbol{x}')^T \operatorname{diag}(\boldsymbol{\theta}) (\boldsymbol{x} - \boldsymbol{x}')}.$$

The choice of the lengthscale hyperparameter vector[1] $\boldsymbol{\theta} \in \mathbb{R}^d$ is crucial to obtain a meaningful prediction. The hyperparameter vector is typically identified using

[1] We consider *anisotropic* correlation functions with a hyperparameter for each dimension. This type of kernels is often referred to as Automatic Relevance Determination (ARD) kernels. In contrast, *isotropic* kernels have $\theta \in \mathbb{R}^1$.

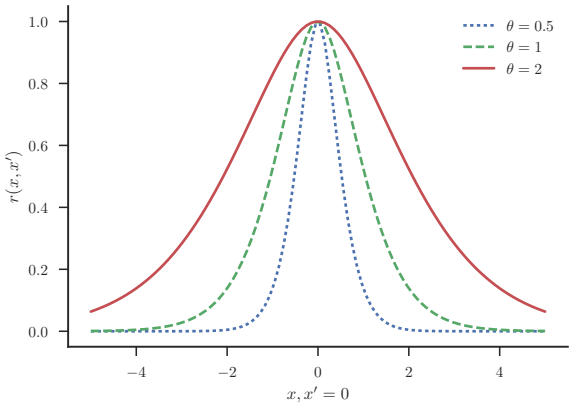


Figure 5.1: Matérn 5/2 correlation function: 1D illustration of the correlation function. The response is shown by varying $\boldsymbol{x}$ while $\boldsymbol{x}'$, for three different hyperparameters.

MLE. Several variants of the likelihood are available, most commonly used is the concentrated log-likelihood:

$$\ln p\left(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{\theta}\right) = \frac{1}{2}\left(n \ln(\sigma^2) + \ln(|\boldsymbol{R}|)\right).$$

However, it was reported the MLE solution can result in a biased prediction variance [23]. As the uncertainty expressed by the prediction variance is crucial in Bayesian optimisation, the uncertainty about the model parameters should be incorporated which cannot be accomplished using a single parameter estimate. Hence, we also consider slice sampling the likelihood as proposed previously by Murray and Adams [24] and Snoek et al. [10] to draw a set of hyperparameter samples $\{\boldsymbol{\theta}_i\}_{i=1}^{h}$. To compute the sampling policies, the policy scores computed for all obtained $\boldsymbol{\theta}$ are averaged, instead of using the averaged prediction mean and variance and computing the policy only once.

5.3 Knowledge-gradient policy

The knowledge-gradient policy was described in Frazier et al. [15] for optimisation over a discrete decision domain $\mathcal{X}$. At iteration n, we have obtained n observations $\{f_0, ...f_{n-1}\}$ corresponding to n decisions $\{\boldsymbol{x}_0, ..., \boldsymbol{x}_{n-1}\}, \boldsymbol{x}_i \in \mathcal{X}$. The information gained from measuring $\boldsymbol{x} \in \mathcal{X}$ is defined as the knowledge-gradient:

$$\alpha_n^{\mathrm{KG}}(\boldsymbol{x}) = \mathbb{E}\left[\max_{\boldsymbol{u}\in\mathcal{X}} \mu_{n+1}(\boldsymbol{u}) \,\middle|\, \boldsymbol{x}_n = \boldsymbol{x}\right] - \max_{\boldsymbol{u}\in\mathcal{X}} \mu_n(\boldsymbol{u}). \tag{5.3}$$

The next sampling decision $\boldsymbol{x}_n$ is chosen as the maximum over the knowledge-gradient,

$$\boldsymbol{x}_n = \arg\max_{\boldsymbol{x}\in\mathcal{X}} \alpha_n^{\text{KG}}(\boldsymbol{x}). \tag{5.4}$$

This corresponds to the knowledge-gradient policy. An algorithm to solve Equation (5.3) was formulated in Frazier et al. [15]. Typically, only a single decision is selected for evaluation (as is also the focus of this chapter), however an extension to select additional decisions in batch was presented recently [25]. This section first reviews the case of approximating the knowledge-gradient in the presence of continuous parameters. Next an explicit formulation in case of deterministic observations is derived.

5.3.1 Knowledge-gradient for continuous parameters

When $\mathcal{X}$ represents a continuous decision domain, Equation (5.3) can no longer be computed. A straightforward solution is discretising $\mathcal{X}$ and handling the problem as a discrete decision problem. However, the computational complexity grows rapidly as the number of feasible decisions grows: this occurs when the dimensionality of the problem is not small, or the ranges of parameters are large.

An approximation method referred to as the KGCP was introduced by Scott et al. [16], avoiding large-scale discretisation of $\mathcal{X}$. Instead of maximising over the entire decision domain, it was shown that including only the past and current sampling decisions is a reasonable approximation:

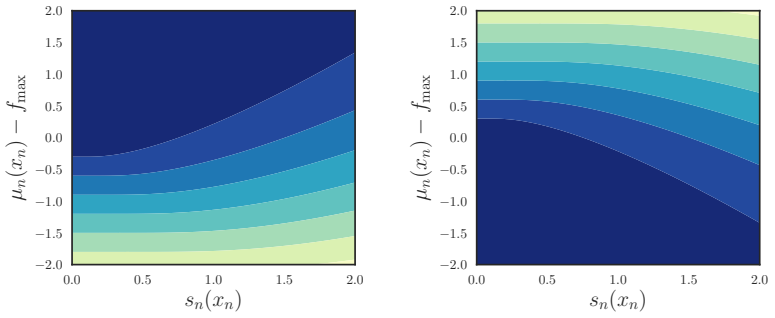
$$\bar{\alpha}_n^{\text{KG}}(\boldsymbol{x}) = \mathbb{E}\left[\max_{i=0,..,n} \mu_{n+1}(\boldsymbol{x}_i) \,\middle|\, \boldsymbol{x}_n = \boldsymbol{x}\right] - \max_{i=0,..,n} \mu_n(\boldsymbol{x}_i)|_{\boldsymbol{x}_n=\boldsymbol{x}}. \tag{5.5}$$

Computing $\bar{\alpha}_n^{\text{KG}}$ is possible using a similar method as originally proposed for the knowledge-gradient for discrete optimisation. The policy for optimal decision making, allowing N_{max} observations of the problem, is summarized in Algorithm 3. Sequentially the maximum of $\bar{\alpha}_n^{\text{KG}}$ is chosen as new decision and the model is improved with the acquired observation. At the end of the sampling process the final model is optimized and the location of global optimum $\hat{\boldsymbol{x}}$ is returned as optimal decision.

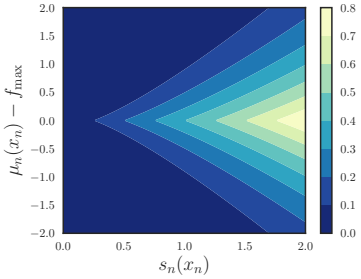
5.3.2 Computing KGCP for deterministic problems

Denoting improvement for maximisation problems as $I_n = \max\left(\mu_{n+1}(\boldsymbol{x}) - f_{\max}, 0\right)$ with $f_{\max} = \max_{i=0,..,n-1}\ f_i$, we can define the EI sampling policy as:

$$\alpha_n^{\text{EI}}(\boldsymbol{x}) = \mathbb{E}\left[I_n\right].$$



(a) Expected Improvement α_n^{EI} *(b) Expected Decrement (Proposition 4)*



(c) $\bar{\alpha}_n^{KG}$ *(Definition 5)*

Figure 5.2: Sampling policies: different policies as a function of prediction variance and improvement. Clearly the KGCP is more conservative and only yields elevated scores if the prediction variance indicates improvement over $f_{\max}$ *could occur.*

Algorithm 3 KGCP Policy

Obtain initial observations for decisions $\boldsymbol{x}_0, ..., \boldsymbol{x}_{N_{\text{init}}-1}$
for $n = N_{\text{init}}$ **to** $N_{\text{max}} - 1$ **do**
  Decide next observation: $\boldsymbol{x}_n = \underset{\boldsymbol{x}\in\mathcal{X}}{\arg\max}\, \bar{\alpha}_n^{\text{KG}}(\boldsymbol{x})$
  Obtain observation f_n for decision $\boldsymbol{x}_n$
  Calculate μ_{n+1} (or update μ_n)
end for
Estimate of optimal decision $\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\mathcal{X}}{\arg\max}\, \mu_{N_{\text{max}}}(\boldsymbol{x})$

It was shown in Scott et al. [16] that

$$\bar{\alpha}_n^{\text{KG}} \leq \alpha_n^{\text{EI}}$$

under the assumption that the observations are deterministic ($\sigma_n^2 = 0$). The latter quantity corresponds to the EI criterion [3, 4] and is used widely as part of EGO in many applications in simulation optimisation [26–28]. EI can be reformulated as follows:

$$\begin{aligned}
\alpha_n^{\text{EI}}(\boldsymbol{x}) &= \mathbb{E}\left[\max\left(\mu_{n+1}(\boldsymbol{x}) - f_{\max}, 0\right)\right], \\
&= \mathbb{E}\left[\max\left(\mu_{n+1}(\boldsymbol{x}), f_{\max}\right)\right] - f_{\max}, \\
&= \mathbb{E}\left[\max\left(\mu_{n+1}(\boldsymbol{x}), \max_{i=0,..,n-1} \mu_n(\boldsymbol{x}_i)\right)\right] - \max_{i=0,..,n-1} \mu_n(\boldsymbol{x}_i), \quad (5.6a) \\
&= \mathbb{E}\left[\max_{i=0,..,n} \mu_{n+1}(\boldsymbol{x}_i) \,\middle|\, \boldsymbol{x}_n = \boldsymbol{x}\right] - \max_{i=0,..,n-1} \mu_n(\boldsymbol{x}_i). \quad (5.6b)
\end{aligned}$$

In Equation (5.6a), we used the property $f_i = \mu_n(\boldsymbol{x}_i) = \mu_{n+1}(\boldsymbol{x}_i), i = 0, .., n-1$ which holds because of the deterministic assumption. The EI has an elegant closed form that is easy to compute. This form is usually given for minimising an objective. Instead, for maximising an objective the closed form is (the argument $\boldsymbol{x}$ is omitted for clarity):

$$\begin{aligned}
\alpha_n^{\text{EI}} &= (\mu_n - f_{\max})\,\Phi(-z) + s_n \phi(z), \\
z &= \frac{f_{\max} - \mu_n}{s_n}.
\end{aligned}$$

Comparing Equations (5.5) and (5.6b), the difference is in the second term: the KGCP includes the model prediction for the current sampling decision whereas EI only includes all previous observations. In order to define the exact relation between KGCP and EI, we infer the inequality case occurring when $\mu_n(\boldsymbol{x}) > f_{\max}$.

Defining this case as the *Expected Decrement (ED)*:

$$\begin{aligned}\mathbb{E}\left[D_n\right] &= \mathbb{E}\left[\max_{i=0,..,n} \mu_{n+1}(\boldsymbol{x}_i) \;\middle|\; \boldsymbol{x}_n = \boldsymbol{x}\right] - \mu_n(\boldsymbol{x}), \\ &= \mathbb{E}\left[\max\left(\mu_{n+1}(\boldsymbol{x}_n) - \mu_n(\boldsymbol{x}_n), f_{\max} - \mu_n(\boldsymbol{x}_n)\right) \mid \boldsymbol{x}_n = \boldsymbol{x}\right].\end{aligned}$$

We can derive an explicit formulation for this quantity as well.

Proposition 4. *Under the assumption $\mu_{n+1}(\boldsymbol{x}_n) \sim \mathcal{N}\left(\mu_n(\boldsymbol{x}_n), s_n^2(\boldsymbol{x}_n)\right)$, computing ED results in an expression of the form of EI for minimization, except for $f_{\max}$ appearing instead of $f_{\min}$ (both in the first term, as well as the arguments z).*

$$\mathbb{E}\left[D_n\right] = \left(f_{\max} - \mu_n\right)\Phi\left(z\right) + s_n\phi\left(z\right). \tag{5.7}$$

Proof *Computation of the closed form of the ED is possible by integrating the expectation. Let $\tilde{F} = \mu_{n+1}(\boldsymbol{x}_n) \sim \mathcal{N}\left(\mu_n(\boldsymbol{x}_n), s_n^2(\boldsymbol{x}_n)\right)$ (omitting $\boldsymbol{x}_n$ for presentation clarity), and Φ and ϕ quantities the standard normal cumulative and probability density functions respectively. We can now split the integration:*

$$\begin{aligned}\mathbb{E}\left[D_n\right] = &\int_{-\infty}^{f_{\max}} \left(f_{\max} - \mu_n\right)\phi\left(\tilde{F} \;\middle|\; \mu_n, s_n^2\right)\mathrm{d}\tilde{F} \\ &+ \int_{f_{\max}}^{\infty} \left(\tilde{F} - \mu_n\right)\phi\left(\tilde{F} \;\middle|\; \mu_n, s_n^2\right)\mathrm{d}\tilde{F}\end{aligned}$$

Substituting $u = \frac{\tilde{F}-\mu_n}{s_n}$:

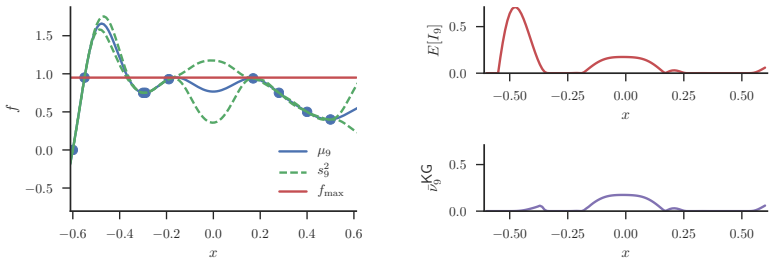
$$\begin{aligned}&= \left(f_{\max} - \mu_n\right)\left[\Phi\left(\frac{\tilde{F} - \mu_n}{s_n}\right)\right]_{\tilde{F}=-\infty}^{\tilde{F}=f_{\max}} + \int_A^{\infty} (s_n u + \mu_n - \mu_n)\phi(u)\,\mathrm{d}u \\ &= \left(f_{\max} - \mu_n\right)\Phi\left(\frac{f_{\max} - \mu_n}{s_n}\right) + s_n\left[-\phi(u)\right]_A^{\infty} \\ &= \left(f_{\max} - \mu_n\right)\Phi\left(\frac{f_{\max} - \mu_n}{s_n}\right) + s_n\phi\left(\frac{f_{\max} - \mu_n}{s_n}\right)\end{aligned}$$

■

Φ and ϕ represent the standard normal cumulative and probability density functions respectively. This allows defining an explicit formula for KGCP.

Definition 5. *Under the assumption of no observation noise ($\sigma_n^2 = 0$) the KGCP is defined as*

$$\bar{\alpha}_n^{KG}(\boldsymbol{x}) = \min\left(\mathbb{E}\left[I_n\right], \mathbb{E}\left[D_n\right]\right).$$



(a) *(b)*

Figure 5.3: Sampling policies: comparison of EI and KGCP criteria for 1D example. In (a) 9 decisions have been observed, and interpolated with a Kriging model. The prediction mean and variance are shown. In (b) the EI and KGCP policies for the same interval are shown, upper and bottom graph respectively. The leftmost area is ignored by KGCP because $\mathbb{E}[D_n] < \mathbb{E}[I_n]$ *due to very low prediction variance. Instead, it focusses on the central region.*

Figure 5.2 illustrates the response of α_n^{EI}, $\mathbb{E}[D_n]$ and $\bar{\alpha}_n^{\text{KG}}$ as a function of $\mu_n(\boldsymbol{x}) - f_{\max}$ and $s_n^2(\boldsymbol{x})$. In the context of a maximisation problem the expected decrement serves a similar purpose as EI for maximisation: it indicates decisions which are believed to lead to worse results compared to the best decision we have observed ($f_{\max}$). The term tends to be smaller than α_n^{EI} in Definition 5 in regions satisfying $\mu_n(\boldsymbol{x}) > f_{\max}$ and small values for $s_n^2(\boldsymbol{x})$. This corresponds to areas of which the model strongly expects improvement upon $f_{\max}$. Because of this notion of *certain improvement* the quantity of $\mathbb{E}[D_n]$ is smaller compared to α_n^{EI}, hence KGCP ends exploitation and focusses on exploration instead.

An illustration of a 1D maximisation problem is given in Figure 5.3: two areas have elevated EI scores; however the most promising area according to EI is assigned a lower KGCP score: although the model indicates the objective function can be improved, $\mathbb{E}[D_n]$ is very small because of a low variance. The KGCP policy trusts the belief of the model and does not further verify this area. The KGCP policy decides to explore the center area instead as it is more uncertain and could contain a new optimum. Note that the prediction variance of the model is more important for KGCP than EI: this could imply KGCP benefits more from slice sampling. We test this hypothesis empirically in Section 5.4.

5.3.3 Gradient of deterministic KGCP

To facilitate the optimisation problem as defined in Equation (5.4), we investigate the computation of the gradient of Definition 5 as required by optimisation methods such as conjugate gradient [29]. The α_n^{EI} is usually a differentiable function, except for deterministic observations in the samples (as the prediction variance will equal zero). In practise, gradients point away from the decisions $\boldsymbol{x}$ avoiding the non-differentiable points. Its derivative can be computed as follows:

$$\frac{\mathrm{d}\alpha_n^{\text{EI}}}{\mathrm{d}\boldsymbol{x}} = (-z\Phi(-z) + \phi(z))\frac{\mathrm{d}s_n}{\mathrm{d}\boldsymbol{x}} - s_n\Phi(-z)\frac{\mathrm{d}z}{\mathrm{d}\boldsymbol{x}}, \tag{5.8a}$$

$$\frac{\mathrm{d}\mathbb{E}\left[D_n\right]}{\mathrm{d}\boldsymbol{x}} = (z\Phi(z) + \phi(z))\frac{\mathrm{d}s_n}{\mathrm{d}\boldsymbol{x}} + s_n\Phi(z)\frac{\mathrm{d}z}{\mathrm{d}\boldsymbol{x}}, \tag{5.8b}$$

$$\frac{\mathrm{d}z}{\mathrm{d}\boldsymbol{x}} = -\frac{1}{s_n}\frac{\mathrm{d}\mu_n}{\mathrm{d}\boldsymbol{x}} + z\frac{\mathrm{d}s_n}{\mathrm{d}\boldsymbol{x}}.$$

However, Definition 5 is not necessarily differentiable over the entire domain due to the $\min(.)$ function. Applying the approach proposed in Zhang et al. [30]:

Lemma 6.

$$\forall \boldsymbol{x} \in \mathcal{X} : \ \alpha_n^{EI} = \mathbb{E}\left[D_n\right] \iff f_{\max} = \mu_n(\boldsymbol{x})$$

Proof

$$\begin{aligned} & \alpha_n^{EI} = \mathbb{E}\left[D_n\right] \\ \Leftrightarrow \quad & (\mu_n - f_{\max})\Phi(-z) = (f_{\max} - \mu_n)\Phi(z) \\ \Leftrightarrow \quad & \mu_n = f_{\max} \end{aligned}$$

■

Lemma 6 implies α_n^{EI} equals $\mathbb{E}\left[D_n\right]$ for $z = 0$. Unfortunately, Equations (5.8a) and (5.8b) are not equal for this case, hence the gradient of Definition 5 is not guaranteed to exist for $z = 0$. Unlike the non-differentiability of the Equations (5.8a) and (5.8b) in the decisions for deterministic problems, these points are reachable which may cause optimisation failure.

Fortunately, both quantities required by Definition 5 can be computed very efficiently which means derivative-free meta-heuristics such as Particle Swarm Optimisation (PSO) [31] can be used to maximise the KGCP. Should a derivative be required however, the minimum function must be replaced by a smoother alternative. The minimum function corresponds to the l_∞ of the inverse of both arguments, hence it can be approximated by any l_a with $a \in \mathbb{Z}^+, 0 \ll a$ which results in a differentiable form, or by applying a soft minimum.

Definition 7. *A smooth (differentiable) form of the deterministic KGCP is defined as*

$$\bar{\alpha}^{KG}_{n,s}(\boldsymbol{x}) = -\frac{\log\left(\exp\left(-k\alpha^{EI}_{n}\right) + \exp\left(-k\mathbb{E}\left[D_n\right]\right)\right)}{k}$$

$$\frac{\mathrm{d}\bar{\alpha}^{KG}_{n,s}}{\mathrm{d}\boldsymbol{x}} = \frac{\exp\left(k\alpha^{EI}_{n}\right)\frac{\mathrm{d}\mathbb{E}[D_n]}{\mathrm{d}\boldsymbol{x}} + \exp\left(k\mathbb{E}\left[D_n\right]\right)\frac{\mathrm{d}\alpha^{EI}_{n}}{\mathrm{d}\boldsymbol{x}}}{\exp\left(k\alpha^{EI}_{n}\right) + \exp\left(k\mathbb{E}\left[D_n\right]\right)}.$$

The constant $k > 0$ controls the smoothing, for $k \to \infty$ the soft version of the KGCP is equivalent to the exact version of Definition 5.

5.4 Experiments

We implemented the KGCP formulation of Definition 5 in the SUMO Toolbox [32, 33], a research platform for surrogate modelling supporting grid-based computing for global surrogate modelling and simulation optimisation. The toolbox is easily extendible and already contains robust and tested implementations of Kriging and Expected Improvement for comparison purposes [21]. It also ships two different sampling policies for optimisation problems: UCB [13], and PES [14] which were both included for all test setups

5.4.1 Test setup

We compare seven different test setups: three different policies (KGCP, EI and UCB) for selecting the next decision are combined with two different methods (MLE and slice sampling as described in Section 5.2.2) for determining the hyperparameters $\boldsymbol{\theta}$. This results in the following test configurations: `KGCP-MLE`, `KGCP-SS`, `EI-MLE`, `EI-SS`, `UCB-MLE` and `UCB-SS`. The PES algorithm was proposed in Hernández-Lobato et al. [14] with slice sampling of the model hyperparameters, hence only this configuration was included (denoted as `PES`). For the slice sampling, 100 samples $\boldsymbol{\theta}$ are drawn from the marginal likelihood distribution: the MLE estimate is used as starting point. The test setups are applied to several optimisation test problems as described in the following sections. Each test setup was replicated 100 times on each test problem as variance in the results is expected: particularly on the multimodal test problems as these rely strongly on exploration for discovery of the optima. The results are averaged, and 95% Confidence Intervals (CI) are computed.

As a starting point, the test problem is first evaluated on a (maximin) Latin Hypercube of size 10, generated by the Translational Propagation algorithm [34]. After

obtaining the corresponding observations, a model is built and its hyperparameters are determined. From there on the sampling policy sequentially identifies a new decision: the policy is optimized by first applying Monte Carlo sampling, and optimizing the 10 best candidates using local search. When a choice is made for the decision $\boldsymbol{x}$, the observation is obtained and a new model is trained. This process continues until a pre-set number of N_{max} observations have been obtained.

All sampling policies are compatible with the Kriging model applied for these experiments. The `PES` method was originally formulated using a zero-mean GP and relies heavily on the posterior mean and covariance of the GP and its hyperparameters. To ensure no problems arise due to the Kriging models, the regression part was chosen to include only a constant regression function (Ordinary Kriging), hence the GP interpolating the residual becomes a zero-mean GP interpolating all response variability. If the regression basis would include other terms (e.g., a linear term) the minimum of the GP interpolating the residual would no longer necessarily correspond to the minimum of the function and further modifications of the algorithm would be required.

Each iteration n, the progress of finding the global optimum is evaluated by computing the Opportunity Cost (OC), which is also referred to as Instant Regret (IR). Defining $\hat{\boldsymbol{x}} = \arg\max_{\boldsymbol{x}} \ \mu_n(\boldsymbol{x})$ for the intermediate model μ_n, the OC equals:

$$\text{OC} = \left(\max_{\boldsymbol{u} \in \mathcal{X}} f(\boldsymbol{u}) \right) - f(\hat{\boldsymbol{x}}). \tag{5.9}$$

The OC represents how close the solution has come to the global optimum, if the process were to be ended. Note that usually in EGO applications $f(\hat{\boldsymbol{x}}) = f_{\max}$. Should the model believe some areas have a model response which is better than $f_{\max}$ it is usually ignored. The KGCP policy, however, aims to select decisions to optimally improve the belief of the model so it can identify the location of the best decision. Hence $\hat{\boldsymbol{x}}$ is chosen as the location of the model optimum.

5.4.2 Test problems

First, the test setups are applied to four different mathematical functions for global optimisation. All of them are defined for minimisation so in our experiments, functions are negated. We included both smooth as well as highly irregular and multimodal test functions, as well as a 6D test problem.

- The Branin function is used often as an optimisation benchmark function. It has two input parameters with ranges $[-5, 10]$ and $[0, 15]$, respectively, and three optima. The response surface is very smooth and does not contain any

abrupt discontinuities. Optimisation of this function is straightforward, and in total $N_{\text{max}} = 20$ points are evaluated.

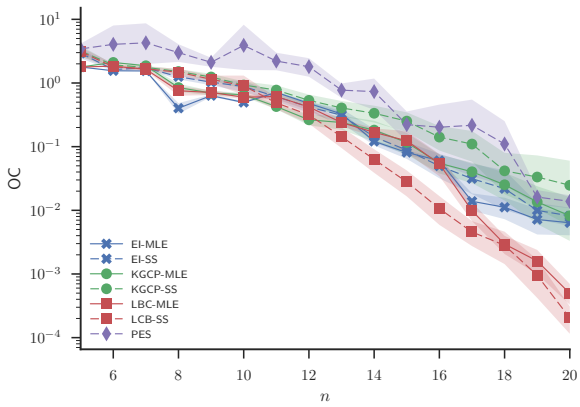
- The second test problem considered is the Hartmann function [20] with 6 input parameters on the interval $[0, 1]$. The function has a few local optima but is not very difficult to model as the response surface is again quite smooth. However the size of its search space is considerably larger due its dimensionality. The total number of evaluations was restricted to $N_{\text{max}} = 40$.

- The Schwefel function [35] is a more challenging problem compared to Branin and Hartmann as it is highly multimodal. The response surface is bumpy but still easy to characterize using Kriging. The range considered was $[-500, 500]$ for both input parameters. The actual global optimum is situated at $\hat{\boldsymbol{x}} = (420.9687, 420.9687)$. Because the optimisation problem is more difficult, we set $N_{\text{max}} = 100$.

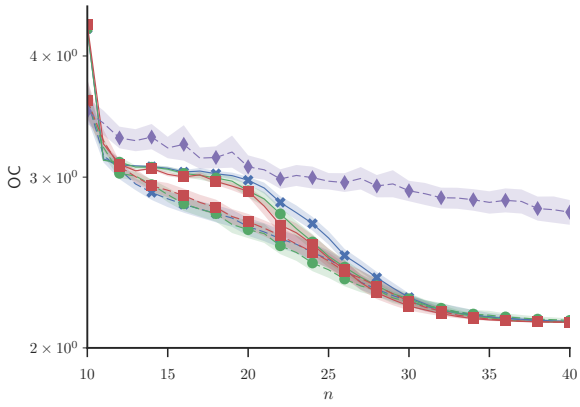
- The last mathematical test problem is the Eggholder function [36]. This function is very multimodal, and its response behaviour is also difficult to approximate. The problem has two input parameters over the range $[-512, 512]$ and the global optimum is situated at $\hat{\boldsymbol{x}} = (512, 404.2319)$. It is surrounded by several local optima which tend to trap optimisation algorithms as they are very deceptive (because they are quite steep). Like the Schwefel function the optimisation of this function was given a computational budget of $N_{\text{max}} = 100$.

The evolution of the OC as more observations are obtained for these test problems is shown in Figure 5.4. In addition, Table 5.1 contains a summary for all experiments. For each test setup the mean OC and its 95% CI are shown at the end of the runs (after N_{max} evaluations).

The Branin function does not require many evaluations in order to find the global optimum. All test configurations perform similar except `EI-MLE` seems to be stuck in a poor choice of hyperparameters for a few iterations: the optimum of the model however is near the true optimum. As more observations are available the situation is corrected causing a rising OC. Note that `EI-SS`, `KGCP-MLE` and `KGCP-SS` avoid this situation. `PES` performs slightly worse than EI in comparison to the results in Hernández-Lobato et al. [14] which might be caused by EI performing better due to the absence of noise. However, for this problem `UCB-MLE` and `UCB-SS` ultimately find the best solution: these methods are oriented towards exploitation, which works well for this test problem as the true optimum is not difficult to find. All methods perform similar on the Hartmann function, although methods with slice sampling drop the OC faster. The `PES` however is less successful, which is in line with the results reported in Hernández-Lobato et al. [14].

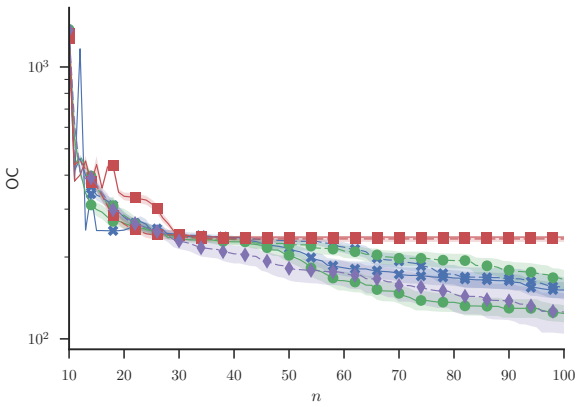


(a) Branin 2D

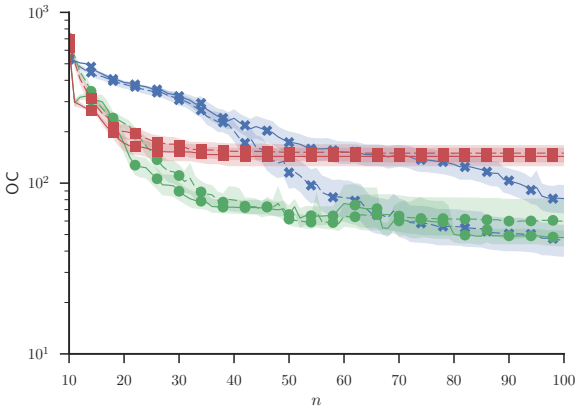


(b) Hartmann 6D

Figure 5.4: Synthetic test problems: evolution of the mean OC as more observations are obtained.



(c) Schwefel 2D



(d) Eggholder 2D

Figure 5.4: Synthetic test problems: evolution of the mean OC as more observations are obtained.

Table 5.1: Test problems: mean OC and 95% CI obtained after obtaining N_{max} observations by averaging over 100 runs, for each test setup for determining the hyperparameters $\boldsymbol{\theta}$ applied to the test problems. The best performing algorithms for each test problem is highlighted in bold.

(a) KGCP

		MLE		SS	
Problem	N	$\mathbb{E}$ (OC)	95% CI	$\mathbb{E}$ (OC)	95% CI
Branin $d = 2$	20	0.006	0.009 to 0.003	0.025	0 to 0.06
Hartmann $d = 6$	40	**2.12**	2.12 to 2.13	2.14	2.13 to 2.15
Schwefel $d = 2$	100	**124.0**	114.8 to 133.1	156.2	150.7 to 179.7
Eggholder $d = 2$	100	48.0	42.8 to 53.2	60	41.8 to 78.1
Truss $d = 10$	250	1.64	1.44 to 1.84	**1.17**	1.0 to 1.35

(b) EI

		MLE		SS	
Problem	N	$\mathbb{E}$ (OC)	95% CI	$\mathbb{E}$ (OC)	95% CI
Branin $d = 2$	20	0.008	0.001 to 0.014	0.008	0.005 to 0.010
Hartmann $d = 6$	40	2.13	2.13 to 2.14	2.13	2.13 to 2.13
Schwefel $d = 2$	100	151.2	140.8 to 161.7	154.0	139.4 to 168.5
Eggholder $d = 2$	100	81.2	65.0 to 97.3	**46.41**	36.3 to 56.5
Truss $d = 10$	250	1.76	1.59 to 1.92	1.50	1.31 to 1.70

Table 5.1: Test problems: mean OC and 95% CI obtained after obtaining N_{max} observations by averaging over 100 runs, for each test setup for determining the hyperparameters $\boldsymbol{\theta}$ applied to the test problems. The best performing algorithms for each test problem is highlighted in bold.

(c) UCB

		MLE		SS	
Problem	N	$\mathbb{E}$ (OC)	95% CI	$\mathbb{E}$ (OC)	95% CI
Branin $d = 2$	20	**0**	0 to 0	**0**	0 to 0
Hartmann $d = 6$	40	2.13	2.12 to 2.13	2.13	2.12 to 2.13
Schwefel $d = 2$	100	236.9	236.9 to 236.9	233.1	233.1 to 233.1
Eggholder $d = 2$	100	143.3	125.7 to 160.9	149.8	132.2 to 167.4
Truss $d = 10$	250	4.63	4.19 to 5.06	2.39	2.02 to 2.76

(d) PES

Problem	N	$\mathbb{E}$ (OC)	95% CI
Branin $d = 2$	20	0.01	0 to 0.01
Hartmann $d = 6$	40	2.76	2.75 to 2.77
Schwefel $d = 2$	100	125.5	104.1 to 146.8
Eggholder $d = 2$	100	-	-
Truss $d = 10$	250	2.84	2.64 to 3.04

The Schwefel test problem, however, is more multimodal. A small difference appears: PES and KGCP−MLE reduce the OC faster. When about 50 observations have been obtained the runs clearly obtain a lower OC. The other test setups hardly differ, with the exception of the UCB cases which quickly get stuck in a local minimum and remain there due to a lack of exploration. Note that the KGCP−SS setup performs worse in comparison to both EI setups.

For the most difficult mathematical test problem, the Eggholder function, the difference is more significant. Again, UCB−MLE and UCB−SS both quickly get stuck in a local minimum. For this test problem, the KGCP−MLE and KGCP−SS configurations quickly improve a lot more compared to EI−MLE and EI−SS. The EI−SS setup performs better for this problem compared to EI−MLE, but only catches up with the KGCP setups after 20 additional observations. When 100 observations have been made, all test setups provide similar OC scores, however KGCP runs for these test problems could already have ended after 40 observations and provide satisfactory results.

The PES algorithm encountered severe numerical stability issues on the Eggholder test problem, causing half of the runs to crash. To ensure the problem was not related to the implementation the experiments were repeated using the original code released by Hernández-Lobato et al. [14] resulting in similar problems. We narrowed the problem down to the inverse of the posterior covariance matrix part of the Expectation Propagation step of the algorithm. For deterministic problems, this matrix has only one element which is not near zero (the last element of the diagonal) which corresponds to the covariance of the sample drawn from the conditional distribution of the global maximiser. For this test problem this element frequently became zero resulting in a violation of the positive definiteness. Hernández-Lobato et al. [14] report stability issues when the sample from the conditional distribution is too close to another previously evaluated point and included a solution. Unfortunately, this seems to be insufficient for this problem. Because so many runs crashed we were unable to generate and include statistical significant results of PES.

Generally, for the mathematical test problems KGCP−MLE obtains the best results. The other test setups are very comparable and show not much differences, with the exception of the UCB setups which are only satisfactory for the easier Branin and Hartmann test problems. The complexity of the Eggholder problem results in a significant difference between the EI and KGCP sampling policies. Surprisingly, we do not see an improvement of KGCP−SS over KGCP−MLE: in fact it performs worse in 3 out of 4 problems which is in contrast with our earlier hypothesis. A possible explanation to this is the nature of the problems: these can be interpolated nicely which implies a well-defined unimodal optimum of the likelihood that can be easily identified with MLE. Slice sampling draws most additional $\boldsymbol{\theta}$ near this optimum of the MLE estimate but the corresponding models are likely to be affected

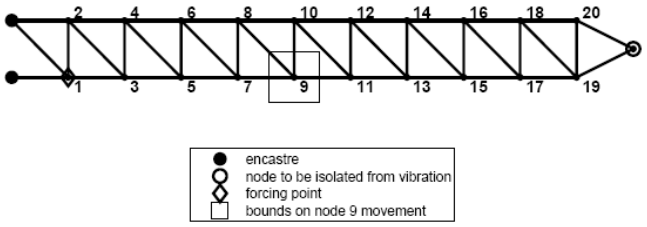


Figure 5.5: Truss problem (10D): two-dimensional structure. Courtesy of Forrester and Jones [37].

negatively. Hence errors are averaged and the policy is not performing as powerful. In other words, for these mathematical functions there is not a lot of uncertainty on the parameter estimate, and the MLE solution on its own appears to be most powerful.

5.4.3 Truss structure optimisation (10D)

Our final test problem is a structural dynamics problem of a two-dimensional truss for maximum passive vibration isolation. The truss is constructed using 42 Euler-Bernoulli beams having two finite elements per beam. The truss is subjected to unit force excitation at node 1, across the 100 Hz to 200 Hz frequency range. The input parameters correspond to moving 5 nodes (8 through 12) in a 0.9×0.9 square. The other nodes are kept fixed according to the structure depicted in Figure 5.5. The band-averaged vibration attenuation at the tip, compared to the baseline structure is to be maximised. The more nodes are included as design parameters, the more multi-modal the response is as many (sub-)optimal configurations are possible. It was reported earlier by Forrester and Jones [37] that standard EI tends to be tricked and has a low probability of identifying the global optimum. Because this experiment is a difficult 10D problem, a total amount of $N_{\max} = 250$ observations are allowed. The same sampling policies, both tested with the MLE and slice sampling approaches are tested, and each configuration is replicated 100 times to compute the 95% CI.

A detailed evolution of the OC as more observations are obtained is illustrated in Figure 5.6. Table 5.1 also provides the mean OC and 95% CI after 250 evaluations for the Truss optimisation. As opposed to the runs on the mathematical test problems, the benefit of slice sampling starts to show on this structural optimisation problem. The uncertainty on the point-estimate of the hyperparameters is larger and the inclusion of different hyperparameters clearly results in an improvement. Both `EI-MLE` and `KGCP-MLE` are less performant, the latter slightly beating the

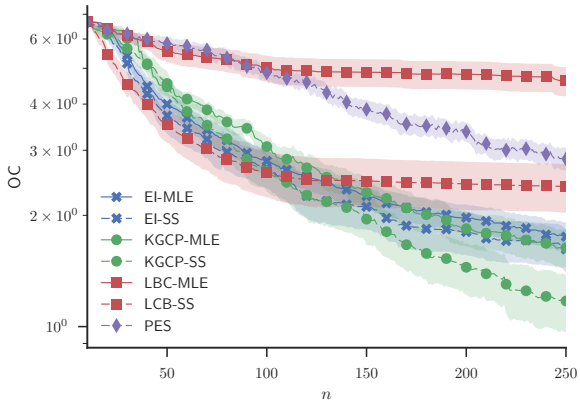


Figure 5.6: Truss problem (10D): evolution of the mean OC as more observations are obtained.

former. The best results are obtained by `KGCP-SS` as it keeps improving when 170 observations have been obtained, whereas `EI-SS` seems to stabilize at this point. However, at the beginning of the run the drop of the OC is slightly better for the `EI-SS` compared to the other test setups. The `UCB-MLE` and `UCB-SS` runs are performing significantly worse and seem to be stuck at some point, as observed previously for the mathematical problems. Using slice sampling the result is slightly better but it is still outperformed by all other test setups. Finally, due to the dimensionality of the problem space `PES` seems to encounter a similar problem as reported on Hartmann: due to its exploring nature it leads to a slower reduction of OC.

5.5 Conclusion

In this work we derived a closed formula for fast computation of the KGCP for expensive optimisation problems, assuming the response is noiseless (deterministic). We lifted the computational complexity disadvantage with respect to EI making the KGCP a feasible choice for this class of problems. For simple problems the KGCP formulation gives comparable results to the popular EI, but as the complexity increases (multimodal problems) the extra tendency of KGCP towards exploration helps to avoid being stuck for some time in a local optimum. For the high-dimensional problems considered its exploitation capabilities seem to give it an advantage over PES which has difficulties locating the optimum due to size of the search space. Because the KGCP potentially gives better results faster and has

similar complexity as EI, we suggest its usage.

It should be kept in mind that KGCP essentially trusts the model if it is certain a region contains a better optimum (better prediction mean, low prediction variance). Unfortunately this trust comes with a risk: it is crucial to ensure that the model, and more specifically the prediction variance is as accurate as possible. However if the optimum of the likelihood is well-defined and the surface is unimodal, only little uncertainty is expected on the model parameters, resulting in the configurations with MLE outperforming slice sampling in our synthetic experiments. For the structural optimisation problem however, this is no longer the case. Hence the choice of whether to use slice sampling or not should be made on the type of problem, and more specifically the uncertainty expected on the hyperparameters which is indicated by the shape of their distribution (i.e., the likelihood), or in a more general sense, by the complexity of approximating the response.

5.6 References

[1] K. Goethals, I. Couckuyt, T. Dhaene, and A. Janssens. "Sensitivity of night cooling performance to room/system design: Surrogate models based on CFD". *Building and Environment* 58 (2012), pp. 23–36.

[2] D. Gorissen. "Grid-enabled adaptive surrogate modeling for computer aided engineering". PhD thesis. Ghent University, 2010, XIV, 362 [in multiple pagination]. ISBN: 9789085783503.

[3] J. Močkus. "On Bayesian methods for seeking the extremum". *Optimization Techniques IFIP Technical Conference*. Ed. by G. Marchuk. Springer. Novosibirsk: Springer Berlin Heidelberg, July 1975, pp. 400–404. ISBN: 978-3-540-37497-8.

[4] D. R. Jones, M. Schonlau, and W. J. Welch. "Efficient Global Optimization of Expensive Black-Box Functions". *J. of Global Optimization* 13.4 (1998), pp. 455–492. ISSN: 0925-5001.

[5] J. Knowles. "ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems". *Evolutionary Computation, IEEE Transactions on* 10.1 (2006), pp. 50–66.

[6] D. Hernández-Lobato, J. M. Hernández-Lobato, A. Shah, and R. P. Adams. "Predictive Entropy Search for Multi-objective Bayesian Optimization". *Proceedings of the 33rd International Conference on Machine Learning (ICML-16)*. Ed. by M.-F. Balcan and K. Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. Manhattan, New York, USA: PMLR, 2016, pp. 1492–1501.

[7] M. T. M. Emmerich, A. H. Deutz, and J. W. Klinkenberg. "Hypervolume-based expected improvement: Monotonicity properties and exact computation". *Congress on Evolutionary Computation (CEC)*. Ed. by M. T. M. Emmerich and P. Hingston. IEEE. New Orleans, LA, USA: Institute of Electrical and Electronics Engineers, Inc., 2011, pp. 2147–2154. ISBN: 978-1-4244-7834-7.

[8] I. Couckuyt, D. Deschrijver, and T. Dhaene. "Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization". *Journal of Global Optimization* 60.3 (2014), pp. 575–594. ISSN: 0925-5001.

[9] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. "Algorithms for hyper-parameter optimization". *Advances in Neural Information Processing Sys-*

tems 24. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger. Granada, Spain: Curran Associates, Inc., 2011, pp. 2546–2554.

[10] J. Snoek, H. Larochelle, and R. P. Adams. "Practical Bayesian optimization of machine learning algorithms". *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger. Lake Tahoe, Sierra Nevada, USA: Curran Associates, Inc., 2012, pp. 2951–2959.

[11] J. Gardner, M. Kusner, K. Q. Weinberger, J. Cunningham, and Z. Xu. "Bayesian Optimization with Inequality Constraints". *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. Ed. by T. Jebara and E. P. Xing. Proceedings of Machine Learning Research. Beijing, China: PMLR, 2014, pp. 937–945.

[12] D. D. Cox and S. John. "SDO: A statistical method for global optimization". *Multidisciplinary design optimization: state of the art* (1997), pp. 315–329.

[13] N. D. Freitas, M. Zoghi, and A. J. Smola. "Exponential Regret Bounds for Gaussian Process Bandits with Deterministic Observations". *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*. Ed. by J. Langford and J. Pineau. Edinburgh, Scotland: ACM, 2012, pp. 1743–1750.

[14] J. M. Hernández-Lobato, M. W. Hoffman, and Z. Ghahramani. "Predictive entropy search for efficient global optimization of black-box functions". *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger. Montreal, Canada: Curran Associates, Inc., 2014, pp. 918–926.

[15] P. Frazier, W. Powell, and S. Dayanik. "The knowledge-gradient policy for correlated normal beliefs". *INFORMS journal on Computing* 21.4 (2009), pp. 599–613.

[16] W. Scott, P. Frazier, and W. Powell. "The correlated knowledge gradient for simulation optimization of continuous parameters using gaussian process regression". *SIAM Journal on Optimization* 21.3 (2011), pp. 996–1026.

[17] W. B. Powell and I. O. Ryzhov. *Optimal learning*. Vol. 841. John Wiley & Sons, 2012.

[18] A. D. Bull. "Convergence rates of efficient global optimization algorithms". *Journal of Machine Learning Research* 12 (2011), pp. 2879–2904.

[19] D. Huang, T. T. Allen, W. I. Notz, and N. Zeng. "Global optimization of stochastic black-box systems via sequential kriging meta-models". *Journal of Global Optimization* 34.3 (2006), pp. 441–466.

[20] V. Picheny, T. Wagner, and D. Ginsbourger. "A benchmark of kriging-based infill criteria for noisy optimization". *Structural and Multidisciplinary Optimization* 48.3 (2013), pp. 607–626.

[21] I. Couckuyt, T. Dhaene, and P. Demeester. "ooDACE Toolbox: A Flexible Object-Oriented Kriging Implementation". *Journal of Machine Learning Research* 15 (2014), pp. 3183–3186.

[22] M. L. Stein. *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media, 2012.

[23] D. den Hertog, J. P. Kleijnen, and A. Y. Siem. "The correct Kriging variance estimated by bootstrapping". *Operational Research Society* 57 (2006), pp. 400–409.

[24] I. Murray and R. P. Adams. "Slice sampling covariance hyperparameters of latent Gaussian models". *Advances in Neural Information Processing Systems 23*. Ed. by J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta. Lake Tahoe, Sierra Nevada, USA: Curran Associates, Inc., 2010, pp. 1732–1740.

[25] J. Wu and P. Frazier. "The Parallel Knowledge Gradient Method for Batch Bayesian Optimization". *Advances in Neural Information Processing Systems 29*. Ed. by D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett. Curran Associates, Inc., 2016, pp. 3126–3134.

[26] I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, and L. Knockaert. "Surrogate-based infill optimization applied to electromagnetic problems". *International Journal of RF and Microwave Computer-Aided Engineering* 20.5 (2010), pp. 492–501.

[27] C. Gazda, D. V. Ginste, H. Rogier, I. Couckuyt, T. Dhaene, K. Stijnen, and H. Pues. "Efficient optimization of the integrity behavior of analog nonlinear devices using surrogate models". *17th IEEE Workshop on Signal and Power Integrity (SPI)*. IEEE. Piscataway, New Jersey, USA: Institute of Electrical and Electronics Engineers, Inc., 2013, pp. 1–4.

[28] M. T. Mehari, E. De Poorter, I. Couckuyt, D. Deschrijver, J. Vanhie-Van Gerwen, D. Pareit, T. Dhaene, and I. Moerman. "Efficient global optimization of multi-parameter network problems on wireless testbeds". *Ad Hoc Networks* 29 (2015), pp. 15–31.

[29] P. E. Gill and W. Murray. *Conjugate-Gradient Methods for Large-Scale Nonlinear Optimization.* Tech. rep. DTIC Document, 1979.

[30] X. Zhang, C.-C. Hang, S. Tan, and P.-Z. Wang. "The min-max function differentiation and training of fuzzy neural networks". *Neural Networks, IEEE Transactions on* 7.5 (1996), pp. 1139–1150.

[31] J. Kennedy and R. Eberhart. "Particle swarm optimization". *Proceedings of the IEEE International Conference on Neural Networks*. IEEE. Perth, WA, Australia: Institute of Electrical and Electronics Engineers, Inc, 1995.

[32] D. Gorissen, K. Crombecq, I. Couckuyt, P. Demeester, and T. Dhaene. "A Surrogate Modeling and Adaptive Sampling Toolbox for Computer Based Design". *Journal of Machine Learning Research* 11 (2010). Available at http://sumo.intec.ugent.be, pp. 2051–2055.

[33] J. van der Herten, I. Couckuyt, D. Deschrijver, and T. Dhaene. "Adaptive classification under computational budget constraints using sequential data gathering". *Advances in Engineering Software* 99 (2016), pp. 137–146.

[34] F. A. Viana, G. Venter, and V. Balabanov. "An algorithm for fast optimal Latin hypercube design of experiments". *International journal for numerical methods in engineering* 82.2 (2010), pp. 135–156.

[35] M. Laguna and R. Martí. "Experimental testing of advanced scatter search designs for global optimization of multimodal functions". *Journal of Global Optimization* 33.2 (2005), pp. 235–255.

[36] M. Jamil and X.-S. Yang. "A literature survey of benchmark functions for global optimisation problems". *International Journal of Mathematical Modelling and Numerical Optimisation* 4.2 (2013), pp. 150–194.

[37] A. I. J. Forrester and D. R. Jones. "Global optimization of deceptive functions with sparse sampling". *12th AIAA/ISSMO multidisciplinary analysis and optimization conference*. Vol. 1012. Victoria, BC, Canada: Aerospace Research Central, Sept. 2008.

"You don't learn to walk by following rules.
You learn by doing, and by falling over."

— Richard Branson

6 Conclusion

6.1 Summary

Chapter 2 provided an in-depth presentation of the fundamental concepts and base approaches for this dissertation and included a full overview of the adaptive surrogate modelling concept (with sequential design). All aspects, including hyperparameter optimisation, choice of quality metric and approximation type, experimental design, and error functions, were discussed in detail. The field of surrogate modelling has also been positioned as a multi-disciplinary research field and the relation with other fields was discussed. Finally, the SUMO toolbox was presented. This state-of-the-art research platform provides an integrated approach, for surrogate modelling with (or without) sequential design ready to be used by engineers and scientists. It is available for academic use and can be downloaded from http://sumo.intec.ugent.be under the Affero General Public License (AGPL)[1].

After an overview and a detailed discussion of existing sequential sampling methods in Chapter 3, the topic of surrogate modelling with classification models for simulations and with outputs restricted to a set of discrete values was discussed. In

[1] http://www.fsf.org/licensing/licenses/agpl-3.0.html

this context, the Neighbourhood-Voronoi (N-V) algorithm was presented [1, 2] for detection and refinement of class boundaries. This novel functionality was added to the SUMO toolbox, and illustrated on a number of applications.

Next, an improvement to a mature sequential sampling method, known as LOLA-Voronoi, was presented in Chapter 4. Although the LOLA-Voronoi algorithm is a very effective method in terms of its number of required expensive evaluations to obtain an accurate model, its computational complexity grows exponentially when more input parameters are considered. The novel FLOLA-Voronoi algorithm reduces the complexity significantly in high-dimensional design spaces by applying Fuzzy logic in the neighbour selection process, but is able to maintain the (limited) number of required expensive evaluations. It also inherits all favourable properties of its ancestor, such as model independence, making it an excellent choice for heterogeneous modelling environments. Additionally, the role of fractional distances in high-dimensions was discussed and a version of FLOLA-Voronoi based on these distances was included in the comparison. It was shown how these distances can offer some advantage as dimensionality grows during the early stages of the process. This comes, however, with a significant computational cost.

Chapter 5 discusses the Knowledge-Gradient for Continuous Parameters (KGCP) algorithm, a sequential sampling methodology for Surrogate-Based Optimisation (SBO). Originally this methodology takes into account output uncertainty as it was defined for problems with a stochastic output. An equality relation between the KGCP and the Expected Improvement (EI) was obtained by assuming deterministic output, resulting in both an efficient method to compute the KGCP as well as a very intuitive interpretation: the KGCP trusts the model belief more in comparison to EI. This novel formulation was implemented in the SUMO toolbox and benchmarked extensively on four benchmark functions and one real-world application, together with three other state-of-the-art adaptive sampling algorithms for optimisation. In addition, two different strategies for hyperparameter optimisation, Maximum Likelihood Estimation (MLE) and Markov Chain Monte Carlo (MCMC), were applied to determine the hyperparameters of the Kriging models. It was shown how the KGCP can be advantageous over the other methods considered when the optimisation surface is complex as it tends to explore faster. Furthermore, the benefit of MCMC only pays off in case some degree of uncertainty over the hyperparameters is present. However, when the posterior of $\boldsymbol{\theta}$ is a very sharp peak, the results with MLE are better.

In summary, this dissertation presents (in Chapter 3) the ability of surrogate modelling with discrete outputs by means of classification and specific sampling algorithms. Which is valuable for constraint satisfaction problems and feasibility studies as well as for inverse modelling. Furthermore, (in Chapter 4) a severe disadvantage of an existing adaptive sampling method was lifted by improving its

computational efficiency in higher-dimensional design spaces. Next (in Chapter 5) another adaptive sampling method was derived given the deterministic assumption, resulting in a very interpretable and efficient formula. Both these improvements lift some of important drawbacks and result in very applicable and reliable methods for selecting evaluations of input-output mappings which are expensive to evaluate, from both the global accuracy as well as optimisation perspectives.

6.2 Future work

6.2.1 High-dimensional problems

A major challenge in many fields of science and engineering is the explosive growth of design spaces caused by the increase of number of considered input parameters. This is often referred to as the *curse of dimensionality* as introduced in Section 1.4.2. This term is slightly misleading, as in practise it is not a single issue but refers to a broad range of difficult and counter-intuitive challenges including information sparsity and the degradation of distance metrics as was explained in Section 4.6.1.

There is no general solution to solve all problems at once, but an important step is reducing some issues related to algorithmic complexity as shown in Chapter 4. In general, however, it can be concluded that the traditional surrogate modelling has some shortcomings when dimensionality increases. The traditional space-coverage concept conflicts heavily with distance metric issues [3, 4], and generally all algorithms involving any kind of exploration emit random-like behaviour as coincidence starts to be involved. The space between data points becomes large and it becomes quite unlikely a new data point discovers information of interest (even though it hidden somewhere). This is further amplified sampling criteria such as EI which tend to become uninformative and feature large monotonous regions. This results in large variability between runs: it becomes difficult to conclude that one optimisation algorithm performs better compared to another when repeating an experiment might result in conflicting findings. Within this context, theoretical convergence analysis such as presented in [5, 6] are very valuable.

Fortunately, it is known that when a large number of parameters are present, many of them do not influence the output significantly. The work of Van Steenkiste et al. [7] shows how surrogate models can be used to efficiently identify parameter relevance up to 20 dimensions, long before global accuracy is achieved. This knowledge can be applied to exclude certain input parameters early during the surrogate modelling process. However, parameters which only influence the output

for a small part of the design space might preliminary be excluded before this behaviour is identified.

A very promising direction is the identification of subspaces which cover the entire output variability. Approaches such as presented in literature [8–11] offer a great advantage as they allow the successful applicability of techniques in low-dimensional spaces by constructing a mapping to a higher dimension. Interestingly, this results in strong data correlations in the high-dimensional space, a necessary condition for metric relevance as defined by Kabán [4]. This observation further holds within the recent successes of deep learning. Although the popular image data sets result in a very high-dimensional input space, the images themselves do not cover the entire space but only cover parts in a correlated and structured manner. This indicates that finding a manifold of lower dimensionality to operate on is a very valuable and theoretically correct way to proceed.

Today, linear embedding approaches exist [8–10], of which the approach of Garnett et al. [11] is very interesting. This approach also includes an adaptive sampling method to jointly learn the linear manifold together with the hyperparameters of a Gaussian Process (GP). A logical extension would be learning non-linear manifolds which can be achieved by, for instance, Gaussian Process Latent Variable Models (GPLVMs) [12–14]. In addition, more research to actively learn these manifolds through adaptive sampling is required as unlike the successes in machine learning, surrogate modelling handles a black-box function with an associated high-dimensional input space for which the manifold still has to be determined rather than a correlated set of data which is already on the manifold.

6.2.2 Generative modelling

One of the core aspirations of artificial intelligence is to develop algorithms and techniques that endow computers with an understanding of our world. Generative models are one of the most promising approaches. In generative modelling, the focus is on finding approximative distributions that reassemble a true data distribution. Generative Adversarial Networks (GANs) [15] are one approach to find the generative distribution. In GANs the generative models are estimated via an adversarial process in which two models are simultaneously trained. The training procedure is comparable to a minimax game between two separate models. Their introduction resulted in significant improvements in image de-noising, superresolution, exploration in reinforcement learning, and neural network pre-training [16, 17].

Other promising approaches are GPLVMs and Deep GPs, which may be preferable in a Data-Efficient Machine Learning (DEML) setting, and more appropriate in

the context of surrogate modelling. Along with manifold learning in the context of high-dimensional problems, applications of the partial observations prediction formulas for filling in blanks [14] and the dynamical versions for prediction and extrapolation along time and frequency axes are of interest. A different promising area would be identification of subspaces for high-dimensional Pareto fronts and learning the mapping to the input space.

6.2.3 Uncertainty quantification application of surrogate models

One of the reasons Bayesian approaches have become very popular in surrogate modelling is their explicit quantification of the uncertainty associated with the input/output mapping. When not much information is available, it is very relevant to collect information about what is unknown. This information then becomes very valuable to make decisions on stopping the data collection process.

Under the traditional setting surrogate modelling does not only assume deterministic output, but also considers the input parameters to be exact. This might be difficult to realise in practise: due to equipment limitations it is not possible to assert all instances of a complex system are exactly the same. A known problem is determining your location with a GPS device: each measurement is subject to an offset coming from a latent distribution. In order to generate some level of confidence about the true location of the GPS device, this distribution must first be estimated, and may then be used to infer the distribution over the current location of the device. The latter distribution may be further conditioned on the history of measurements. Also during production of electronic circuits it is impossible to be certain that each chip has exactly the same specification. The propagation of this input uncertainty through the mapping results in output uncertainty: it can be very important to quantify the effects of this uncertainty propagation. In the optimisation community this is known as *robust optimisation* which prefers smooth local optima over sharp pointy global optima. The latter tend to lead to bad results when input conditions start to vary.

The most straightforward approach to quantify the output uncertainty as a function of the input with given distributions over the inputs is by means of Monte Carlo (MC) sampling. This can be computationally intensive when many parameters are present and results in empirical distributions which cannot be used in further computation directly. However, the MC method is applicable to any input/output mapping which can be (easily) evaluated. However, there are also analytical methods, often for GP models. For prediction of uncertain test points, Girard et al. [18] derived formula to calculate the first and second moments of the distribution, resulting from propagating a Gaussian density through a GP. As an exception, for

an Radial Basis Function (RBF) kernel all moments can be computed [19]. It is also possible to train GPs with uncertain training data [20]. Later, Damianou [14] gave a different approach to training GPs with uncertain training data by using Variational GPLVMs. The latter work also provides a broad background of other work related to uncertainty propagation through GPs.

Practical applications of these analytical methods for uncertainty propagation within the field of engineering are still quite exceptional. However, this opens up a wide range of possibilities for feasibility studies and robust design optimisation. In addition to the input/output behaviour, the engineer would also be able to perform an analysis of the production constraints and incorporate this knowledge in design decisions.

6.3 References

[1] P. Singh, D. Deschrijver, D. Pissoort, and T. Dhaene. "Adaptive classification algorithm for EMC-compliance testing of electronic devices". *Electronics Letters* 49.24 (2013), pp. 1526–1528.

[2] P. Singh, J. van der Herten, D. Deschrijver, I. Couckuyt, and T. Dhaene. "A sequential sampling strategy for adaptive classification of computationally expensive data". *Structural and Multidisciplinary Optimization* (2016), pp. 1–14.

[3] D. Francois, V. Wertz, and M. Verleysen. "The concentration of fractional distances". *Knowledge and Data Engineering, IEEE Transactions on* 19.7 (2007), pp. 873–886.

[4] A. Kabán. "Non-parametric detection of meaningless distances in high dimensional data". *Statistics and Computing* 22.2 (2012), pp. 375–385.

[5] N. D. Freitas, M. Zoghi, and A. J. Smola. "Exponential Regret Bounds for Gaussian Process Bandits with Deterministic Observations". *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*. Ed. by J. Langford and J. Pineau. Edinburgh, Scotland: ACM, 2012, pp. 1743–1750.

[6] J. Bect, F. Bachoc, and D. Ginsbourger. "A supermartingale approach to Gaussian process based sequential design of experiments". *arXiv preprint arXiv:1608.01118* (2016).

[7] T. Van Steenkiste, J. van der Herten, I. Couckuyt, and T. Dhaene. "Sensitivity Analysis of Expensive Black-box Systems using Metamodeling". *Proceedings of the 49th Conference on Winter Simulation*. Ed. by T. M. K. Roeder, P. I. Frazier, R. Szechtman, E. Zhou, T. Huschka, and S. E. Chick. Washington, DC, USA: Institute of Electrical and Electronics Engineers, Inc, 2016.

[8] J. Son, S. Gupta, and G. Tan. "Bayesian Optimization in High Dimensional Input Space". *Proceedings of the 9th EAI International Conference on Simulation Tools and Techniques*. SIMUTOOLS'16. Prague, Czech Republic: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2016, pp. 18–27. ISBN: 978-1-63190-120-1.

[9] Z. Wang, F. Hutter, M. Zoghi, D. Matheson, and N. de Feitas. "Bayesian optimization in a billion dimensions via random embeddings". *Journal of Artificial Intelligence Research* 55 (2016), pp. 361–387.

[10] M. Binois, D. Ginsbourger, and O. Roustant. “A Warped Kernel Improving Robustness in Bayesian Optimization Via Random Embeddings”. *Learning and Intelligent Optimization: 9th International Conference, LION 9, Lille, France, January 12-15, 2015. Revised Selected Papers*. Ed. by C. Dhaenens, L. Jourdan, and M.-E. Marmion. Cham: Springer International Publishing, 2015, pp. 281–286. ISBN: 978-3-319-19084-6.

[11] R. Garnett, M. A. Osborne, and P. Hennig. “Active learning of linear embeddings for Gaussian processes”. *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence*. Ed. by Z. M.L. and T. J. Quebec, Canada: AUAI Press, 2014, pp. 230–239.

[12] N. Lawrence. “Probabilistic non-linear principal component analysis with Gaussian process latent variable models”. *Journal of Machine Learning Research* 6.Nov (2005), pp. 1783–1816.

[13] M. K. Titsias and N. D. Lawrence. “Bayesian Gaussian Process Latent Variable Model.” *AISTATS*. Ed. by Y. W. Teh and M. Titterington. Vol. 9. Proceedings of Machine Learning Research. Sardinia, Italy: PMLR, 2010, pp. 844–851.

[14] A. Damianou. “Deep Gaussian processes and variational propagation of uncertainty”. PhD thesis. University of Sheffield, 2015.

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. “Generative Adversarial Nets”. *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger. Curran Associates, Inc., 2014, pp. 2672–2680.

[16] R. Houthooft, X. Chen, X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel. “VIME: Variational Information Maximizing Exploration”. *Advances in Neural Information Processing Systems 29*. Ed. by D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett. Curran Associates, Inc., 2016, pp. 1109–1117.

[17] A. Radford, L. Metz, and S. Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks”. *arXiv preprint arXiv:1511.06434* (2015).

[18] A. Girard, C. E. Rasmussen, J. Q. Candela, and R. Murray-Smith. “Gaussian process priors with uncertain inputs-application to multiple-step ahead time series forecasting”. *Advances in neural information processing systems* (2003), pp. 545–552.

[19] T. O'Hagan. *Uncertainty analysis: the variance of the variance*. Tech. rep. Managing Uncertainty in Complex Models (MUCM), 2011.

[20] A. Mchutchon and C. E. Rasmussen. "Gaussian Process Training with Input Noise". *Advances in Neural Information Processing Systems 24*. Ed. by J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger. Curran Associates, Inc., 2011, pp. 1341–1349.

"If we are not ashamed to think it, we should not be ashamed to say it."

— Marcus Tullius Cicero

A

Multi-Objective Optimisation with Student-t Processes

J. van der Herten, I. Couckuyt, and T. Dhaene.

***BayesOpt 2016 Workshop at the 30th Conference on Neural Information Processing Systems*. Barcelona, Spain, 2016**

Abstract *Student-t processes have recently been proposed as an appealing alternative non-parameteric function prior. They feature enhanced flexibility and predictive variance. In this work the use of Student-t processes are explored for multi-objective Bayesian optimisation. In particular, an analytical expression for the hypervolume-based probability of improvement is developed for independent Student-t process priors of the objectives. Its effectiveness is shown on a multi-objective optimisation problem which is known to be difficult with traditional Gaussian processes.*

A.1 Introduction

The use of Bayesian models and acquisition functions to guide the optimisation of expensive, noisy, black-box functions (Bayesian Optimisation) has become more popular over the years, and has recently been applied to a wide variety of problems in several fields. The next candidate for evaluation of the computationally expensive black-box function is selected by optimizing an acquisition function relying on Bayesian model(s) approximating the previously observed responses of the black-box function(s). Within Machine Learning, it has for instance been applied to optimize model hyperparameters [2, 3], as model training involving a lot of data typically makes use of traditional numeric optimisation infeasible.

Within the field of engineering, Jones et al. [4] introduced the combination of Expected Improvement (EI) and Kriging models for optimisation of computer simulations which can take up to several days to perform a single run. This situation is commonly encountered in product design involving, i.a., Computational Fluid Dynamics (CFD) and Finite Element Methods (FEM). Multi-objective (or multi-task) optimisation has gained a lot of attention in engineering optimisation as product design inherently involves trade-offs as typically several (conflicting) aspects are involved. Frequently used are hypervolume-based acquisition functions such as Hypervolume Expected Improvement [5] or Hypervolume Probability of Improvement (HvPoI) [6], assuming a Gaussian Process (GP) prior for each objective. More recently, a Multi-objective version of the Predictive Entropy Search has been proposed [7].

Naturally, the correctness of the approximation of the objective(s) is crucial to perform successful optimisation. Erroneous model fits lead to selection of new evaluations based on false beliefs making the discovery of optima unlikely, especially when the input space is large. While GPs have received much attention both as a modelling strategy and within Bayesian optimisation, recently Student-t Process (TP) process priors have been proposed [8–11] for use with EI. In this contribution we consider the TP prior in the context of multi-objective Bayesian optimisation, and develop an analytical expression of the HvPoI acquisition function for it accordingly. TPs have shown to be promising, and their properties such as additional flexibility and enhanced predictive variance seem to be appealing properties for Bayesian optimisation. A brief overview of TPs is given in Appendix A.2, and the formulation of HvPoI assuming TP priors for the objectives is given in Appendix A.3. The performance of this approach is then compared to HvPoI with GP priors in Appendix A.4 on a multi-objective problem.

A.2 Student-t processes

Given a d-dimensional input space $\mathcal{X} \subset \mathbb{R}^d$, f is a Student-t process with degrees of freedom $\nu > 2$, a continuous mean function γ and a parametrized kernel function k. For any set $X \subset \mathcal{X}$ of n inputs $\boldsymbol{x}$, the (noisy) observations of the mapping of these inputs by f is distributed according to a Multivariate Student-t distribution (MVT): $\boldsymbol{y} \sim \text{MVT}_n(\nu, \boldsymbol{\gamma}, \boldsymbol{K} + \sigma_n^2 \boldsymbol{I})$ with $K_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. The likelihood corresponds to the probability density function of an MVT:

$$p(\boldsymbol{y}|X, \nu, \boldsymbol{\theta}) = \frac{\Gamma\left(\frac{\nu+n}{2}\right)}{\left((\nu-2)\,\pi\right)^{\frac{n}{2}}\,\Gamma\left(\frac{\nu}{2}\right)}|\boldsymbol{K}|^{-1/2}\left(1 + \frac{\beta}{\nu-2}\right)^{-\frac{\nu+n}{2}}, \tag{A.1}$$

with $\beta = (\boldsymbol{y} - \boldsymbol{\gamma})^T K (\boldsymbol{y} - \boldsymbol{\gamma})$. Shah et al. [11] have shown that considering $\boldsymbol{y}|\sigma \sim \mathcal{GP}\left(\gamma, (\nu-2)\sigma\right)$ and marginalizing σ out assuming an *inverse Wishart process* prior, recovers Equation (A.1).

For an arbitrary $\boldsymbol{x}_\star \in \mathcal{X}$ the predictive distribution is also a MVT:

$$\begin{aligned} f(\boldsymbol{x}_\star)|X, \boldsymbol{y}, \boldsymbol{\theta}, \nu &\sim \text{MVT}_1\left(\nu + n, \gamma(\boldsymbol{x}_\star), s^2_{\text{tp}}(\boldsymbol{x}_\star)\right), \\ s^2_{\text{tp}}(\boldsymbol{x}_\star) &= \frac{\nu + \beta - 2}{\nu + n - 2} s^2(\boldsymbol{x}_\star). \end{aligned} \tag{A.2}$$

The quantities μ and s^2 are identical to the predictive mean and variance of a GP (assuming the same kernel and parameters). Recent work also shows marginalizing the output scale also yields a related MVT predictive distribution [12, 13]. This differs from non-analytical marginalization of the kernel lengthscales with Markov chain Monte Carlo methods as applied frequently in Bayesian optimisation (see [14] for a comparison of the latter with traditional maximum likelihood estimates).

A fundamental difference is observed in Equation (A.2): the variance prediction includes the observed responses, as opposed to GP which only considers the space between inputs. This allows a TP to anticipate changes in covariance structure. Furthermore it was proven that a GP is a special case of a TP, with $\nu \to \infty$. However, the approach applied for GPs to include noise as part of the likelihood can not be applied for TPs, as the sum of two independent MVTs is not analytically tractable. Instead, a diagonal white noise kernel is added to allow approximation of noisy observations.

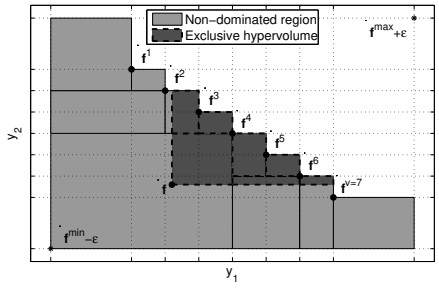


Figure A.1: Pareto set: Illustration (members illustrated by $\boldsymbol{f}^i$) with two objective functions. $\boldsymbol{f}^{\min}$ and $\boldsymbol{f}^{\max}$ denote the ideal and anti-ideal point respectively. The shaded areas (both light and dark) represent the non-dominated region and is decomposed into q cells by a binary partitioning procedure. These cells provide integration bounds to compute $I(\boldsymbol{f}, P)$. Courtesy of Couckuyt et al. [15].

A.3 Hypervolume-based probability of improvement

Given a multi-objective (or multi-task) deterministic optimisation problem, each evaluated input $\boldsymbol{x}_i$ has q observed reponses $\boldsymbol{f}_i = [f^{(1)}(\boldsymbol{x}_i), ..., f^{(p)}(\boldsymbol{x}_i)]$. The observed responses together form a matrix $\boldsymbol{F} \in \mathbb{R}^{n \times q}$. The rows of this matrix correspond to points in the q-dimensional objective space. Of interest are the non-dominated solutions forming the Pareto set $P \subset \boldsymbol{F}$. Ideally, we like to find the following point :

$$\hat{\boldsymbol{x}} = \max_{\boldsymbol{x} \in \mathcal{X}} I(\boldsymbol{f}, P),$$

with $\boldsymbol{f} = f(\hat{\boldsymbol{x}})$ and $I(.)$ represents the improvement function which is defined in this work using the hypervolume indicator as,

$$I(\boldsymbol{f}, P) = \begin{cases} \mathcal{H}(P \cup \boldsymbol{f}) - \mathcal{H}(P) & \boldsymbol{f} \in \mathcal{D} \\ 0 & \text{otherwise.} \end{cases}$$

with $\mathcal{D}$ the non-dominated section of the objective space, $\mathcal{H}(.)$ defined as the hypervolume of the section of the objective space dominated by the Pareto front (bounded by a reference point $\boldsymbol{f}^{\max}$).

The situation is illustrated in Figure A.1: the exclusive (or contributing) hypervolume corresponds to $\mathcal{H}(P \cup \boldsymbol{f}) - \mathcal{H}(P)$. Because $\boldsymbol{f}$ is a (black-box) mapping of q objective functions $f^{(i)}$ of an unknown $\boldsymbol{x}$, and because each evaluation is expensive, direct application of traditional numerical optimisation methods is infeasible. Instead, we approximate each f and optimize an acquisition function incorporating the information provided by the predictive distributions of the approximations of

the objectives. The optimum of the acquisition yields a candidate $\hat{\boldsymbol{x}}$ to be evaluated on all $f^{(i)}$.

We propose the *Hypervolume Probability of Improvement* as proposed earlier by Couckuyt et al. [15] as it is tractable and scales to a higher number of objectives, however we assume each $f^{(i)} \sim \mathcal{TP}$ instead of a GP. Formally, this acquisition function is defined as

$$\begin{aligned} \alpha_n^{\text{HvPoI}}(\boldsymbol{x}) &= I(\boldsymbol{\gamma}, P)p(\boldsymbol{x} \in \mathcal{D}) \\ \boldsymbol{\mu} &= [\mu^{(1)}(\boldsymbol{x}), ..., \mu^{(q)}(\boldsymbol{x})]. \end{aligned}$$

The latter term of the multiplication represents the probability a new point is located in $\mathcal{D}$ and, hence, requires an integration over that region. Exact computation of this integral is performed by decomposing $\mathcal{D}$ into z cells spanned by upper and lower bounds $[\boldsymbol{l}^k, \boldsymbol{u}^k]$. This decomposition can be done using a binary partition algorithm (which scales poorly as the number of objectives grows) as illustrated in Figure A.1, or by applying faster algorithms such as the Walking Fish Group [16]. We then make use of the predictive distribution of the TPs:

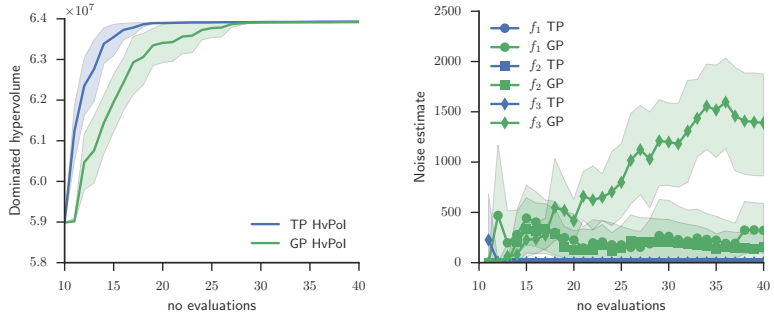
$$p(\boldsymbol{x} \in \mathcal{D}) = \sum_{k=1}^{z} \prod_{j=1}^{q} \left(\Phi_{\nu+n} \left(\frac{u_j^k - \mu^{(j)}(\boldsymbol{x})}{s_{\text{tp}}^{2,(j)}(\boldsymbol{x})} \right) \Phi_{\nu+n} \left(\frac{l_j^k - \mu^{(j)}(\boldsymbol{x})}{s_{\text{tp}}^{2,(j)}(\boldsymbol{x})} \right) \right).$$

In this context, Φ_ν represents the cumulative density function of a $\text{MVT}_1(\nu, 0, 1)$. In addition, we can simply compute the volume of the exclusive volume using the existing q cells with no extra computation as follows (assuming $\boldsymbol{\mu}$ is non-dominated):

$$\mathcal{H}(P \cup \boldsymbol{\mu}) - \mathcal{H}(P) = \sum_{k=1}^{z} \prod_{j=1}^{q} \left(u_j^k - \left(\max l_j^k, \mu_j(\boldsymbol{x}) \right) \right).$$

A.4 Illustration

We illustrate the effectiveness of the TP prior on the DTLZ1 function, including 6 input parameters and 3 output parameters. The function itself is computed analytically, with some mild Gaussian noise added. Couckuyt et al. [15] report difficulties approximating the first objective, hence we try the traditional HvPoI in combination with GP priors, and compare it with the modified version as introduced in Appendix A.3 with TP priors. The initial set of data points consists of an optimised Latin Hypercube of 10 points. The acquisition function is then permitted to select an additional 30 data points for evaluation. For both TP and GP, the Radial Basis Function (RBF) kernel was used, and the hyperparameters $\boldsymbol{\theta}$ including ν were



(a) *(b)*

Figure A.2: DTLZ1 function: (a) Comparison of the growth of the dominated hypervolume for the DTLZ1 function, for 10 experiments using both GP and TP priors for the objectives. The mean and 95% confidence intervals are shown. (b) The noise parameter for all three objectives approximated by GP and TP. For GP, the noise is part of the likelihood whereas for TP a diagonal matrix was added to the kernel matrix. Clearly, the TP are more flexible and do not consider the evaluated data noisy.

optimised with multi-start Sequential Quadratical Programming (SQP). Note that the optimisation can result in a very large value ν, causing the TP to become a GP. Hence, we expect better or equal performance, not worse. Both experiments were repeated 10 times.

As performance metric, the hypervolume indicator (size of the dominated hypervolume with respect to a fixed reference point $\boldsymbol{f}^{\max} = [400, 400, 400]$) is recorded after every function evaluation. The average hypervolume and 95% confidence intervals were computed and plotted in Figure A.2a. Clearly, the runs using the TP approximations of the objectives obtain larger hypervolumes faster. The GP experiments lag behind although they also eventually manage to obtain the same hypervolume indicator performance after additional evaluations. In the end, TPs are able to find a decent hypervolume in about 30% of the function evaluations needed by the GPs for the same hypervolume indicator performance.

Closer investigation reveals the GP approximations for some of the objective functions have large noise levels, varying significantly as more evaluations are added, whereas the TPs do not as illustrated in Figure A.2b. It seems the GP is not flexible enough to approximate the objective functions and has to increase the noise variance to avoid ill-conditioning of the kernel matrix. The TPs compensate for this by decreasing the degrees of freedom, which also affects the prediction variance resulting in better selection of evaluation candidates.

A.5 Conclusion

Student-t processes present themselves as an appealing alternative for Gaussian Processes in the context of Bayesian Optimisation. Their robustness was proven earlier by Shah et al. [11] and their enhanced prediction variance can make them more informative for acquisition functions leading to faster discovery of optima. We demonstrated this on a multi-objective optimisation problem, using an adapted HvPoI criterion.

To make better use of the enhanced prediction variance we aim to adapt the Hypervolume Expected Improvement in further work, as the HvPoI acquisition function does not consider the improvement part of the integration [15]. In addition other acquisition functions can be modified to be used with TPs, although for some of the more complex acquisition functions the Student-t distribution might introduce tractability challenges. We will be looking at TPs for multi-objective optimisation as in [7]. Objectives can then be evaluated independently depending on the expected information gain of each.

A.6 References

[1] J. van der Herten, I. Couckuyt, and T. Dhaene. "Hypervolume-based Multi-objective Bayesian Optimization with Student-t Processes". *BayesOpt 2016 Workshop at the 30th Conference on Neural Information Processing Systems*. Barcelona, Spain, 2016.

[2] J. Snoek, H. Larochelle, and R. P. Adams. "Practical Bayesian optimization of machine learning algorithms". *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger. Lake Tahoe, Sierra Nevada, USA: Curran Associates, Inc., 2012, pp. 2951–2959.

[3] K. Swersky, J. Snoek, and R. P. Adams. "Freeze-thaw Bayesian optimization". *arXiv preprint arXiv:1406.3896* (2014).

[4] D. R. Jones, M. Schonlau, and W. J. Welch. "Efficient Global Optimization of Expensive Black-Box Functions". *J. of Global Optimization* 13.4 (1998), pp. 455–492. ISSN: 0925-5001.

[5] M. T. M. Emmerich, K. C. Giannakoglou, and B. Naujoks. "Single- and multiobjective evolutionary optimization assisted by Gaussian random field metamodels." *Evolutionary Computation, IEEE Transactions on* 10.4 (2006), pp. 421–439. ISSN: 1089-778X.

[6] I. Couckuyt, D. Deschrijver, and T. Dhaene. "Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization". *Journal of Global Optimization* 60.3 (2014), pp. 575–594. ISSN: 0925-5001.

[7] D. Hernández-Lobato, J. M. Hernández-Lobato, A. Shah, and R. P. Adams. "Predictive Entropy Search for Multi-objective Bayesian Optimization". *Proceedings of the 33rd International Conference on Machine Learning (ICML-16)*. Ed. by M.-F. Balcan and K. Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. Manhattan, New York, USA: PMLR, 2016, pp. 1492–1501.

[8] A. O'Hagan. "Bayes–hermite quadrature". *Journal of statistical planning and inference* 29.3 (1991), pp. 245–260.

[9] A. O'Hagan, J. O. Berger, M. C. Kennedy, and J. E. Oakley. *Uncertainty Analysis and other Inference Tools for Complex Computer Codes*. Ed. by J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith. 1998.

[10] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. MIT Press, 2006. ISBN: 026218253X.

[11] A. Shah, A. G. Wilson, and Z. Ghahramani. "Student-t Processes as Alternatives to Gaussian Processes." *AISTATS*. Proceedings of Machine Learning Research. PMLR, 2014, pp. 877–885.

[12] R. B. Gramacy and D. W. Apley. "Local Gaussian process approximation for large computer experiments". *Journal of Computational and Graphical Statistics* 24.2 (2015), pp. 561–578.

[13] S. Montagna and S. T. Tokdar. "Computer Emulation with Nonstationary Gaussian Processes". *SIAM/ASA Journal on Uncertainty Quantification* 4.1 (2016), pp. 26–47.

[14] J. van der Herten, I. Couckuyt, D. Deschrijver, and T. Dhaene. "Fast Calculation of the Knowledge Gradient for Optimization of Deterministic Engineering Simulations". *Submitted to the Journal of Machine Learning Research (JMLR)* (2017).

[15] I. Couckuyt, T. Dhaene, and P. Demeester. "ooDACE Toolbox: A Flexible Object-Oriented Kriging Implementation". *Journal of Machine Learning Research* 15 (2014), pp. 3183–3186.

[16] L. While, L. Bradstreet, and L. Barone. “A fast way of calculating exact hypervolumes”. *Evolutionary Computation, IEEE Transactions on* 16.1 (2012), pp. 86–95.

"Logic will get you from A to Z; imagination will get you everywhere."

— Albert Einstein

B

Flexibility Forecasting

J. van der Herten, F. Depuydt, L. De Baets, D. Deschrijver, M. Strobbe, C. Develder, T. Dhaene, R. Bruneliere, and J.-W. Rombouts.

***International Energy Conference (ENERGYCON)*. Leuven, Belgium: Institute of Electrical and Electronics Engineers, Inc, 2016, pp. 1–6**

Abstract *Power-intensive industry plays a key role in balancing supply and demand in the energy grid: by offering flexible power, industry can reduce operating costs and grid operators can avoid technical failures. Recently, research has started to try and address the challenging question of determining the amount of power curtailment (i.e., how much power can be reduced for how long) without violating any process constraints. We consider several machine learning methods to assess the curtailment potential in a coldstore, based on historical data.*

B.1 Introduction

Due to the deregulation of energy markets and the increased production of energy by renewable sources, grid operators worldwide are in need of *flexible* power to balance supply and demand at any time and to avoid technical failures. Although interesting for stability and usability of renewable sources [2], storing power when supply exceeds demand is very costly at this point and therefore economically infeasible. A lot of effort has been put into providing flexibility on the supply side of the grid by introducing versatile solutions such as gas plants that can be enabled quickly when demand rises and might exceed the supply.

Nowadays, regulating the demand side as a function of fluctuations in the power supply is gaining more attention since balancing power consumption can also help to reduce power consumption peaks [3]. A possible example is the regulated and coordinated charging of electric vehicles [4].

Power-intensive industry plays a big role on the demand side, and has the potential to offer flexible power to the grid by optimizing their day-to-day operations. Power supply for processes that have a certain type of buffer capacity can be stopped when there is no immediate negative impact in terms of output, stability or quality of the process. By resuming power supply to these processes when the load on the grid is lower, power consumption is shifted in time to a period with excess capacity, thus avoiding a peak load. This time shift also holds benefits for industry in terms of monetisation, due to a lower consumption when power is very expensive.

In order to exploit this flexible power, this paper proposes an innovative implementation for the assessment and modelling of power flexibility of a step-by-step approach which is intellectual property of Restore NV [5]. The novel implementation can be used to assess and quantify energy flexibility of an industrial process at any time and was developed as part of the MonIEFlex ICON research project [6]. It was tested on a real-world coldstore application consisting of several compressors (consuming power) and three different rooms. In Appendix B.2, the concept of power flexibility is first formally introduced, and related to earlier work. The collection of the data, and the required processing steps are described in Appendix B.3. Appendix B.4 explains the application of machine learning to build a predictive model to predict how long power can be curtailed without affecting the process negatively (the time to reach the *boundary condition*). The resulting models are then analysed to identify the impact of the parameters on the predictions in Appendix B.5.

B.2 Power flexibility

In order to exploit flexible power of industrial processes to improve demand scheduling, the grid operator has to quantify flexible power. Formally this can be expressed as

$$F = \int_0^{\Delta t} \Delta P(t)\, dt\,. \tag{B.1}$$

This represents the size of the power curtailment $\Delta P > 0$ as a function of time, and the time Δt it is imposed. Assuming ΔP is constant Equation (B.1) simplifies to

$$F = \Delta P\, \Delta t.$$

The time to reach a critical boundary is affected by internal process values and configuration parameters of the process, as well as external influences such as the weather. In case of the coldstore application, ΔP represents the amount of power saved by disabling the cooling installation. This quantity is known and can be directly observed and modified as it is part of the process control. Obtaining Δt is more complex, as it is influenced by the state of the process and environmental parameters. Certain state parameters of the process, such as temperature, should not exceed certain *boundary conditions* if ΔP power is curtailed for a time interval: we require that the power curtailment does not impact the process in a negative way (loss of quality, losses, etc.). To determine the total flexiblity, Δt represents the time to reach the boundary condition and should not be exceeded.

In [7], an approach is described to predict the temperature evolution of domestic fridges. Cooling cycles are predicting by matching cycles observed earlier. The temperature of a domestic fridge is defined mostly by the working of its thermostat: the evolution is very periodical and only interrupted by opening doors, which typically occurs only sporadically. An industrial cooling system, however, results in more complex interactions, making the approach of [7] unreliable under this setting.

A different approach to obtain Δt, is to model all physical relationships with Computational Fluid Dynamics (CFD) simulations with thermal components, using for instance the FloTHERM package [8]. Although these evaluations can be computationally expensive, this method can accurately model the evolution of the temperature. The downside of this approach is it requires a full study of the physical relationships of the process, which makes the solution very specific. Furthermore, it is not always fully known a priori what environmental parameters influence the process: for instance during the MonieFlex project, it was found that for the coldstore case Δt was much more sensitive to the weather than was initially anticipated, even by domain experts.

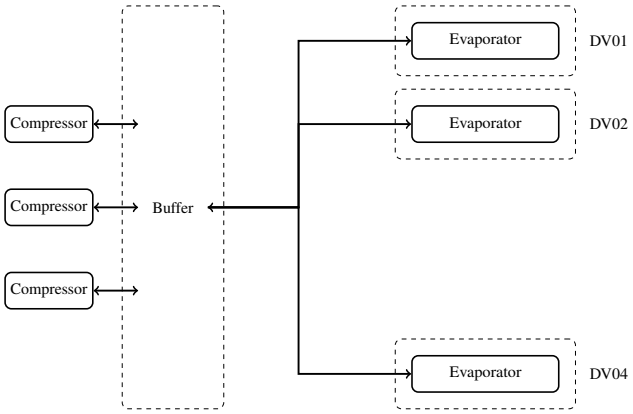


Figure B.1: Coldstore: schematic illustration of the cooling installation.

In this paper an alternative approach is proposed: historical data of the coldstore is collected, analysed and processed to generate a training set for machine learning methods that learn the relationship between its inputs (the state, decision and environmental parameters) and Δt (the output). The resulting predictive model can be used to assess instantaneous flexibility and has the additional benefit that sensitivity analysis can be performed to study the impact of the changes in the inputs to the output. Although this paper focusses on coldstores, this approach is general and can also be applied to different types of industrial processes with minor modifications, i.e., adapt the processing step slightly to compute Δt from the historical data to generate the training sets for the machine learning step.

B.3 Data collection and processing

A schematic overview of the coldstore is given in Figure B.1. Several compressors (consuming $450\,\mathrm{A}$, $370\,\mathrm{A}$ and $200\,\mathrm{A}$ respectively) fill an ammonia buffer. This buffer is consumed by an evaporator in each of the three coldstores (DV01, DV02, DV04). Two coldstores are physically adjacent (DV01 and DV02) whereas DV04 is located in a separate building, as illustrated in Figure B.2.

The compressors can either be on or off, whereas the evaporators can also work at half power. The buffer capacity is small: when the evaporators are running without the compressors enabled, all cooling buffer capacity is consumed within two minutes. On the other hand, enabling a single compressor without any of the evaporators running builds up the buffer capacity in less than 5 minutes.

Throughout a year, all process-related parameters relevant to estimate the power

Table B.1: Coldstore: recorded parameters

Type	Parameters	Symbols	Unit
State parameters $\boldsymbol{V_s}$	Coldstore temperatures	$T_{\text{DV01}}, T_{\text{DV02}}$ T_{DV04}	°C
Decision parameters $\boldsymbol{V_d}$	Compressor states Evaporator states	$C_{\text{A}}, C_{\text{B}}, C_{\text{C}}$ $E_{\text{DV01}}, E_{\text{DV02}}$ E_{DV04}	A on/off
Environmental parameters $\boldsymbol{V_e}$	Wind speed	W_s	m s^{-1}
	Wind direction	W_{dir}	°
	Humidity	H	%
	Outside temperature	T_{ext}	°C

flexibility were monitored and collected once every minute at the coldstore. An overview is given in Table B.1. The buffer responds almost instantaneously, so its state can be neglected. The result is a historic set of data represented as a matrix $\tilde{\boldsymbol{V}} \in \mathbb{R}^{n \times d}$. The columns of the matrix are vectors that comprise the evolution of d parameters over time $t = 0, ..., n-1$.

In order to build a statistical model that predicts Δt, $\tilde{\boldsymbol{V}}$ needs to be processed in two consecutive steps:

1. Obtain the *boundary conditions* $\boldsymbol{V_s}^{\pm}$ for the state parameters $T_{\text{DV01}}, T_{\text{DV02}}, T_{\text{DV01}}$ based on their distribution.
2. For each coldstore x, process $\tilde{\boldsymbol{V}}$ to obtain a dataset consisting of records of the form $(T_x, \boldsymbol{V_d}, \boldsymbol{V_e}, \Delta t)$

The motivation for these steps is given in Appendices B.3.1 and B.3.2.

B.3.1 Identification of boundary conditions

For the coldstore case, the three temperature parameters for the corresponding coldstores $T_{\text{DV01}}, T_{\text{DV02}}, T_{\text{DV04}}$ are subject to constraints: the coldstores must always remain below a certain threshold, to prevent quality decay of products stored within. For automated derivation of the boundary conditions $\boldsymbol{V_s}^{\pm}$ of the state variables ($\boldsymbol{V_s}$), Self-Organizing Map (SOM) [9] were applied, using the implementation available in the SOM-Toolbox [10]. The algorithm trains a set of d-dimensional feature vectors (denoted as $\{\boldsymbol{s}_1, ..., \boldsymbol{s}_r\} \subset \mathbb{R}^d$) to fit the topology of $\boldsymbol{V_s}$. The number of nodes r and its structure are automatically determined based on the amount of

data vectors and the principal eigenvectors of the supplied data. After training, we choose the boundaries corresponding to the interval defined by the minimum and maximum value of the feature vectors $\boldsymbol{s}$ for that dimension. Figure B.3 illustrates how a SOM is trained to a two-dimensional test dataset, and how the boundaries are derived from the SOM.

Using the described approach, it is found that a critical upper bound for the coldstore temperature in DV01, DV02 and DV04 is $-18\,^{\circ}\mathrm{C}$. The engineers working at the coldstore factory confirmed this result. Based on tests performed on different data sets, it was found that the approach based on SOMs gave better results on data with a split distribution or correlations compared to simpler approaches such as choosing the 2.5%-97.5% percentile range.

B.3.2 Computing time to reach boundary condition Δt

In order to train a machine learning model to predict the time to reach the boundary constraints, this value must first be extracted from the recorded data. Unfortunately, under normal operating conditions the coldstores remain below the boundary conditions of $-18\,^{\circ}\mathrm{C}$, which means Δt must first be computed. The preprocessing routine described below is applied once to $\tilde{\boldsymbol{V}}$ for all three coldstores. For a coldstore x, all rising segments of the corresponding temperature T are extracted from the time series. The collected segments are further preprocessed to remove the anomalous cases such as missing data, very short rising segments because the evaporators are re-enabled, maintenance etc. A set of data segments is obtained which represent rising temperature curves in the coldstore. According to Newton's law of cooling, the observed temperature values y evolve according to an exponential, so a curve $\tilde{y}$ of the form

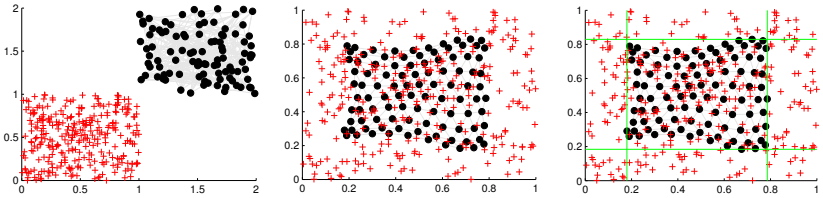
$$\tilde{y}(t) = T_0 \exp\left(\frac{at}{L}\right)$$

is fitted to each segment. T_0 corresponds to the coldstore temperature at the start of the segment, L to the length of the segment. The parameter a is determined through numerical optimisation, minimising the Mean Square Error (MSE) between the curve predictions $\tilde{\boldsymbol{y}}$ and the observed temperatures $\boldsymbol{y}$:

$$\mathrm{MSE}(\boldsymbol{y}, \tilde{\boldsymbol{y}}) = \sum_{i=0}^{n-1} \frac{(y_i - \tilde{y}_i)^2}{n}. \tag{B.2}$$

By extrapolating the exponential curve, the intersection with the boundary condition yields Δt, expressed in minutes. For this procedure, we assumed the outside temperature (which impacts a) remains constant. Given the average and maximum values found for Δt in Table B.2 this seems a reasonable assumption. It was pointed

Figure B.2: Coldstore: satellite image of the industrial site, depicting the three coldstore buildings.



(a) *(b)* *(c)*

Figure B.3: Boundary condition identification: illustration of the SOM-based method to define boundary conditions for parameters, on a uniformly generated 2D dataset. Figure B.3a shows the data (red) and the randomly initialised SOM. After training the SOM has adapted to the data topology as shown in Figure B.3b. The minimum and maximum values of the feature vectors for each parameter are the boundaries as illustrated by the green lines in Figure B.3c.

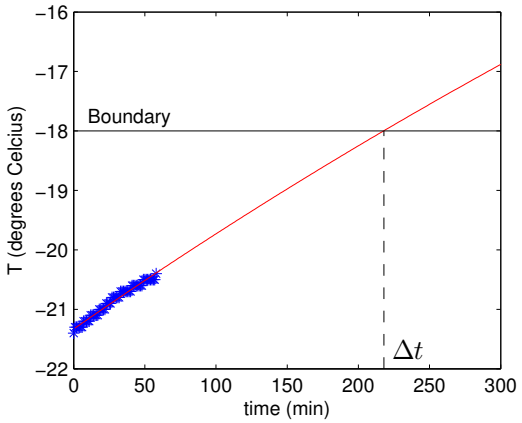


Figure B.4: Computing Δt: illustration of an exponential fit of a rising temperature curve in a coldstore.

out that the temperature of the cargo also influences a: for the curve fitting this has no specific influence as the optimisation determines a for each curve. However, inclusion of this parameter in $\boldsymbol{V_s}$ would account for this source of variability later on during the modelling process. Unfortunately, the temperature of the cargo was not recorded at the factory, and could not be included.

Each segment is then translated into a data point of the form $(T_x, \boldsymbol{V_d}, \boldsymbol{V_e}, \Delta t)$. Further analysis reveals $\boldsymbol{V_d}$ also consists of irrelevant parameters: only if an evaporator is off, a rising temperature is observed. Because of the properties of the installation this implies the corresponding compressors are also switched off (short buffer). Hence, we drop these parameters upfront[1]. Finally $(T_x, \boldsymbol{V_e})$ will be used as model inputs to predict the output Δt. Because DV01 and DV02 are neighbouring coldstores, the temperature of DV02 is also included as additional input parameter for the data points of DV01 and vice-versa, as the temperature of the neighbouring room can potentially influence Δt.

B.4 Modelling Δt

In order to build a model of the Δt, the SUMO toolbox [11] was chosen as experimental platform. It contains state-of-the-art machine learning algorithms and automates the overall model building process.

Each dataset representing one the coldstores was randomly split into a training data

[1] This can be different if a similar approach is used to compute the flexibility of a different type of process.

Table B.2: Data: characteristics

	DV01	DV02	DV04
Data points	46780	48704	15813
Training set	37424	38963	12650
Testing set	9356	9741	3163
$\Delta t_{\max}$ (min)	290	298	300
$\Delta t_{\min}$ (min)	1.25	1	2
Δt_{avg} (min)	68	57	55
Δt_{med} (min)	52	44	42

set (80% of the available entries), and a separate data set for testing the performance of the final models (20% of the entries). This splitting rate is common: the training set is sufficiently large to perform cross-validation, while the test set is still sufficiently large (see Table B.2) to ensure it is representative for normal operating conditions. Several models types were trained for each training dataset: Gaussian Processes (GPs) [12, 13], Support Vector Machine (SVM) [14], Least-Squares Support Vector Machine (LS-SVM) [15], Extreme Learning Machines (ELM) [16] and Artificial Neural Network (ANN). Cross-validation using the MSE error function as defined in Equation (B.2) on the training dataset was chosen as objective function for the hyperparameter optimisation step.

The kernel and regularisation hyperparameters of SVM and LS-SVM were optimised by applying several global optimisation methods: Simulated Annealing (SA) [17], Particle Swarm Optimisation (PSO) [18], Dividing Rectangles (DIRECT) [19] and Pattern Search [20], to select the optimum. For ELM, the number of hidden neurons (up to 5000) and the initialisation range of the weights connecting the input layer to the hidden layer were optimised using the same set of global optimisers. To speed up the matrix inversion for large ELM networks Graphical Processing Unit (GPU) acceleration was used. The architecture and the initial weights of the ANN were considered hyperparameters and optimised using a genetic algorithm (10 generations, 10 individuals each). The training function used was Levenberg-Marquardt backpropagation with Bayesian regularisation (300 epochs) [21].

The best model for each model type was then evaluated on the test set and the Root-Mean-Square Error (RMSE), Average Absolute Error (AAE), Average Relative Error (ARE) and Bayesian Error Estimation Quotient (BEEQ) as given in Table 2.1 were calculated. Computing this combination of error functions allows an in-depth evaluation of the model performance. The BEEQ error measure is less known,

Table B.3: Modelling Δt: errors for the final models obtained for each coldstore. The MSE score is computed using 5-fold cross-validation and minimised during hyperparameter optimisation. All other scores are calculated on the test set.

Model type	MSE	RMSE	AAE	ARE	BEEQ
DV01					
GP	54.5	**3.30**	2.18	**0.08**	0.05
SVM	50.7	3.94	2.27	**0.08**	0.05
LS-SVM	**36.2**	3.63	**2.10**	**0.08**	**0.04**
ELM	1016	31.7	20.7	0.44	0.35
ANN	962	30.8	20.3	0.44	0.34
DV02					
GP	34.4	3.34	2.24	**0.11**	0.07
SVM	33.9	3.46	2.20	0.13	0.06
LS-SVM	**30.5**	**3.31**	**2.08**	0.12	**0.05**
ELM	202	9.40	6.07	0.37	0.15
ANN	643	25.1	16.6	0.46	0.36
DV04					
GP	**63.9**	**4.29**	2.98	**0.17**	0.09
SVM	128	5.30	3.26	0.18	0.09
LS-SVM	68.1	4.83	**2.86**	0.18	**0.07**
ELM	288	13.3	9.62	0.47	0.25
ANN	521	21.5	14.8	0.53	0.36

but related to the well known R^2 or Root-Relative-Square Error (RRSE) [2]. It differs by applying a geometric average instead of an arithmetic average. Like other harmonic error functions such as the Geometric Average Error (GAE) or Harmonic Average Error (HAE), this makes the BEEQ more positive as it is less affected by extremes (which may occur in our case due to working with real data). For additional information on the properties of the error functions used, the reader is referred to [22] and [23].

The results of the modelling experiments are given in Table B.3. For all three coldstores, there is major difference between the performance of the kernel-based methods, as opposed to the parametric methods. The former methods perform very similar, with only minor differences: for DV01 and DV02 LS-SVM seems to obtain a slightly higher accuracy compared to SVM and GP, for DV04 GP seems to perform very well, yet the AAE and BEEQ on the validation set is slightly lower for the LS-SVM model. For DV04, SVM seems to be underperforming on the training data (obtaining a higher cross-validation score), yet the model performs reasonably on the test set, scoring higher errors mostly for longer Δt values (resulting in a higher AAE while retaining a similar value for ARE). Overall, all three kernel based techniques obtain satisfactory results as indicated by low ARE and BEEQ scores, as well as an AAE of about 2 to 3 minutes.

On the other hand, the scores obtained with the network methods are disappointing. Both ELM as well as ANN obtain higher cross-validation scores, which is further reflected in the scores on the validation set. Although still performing better compared to the simplest model available (the mean) as indicated by the BEEQ, the performance is nowhere near to the kernel based results, for all three coldstores. It is possible for different model types to achieve different results, yet the difference between the methods is very significant. A possible reason is a failure of the optimisation methods used to find the optimal network architecture, but it occurs for all three coldstores using two different techniques. Further analysis of the models in the next section provides more insight on the response behaviour, providing additional information on the reason why no good architecture is found.

B.5 Sensitivity analysis

For DV01 and DV02, the LS-SVM models were labeled best, whereas for DV04, the GP model was chosen. Using these models, the Sobol indices [24] were computed to perform sensitivity analysis. This post-processing step provides us information on what parameters (or interactions thereof) influence the output variability the most. Therefore this step assists to understand better what influences

[2] $R^2 = 1 - \text{RRSE}^2$

Δt most and provides relevant information for power flexibility forecasting. Highly sensitive input parameters must be predicted very accurately in order to minimise the forecasting error.

Table B.4 displays the first order Sobol indices (main effects, and total effects) for all three coldstores, as well as the residual[3].

Clearly, variable interactions dominate predictions of the best models. This observation could also partially explain the poor performance of the ANN and ELM models: neurons of these networks take linear combinations of input parameters, and pass these through an activation function. In order to approximate the behaviour of interactions, the output of several neurons needs to be aggregated so the interaction is represented as a series expansion. The Sobol indices indicating the main effects are very low (especially for DV01 and DV02), and the residuals indicate a lot of the output variance is approximated by interactions (for the kernel methods). Computation of the higher order indices revealed that this does not stop at second order or third order: given these complex interactions encountered, quite a large network would be required to capture all output variability which also brings a risk of overfitting. Hence, the optimisation of the network architecture has failed to find a suitable architecture which provides a good approximation but doesn't overfit the training set. Presenting these interaction terms directly as inputs to the network should increase their performance. A second reason for the performance of the parametric methods could be the absence of the temperature of the cargo when the cooling is stopped. This parameter does affect the time constant of the exponential fits, and introduces a source of variability which is not presented as input parameter.

The total indices reveal that besides the compressor and evaporator variables, all other included variables are important and occur in interactions. The only difference observed is situated between DV01 and DV02 on one hand, and DV04 on the other hand. For the latter, the total indices for the weather parameters are significantly lower, and the temperature of the coldstore itself has a much higher impact (as also indicated by the main effect and the residual of DV04). This is consistent with the fact that the north side of the coldstore is better covered by another building and trees, as observed on Figure B.2.

B.6 Conclusion

In this paper we introduced an approach developed for prediction of the time to reach a boundary constraint (Δt) in an industrial coldstore process using machine

[3]The percentage of output variance which is caused by higher order interactions.

Table B.4: Sensitivity analysis: First order Sobol indices

	Indices			Total indices		
	DV01	DV02	DV04	DV01	DV02	DV04
T_{DV01}	0.03	0.02	-	0.75	0.50	-
T_{DV02}	< 0.01	0.08	-	0.78	0.63	-
T_{DV04}	-	-	0.34	-	-	0.61
W_s	< 0.01	0.02	0.12	0.67	0.61	0.41
W_{dir}	< 0.01	< 0.01	0.03	0.62	0.57	0.30
H	< 0.01	< 0.01	0.03	0.61	0.56	0.30
T_{ext}	< 0.01	< 0.01	< 0.01	0.51	0.46	0.11
Residual	0.96	0.86	0.47			

learning, with the goal of computing instantaneous power flexibility. It involves preprocessing historical data to construct a training set, which is then used to learn the relationship between the input parameters and Δt. Application of this approach to a real-world coldstore process indicates Δt can be predicted accurately, but kernel-based methods were better able to capture complex interactions compared to the parametric methods tested in this paper. Further analysis of the final model (for instance using Sobol indices) can then be used to analyse the influence of the input parameters on Δt which can guide the decision making on how to optimise the process environment as well.

The approach can also be used to predict the power flexibility in the future (*forward flexibility*) which is even more important for load scheduling on the power grid. For the coldstore case, this would mostly involve forecasting of environmental parameters (i.e. weather forecasting). Given the coldstore state, the predicted weather values can be used to predict the flexibility for the following hours or days. The prediction would be as reliable as the weather predictions. This can then further be used for optimisation of the power flexibility given the market parameters.

B.7 References

[1] J. van der Herten, F. Depuydt, L. De Baets, D. Deschrijver, M. Strobbe, C. Develder, T. Dhaene, R. Bruneliere, and J.-W. Rombouts. "Energy Flexibility Assessment of an Industrial Coldstore Process". *International Energy Conference (ENERGYCON)*. Leuven, Belgium: Institute of Electrical and Electronics Engineers, Inc, 2016, pp. 1–6.

[2] V. Virasjoki, P. Rocha, A. S. Siddiqui, and A. Salo. "Market Impacts of Energy Storage in a Transmission-Constrained Power System". *Power Systems, IEEE Transactions on* 31.5 (2015), pp. 4108–4117.

[3] A.-H. Mohsenian-Rad, V. W. Wong, J. Jatskevich, R. Schober, and A. Leon-Garcia. "Autonomous demand-side management based on game-theoretic energy consumption scheduling for the future smart grid". *Smart Grid, IEEE Transactions on* 1.3 (2010), pp. 320–331.

[4] K. Mets, T. Verschueren, W. Haerick, C. Develder, and F. De Turck. "Optimizing smart energy control strategies for plug-in hybrid electric vehicle charging". *IEEE/IFIP a network operations and management symposium workshops*. Osaka, Japan: Institute of Electrical and Electronics Engineers, Inc, 2010, pp. 293–299. ISBN: 9781424460373.

[5] J.-W. Rombouts, J. Gheerardyn, P.-J. Mermans, and L. Snijers. "Automated demand response energy management system". 8417391. Apr. 2013.

[6] *iMinds ICON project MonIEFlex*. https://www.iminds.be/en/projects/2014/03/20/monieflex. 2015.

[7] V. Lakshmanan, M. Marinelli, A. M. Kosek, F. Sossan, and P. Norgard. "Domestic refrigerators temperature prediction strategy for the evaluation of the expected power consumption". *4th IEEE/PES Innovative Smart Grid Technologies Europe (ISGT EUROPE)*. Piscataway, New Jersey, USA: Institute of Electrical and Electronics Engineers, Inc, 2013, pp. 1–5.

[8] B. Zahn et al. "Evaluating thermal characterization accuracy using CFD codes-a package level benchmark study of IcePak TM and Flotherm R". *The Sixth Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITHERM)*. Piscataway, New Jersey, USA: Institute of Electrical and Electronics Engineers, Inc, 1998, pp. 322–329.

[9] T. Kohonen. *Self-organizing maps*. 3rd ed. Vol. 30. Springer Series in Information Sciences. Springer-Verlag Berlin Heidelberg, 2001. ISBN: 978-3-540-67921-9.

[10] J. Vesanto, J. Himberg, E. Alhoniemi, and J. Parhankangas. "Self-organizing map in Matlab: the SOM Toolbox". *Proceedings of the Matlab DSP conference*. Vol. 99. 1999, pp. 16–17.

[11] D. Gorissen, K. Crombecq, I. Couckuyt, P. Demeester, and T. Dhaene. "A Surrogate Modeling and Adaptive Sampling Toolbox for Computer Based Design". *Journal of Machine Learning Research* 11 (2010). Available at http://sumo.intec.ugent.be, pp. 2051–2055.

[12] C. E. Rasmussen and H. Nickisch. "Gaussian processes for machine learning (GPML) toolbox". *The Journal of Machine Learning Research* 11 (2010), pp. 3011–3015.

[13] I. Couckuyt, D. Deschrijver, and T. Dhaene. "Fast calculation of multiobjective probability of improvement and expected improvement criteria for pareto optimization". *Journal of Global Optimization* 60.3 (2013), pp. 575–594.

[14] C.-C. Chang and C.-J. Lin. "LIBSVM: a library for support vector machines". *ACM Transactions on Intelligent Systems and Technology (TIST)* 2.3 (2011), p. 27.

[15] J. A. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, J. Suykens, and T. Van Gestel. *Least squares support vector machines*. Vol. 4. World Scientific, 2002.

[16] G.-B. Huang, D. H. Wang, and Y. Lan. "Extreme learning machines: a survey". *International Journal of Machine Learning and Cybernetics* 2.2 (2011), pp. 107–122.

[17] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, et al. "Optimization by simulated annealing". *science* 220.4598 (1983), pp. 671–680.

[18] J. Kennedy and R. Eberhart. "Particle swarm optimization". *Proceedings of the IEEE International Conference on Neural Networks*. IEEE. Perth, WA, Australia: Institute of Electrical and Electronics Engineers, Inc, 1995.

[19] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. "Lipschitzian optimization without the Lipschitz constant". *Journal of Optimization Theory and Applications* 79.1 (1993), pp. 157–181.

[20] C. Audet and J. E. Dennis Jr. "Analysis of generalized pattern searches". *SIAM Journal on Optimization* 13.3 (2002), pp. 889–903.

[21] M. T. Hagan and M. B. Menhaj. “Training feedforward networks with the Marquardt algorithm”. *Neural Networks, IEEE Transactions on* 5.6 (1994), pp. 989–993.

[22] D. Gorissen. “Grid-enabled adaptive surrogate modeling for computer aided engineering”. PhD thesis. Ghent University, 2010, XIV, 362 [in multiple pagination]. ISBN: 9789085783503.

[23] X. R. Li and Z. Zhao. “Evaluation of estimation algorithms part I: incomprehensive measures of performance”. *Aerospace and Electronic Systems, IEEE Transactions on* 42.4 (2006), pp. 1340–1358.

[24] I. M. Sobol. “Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates”. *Mathematics and computers in simulation* 55.1 (2001), pp. 271–280.