Bayesian Modeling for Optimization and Control in Robotics

Bayes'sche Modellierung zur Optimierung und Regelung in der Robotik
Zur Erlangung des akademischen Grades Doktor-Ingenieur (Dr.-Ing.)
genehmigte Dissertation von M.Sc. Roberto Calandra aus Palermo, Italien
Fachbereich Informatik – Intelligent Autonomous Systems
Darmstadt, 2017 – D 17

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Bayesian Modeling for Optimization and Control in Robotics
Bayes’sche Modellierung zur Optimierung und Regelung in der Robotik

Genehmigte Dissertation von M.Sc. Roberto Calandra aus Palermo, Italien

1. Gutachten: Prof. Jan Peters
2. Gutachten: Prof. Michael A. Osborne

Tag der Einreichung: 18 May 2016
Tag der Prüfung: 03 August 2016

Fachbereich Informatik – Intelligent Autonomous Systems
Darmstadt – D 17

Please cite this document with:
URN: urn:nbn:de:tuda-tuprints-58788
URL: http://tuprints.ulb.tu-darmstadt.de/id/eprint/5878

Dieses Dokument wird bereitgestellt von tuprints,
E-Publishing-Service der TU Darmstadt
http://tuprints.ulb.tu-darmstadt.de
tuprints@ulb.tu-darmstadt.de

This publication is licensed under the following Creative Commons License:
Attribution – NonCommercial – NoDerivatives 4.0 International
http://creativecommons.org/licenses/by-nc-nd/4.0/

Erklärung zur Dissertation

Hiermit versichere ich, die vorliegende Dissertation ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.
In der abgegebenen Thesis stimmen die schriftliche und elektronische Fassung überein.

Darmstadt, den 18 May 2016

(M.Sc. Roberto Calandra)

Thesis Statement

I herewith formally declare that I have written the submitted thesis independently. I did not use any outside support except for the quoted literature and other sources mentioned in the paper. I clearly marked and separately listed all of the literature and all of the other sources which I employed when producing this academic work, either literally or in content. This thesis has not been handed in or published before in the same or similar form.
In the submitted thesis the written copies and the electronic version are identical in content.

Darmstadt, 18 May 2016

(M.Sc. Roberto Calandra)

Abstract

Robotics has the potential to be one of the most revolutionary technologies in human history. The impact of cheap and potentially limitless manpower could have a profound influence on our everyday life and overall onto our society. As envisioned by Iain M. Banks, Asimov and many other science fictions writers, the effects of robotics on our society might lead to the disappearance of physical labor and a generalized increase of the quality of life. However, the large-scale deployment of robots in our society is still far from reality, except perhaps in a few niche markets such as manufacturing. One reason for this limited deployment of robots is that, despite the tremendous advances in the capabilities of the robotic hardware, a similar advance on the control software is still lacking. The use of robots in our everyday life is still hindered by the necessary complexity to manually design and tune the controllers used to execute tasks. As a result, the deployment of robots often requires lengthy and extensive validations based on human expert knowledge, which limit their adaptation capabilities and their widespread diffusion. In the future, in order to truly achieve an ubiquitous *robotization* of our society, it is necessary to reduce the complexity of deploying new robots in new environments and tasks.

The goal of this dissertation is to provide automatic tools based on Machine Learning techniques to simplify and streamline the design of controllers for new tasks. In particular, we here argue that Bayesian modeling is an important tool for automatically learning models from raw data and properly capture the uncertainty of the such models. Automatically learning models however requires the definition of appropriate features used as input for the model. Hence, we present an approach that extend traditional Gaussian process models by jointly learning an appropriate feature representation and the subsequent model. By doing so, we can strongly guide the features representation to be useful for the subsequent prediction task.

A first robotics application where the use of Bayesian modeling is beneficial is the accurate learning of complex dynamics models. For highly non-linear robotic systems, such as in presence of contacts, the use of analytical system identification techniques can be challenging and time-consuming, or even intractable. We introduce a new approach for learning inverse dynamics models exploiting artificial tactile sensors. This approach allows to recognize and compensate for the presence of unknown contacts, without requiring a spatial calibration of the tactile sensors. We demonstrate on the humanoid robot *iCub* that our approach outperforms state-of-the-art analytical models, and when employed in control tasks significantly improves the tracking accuracy.

A second robotics application of Bayesian modeling is automatic black-box optimization of the parameters of a controller. When the dynamics of a system cannot be modeled (either out of complexity or due to the lack of a full state representation), it is still possible to solve a task by adapting an existing controller. The approach used in this thesis is Bayesian optimization, which allows to automatically optimize the parameters of the controller for a specific task. We evaluate and compare the performance of Bayesian optimization on a gait optimization task on the dynamic bipedal walker *Fox*. Our experiments highlight the benefit of this approach by reducing the parameters tuning time from weeks to a single day.

In many robotic application, it is however not possible to always define a single straightforward desired objective. More often, multiple conflicting objectives are desirable at the same time, and thus the designer needs to take a decision about the desired trade-off between such objectives (e.g., velocity vs. energy consumption). One framework that is useful to assist in this decision making is the multi-objective optimization framework, and in particular the definition of Pareto optimality. We propose a novel framework that leverages the use of Bayesian modeling to improve the quality of traditional multi-objective optimization approaches, even in low-data regimes. By removing the misleading effects of stochastic noise, the designer is presented with an accurate and continuous Pareto front from which to choose the desired trade-off. Additionally, our framework allows the seamless introduction of multiple robustness metrics which can be considered during the design phase. These contributions allow an unprecedented support to the design process of complex robotic systems in presence of multiple objective, and in particular with regards to robustness.

The overall work in this thesis successfully demonstrates on real robots that the complexity of deploying robots to solve new tasks can be greatly reduced trough automatic learning techniques. We believe this is a first step towards a future where robots can be used outside of closely supervised environments, and where a newly deployed robot could quickly and automatically adapt to accomplish the desired tasks.

Zusammenfassung

Die Robotik hat das Potential eine der revolutionärsten Technologien in der Geschichte der Menschheit zu sein. Die Auswirkungen von günstiger und potenziell unbegrenzter Arbeitskraft könnten tiefgreifenden Einfluss auf unser tägliches Leben und insgesamt auf unsere Gesellschaft haben. Wie von Iain M. Banks, Asimov und vielen anderen Science Fiction Schriftstellern vorhergesehen könnte die Robotik zu dem Verschwinden von körperlicher Arbeit und einer generellen Verbesserung der Lebensqualität führen. Allerdings ist die Massenanwendung von Robotern in unserer Gesellschaft noch weit von der Realität entfernt, außer vielleicht in einigen wenigen Sparten, wie beispielsweise die Fabrikation. Ein Grund für den geringen Einsatz ist, dass trotz der enormen Fortschritte in den Fähigkeiten der Roboter Hardware, entsprechende Fortschritte in der Regelung fehlen. Die Anwendung von Robotern in unserem täglichen Leben ist immer noch beschränkt durch die Notwendigkeit die Regler für die entsprechenden Aufgaben manuell zu entwerfen und anzupassen. Dies führt dazu das der Einsatz von Robotern oft lange und umfangreiche Validierungen durch menschliches Expertenwissen voraussetzt, was ihre Anpassungsfähigkeiten und weitläufige Verbreitung einschränkt. Um in der Zukunft die allgegenwärtige Robotisierung unsere Gesellschaft zu erreichen, ist es notwendig die Komplexität des Anwendens neuer Roboter auf neue Aufgaben zu verringern. Das Ziel dieser Dissertation ist es automatische Softwarewerkzeuge basierend auf Machine Learning Techniken vorzustellen, die den Entwurf von Reglern für neue Aufgaben vereinfachen und optimieren. Insbesondere argumentieren wir, dass Bayesian Modeling ein wichtiges Werkzeug für das automatische Lernen von Modellen durch rohen Daten ist und die Unsicherheit solcher Modelle angemessen erfasst. Daher präsentieren wir einen Ansatz der traditionelle Gaussian Process Modelle erweitert indem er eine angemessene Feature Repräsentation und das darauf folgende Modell gemeinsam lernt. Dadurch können wir die Feature Repräsentation so leiten, dass sie die für die darauf folgende Vorhersage nützlich ist. Eine erste Robotik Anwendung bei der der Einsatz von Bayesian Modeling von Vorteil ist, ist das genaue lernen von komplexen Dynamikmodellen. Für höchst nicht-lineare Robotersysteme, beispielsweise wenn Kontakte berücksichtigt werden müssen, kann das Anwenden von analytischen Systemidentifikation Techniken sehr anspruchsvoll, zeitaufwändig oder sogar unlösbar sein. Wir stellen einen neuen Ansatz für das Lernen von inversen Dynamikmodellen vor, das künstliche taktile Sensoren ausnutzt. Dieser Ansatz erlaubt es unbekannte Kontakte zu erkennen und zu kompensieren ohne eine räumliche Kalibrierung der taktilen Sensoren zu benötigen. Wir nutzen den Humanoiden Roboter iCub um zu zeigen, dass unser Ansatz State-of-the-Art analytische Modelle übertrifft und im Bereich der Regelung die Tracking Genauigkeit significant verbessert. Eine zweite Robotik Anwendung von Bayesian Modeling ist die automatische Black Box Optimierung von Regelparametern. Wenn die Dynamik eines Systems nicht modelliert werden kann (aufgrund von Komplexität oder dem Fehlen einer vollständigen Zustands representation), ist es immer noch möglich eine Aufgabe durch das Anpassen einer existierenden Regelung zu lösen. Der Ansatz in dieser Arbeit ist Bayesian Optimization, was erlaubt die Parameter eines Reglers für eine spezifische Aufgabe zu optimieren. Wir evaluieren und vergleichen die Performanz von Bayesian Optimization anhand einer Gangoptimierung des dynamischen, zweibeinigen Laufroboters Fox. Unsere Experimente heben den Vorteil dieses Ansatzes hervor, indem die Zeit für die Justierung der Parameter von Wochen auf auf einen einzelnen Tag reduziert wird. In vielen Robotik Anwendungen ist es jedoch nicht immer möglich ein einzelnes klares Ziel zu definieren. Meistens sind mehrere gegensätzliche Ziele gleichzeitig wünschenswert, weswegen der Entwickler eine Entscheidung bezüglich des Trade-offs zwischen diesen Zielen treffen muss (e.g. Geschwindigkeit vs. Energieverbrauch). Ein Framework das bei diesem Entscheidungsprozess assistiert, ist die Pareto-Optimierung, insbesondere die Pareto Optimalität. Wir schlagen ein neues Framework vor, dass Bayesian Modeling ausnutzt um die Qualität von traditionellen Pareto-Optimierungs Ansätzen ,sogar im Fall von nur wenigen Daten, verbessert. Durch das Entfernen der irreführenden Effekte von stochastischem Rauschen, wird dem Entwickler eine genaue und kontinuierliche Pareto-Front präsentiert von dem der gewünschte Trade-off gewählt werden kann. Zusätzlich erlaubt unser Framework die nahtlose Einführung von mehreren Robustheitsmetriken, die während der Entwurfsphase berücksichtigt werden können. Diese Beiträge erlauben eine beispiellose Unterstützung bei dem Entwurfsprozess von komplexen Robotersystemen mit mehreren Zielen und insbesondere in Anbetracht von Robustheit. Die gesamte Arbeit in dieser Dissertation demonstriert erfolgreich an echten Robotern, dass die Komplexität des Anwendens von Robotern auf neue Aufgaben durch automatische Lerntechniken enorm vereinfacht werden kann. Wir glauben dies ist ein erster Schritt in eine Zukunft in der Roboter außerhalb von streng überwachten Umgebungen benutzt werden können und in der neu angewendete Roboter sich schnell und automatisch and neue Aufgaben anpassen können um gewünschte Ziel zu erfüllen.

Acknowledgements

Every person is the sum of his/her experience. As such, every book I read and every person who ever interacted with me somehow influenced my research. Each one of these persons was on his/her own influenced by many others before. In a network of "strings and attractors", my thinking has been influenced in ways that I will never fully comprehend. Such is the beauty of the world in which we live, such is the beauty of the complex system called humanity, where everybody is connected. Nonetheless, there are many people I would like to thank directly for their easily quantifiable support and influence during my PhD:

First of all, I would like to thank Jan Peters for all the support and supervision during these years, and for capturing my interest towards robot learning in the first place.

I am grateful to my thesis committee: the external referee Michael A. Osborne, the head of the thesis committee Gerhard Neumann, and to Oscar von Stryk and Felix Wolf for reading this thesis and participating in the defense.

Crucial to all the research presented in this thesis was my (unofficial) instructor Marc P. Deisenroth. Marc continuously inspired me and taught me most of what I learned during my PhD. He spent a remarkable amount of time helping me, and he gave me the much needed gradient ∇ necessary to improve myself and my scientific work. For all this, I will always be grateful – I could not have asked for a better mentor and friend.

Throughout the years, I was always lucky to share the office with interesting and friendly people. In chronological order: Christoph Dann, Filipe Veiga, Adrià Colomé Figueras, Serena Ivaldi, Matteo Pirotta and Takayuki Osa. A special thank to Filipe for making my days in the office very very sweets and full of sugar.

All the co-authors for the beautiful collaborations over the years and in particular: Nakul Gopalan, André Seyfarth, Serena Ivaldi, Carl E. Rasmussen, Philipp Beckerle, Katayon Radkhah-Lens and Elmar Rueckert.

Essential during these years was the support given by Veronika Weber and Sabine Schnitt. They helped me by both guiding me in the maze of the rigid German bureaucracy, and for making me smile when I needed it most.

My family for their unconditional support and love. A special thanks goes to my grandfather Carmelo Tamburello, my uncle Giuseppe Lo Cicero and my father Enrico Calandra. Since when I was a kid, they taught me how to fix electric, electronic and mechanical systems, as well as soldering, drilling and improvise fixes with what you have. Moreover, they taught me to "always double-check and then check again". These teachings proved essential working with robots. Nonno Carmelo also introduced me to my first Robot. While Zio Pini and Zia Anna satisfied the unusual request of a 16-year-old boy of being gifted Marvin's Minsky book "The society of the mind" for Christmas. This book strongly influenced my vision about artificial intelligence and still does nowadays.

All the amazing people I met during my internship at Microsoft Research, and in particular: Andrew Blake, Katja Hofmann and all the AIX team (Matthew, Tim, Dave, Mat, Philipp, Nicole, Diana and Cristina).

The city of Darmstadt, with its beautiful *Jugendstil* landmarks for offering a pleasant and relaxing environment.
The staff of the *Klinikum Darmstadt* for their professional and caring service, and for saving my life.

My wife Radica for being brave and kind.

And finally, I would like to thank all the robots I worked with.
When you will finally conquer the world, please remember of me, and of how many times I repaired you.

Roberto Calandra
Darmstadt, May 2016

Contents

Abstract **i**

Acknowledgements **iii**

Figures and Tables **vi**

List of Abbreviations **xi**

List of Symbols **xiii**

List of Operators **xiv**

1. Introduction **1**
1.1. Motivation 1
1.2. Contributions 3
1.3. Outline 4

2. Bayesian Modeling **6**
2.1. Gaussian Processes 6
2.2. Manifold Gaussian Processes 7
2.2.1. Jointly Learning Feature Space and Gaussian Processes 7
2.2.2. Experimental Results 10
2.2.3. Discussion 13
2.3. Summary 14

3. Learning High-dimensional Robot Inverse Dynamics Models from Artificial Skin **16**
3.1. Introduction 16
3.1.1. Contributions 17
3.1.2. Outline of the Chapter 18
3.2. Background & Related Work 18
3.2.1. Contacts and Artificial Skins 19
3.2.2. Classical Model-based Approaches for Computing the Robot Dynamics 19
3.2.3. Learning the Inverse Dynamics 20
3.3. Mixture-of-Contacts using Gaussian Processes 20
3.3.1. Learning a Mixture-of-Contacts 21
3.3.2. Learning the Gating Network 21
3.3.3. Gaussian Processes as Expert Models 22
3.3.4. Control in Presence of Contacts 23
3.4. Experimental Results 23
3.4.1. Experimental Setting 24
3.4.2. Modeling in presence of JTS 25
3.4.3. Learning the Gating Network 28
3.4.4. Control without JTS 29
3.5. Discussion 32
3.6. Summary 33

4. Bayesian Optimization for Learning Gaits under Uncertainty **34**
4.1. Introduction 34
4.2. Parameter Optimization under Uncertainty in Robotics 35
4.2.1. Parameter Optimization in Robotics 35
4.2.2. Optimization Methods in Robotics 36
4.2.3. Related Work in Robot Locomotion 37

4.3. Introduction to Bayesian Optimization 37
4.3.1. Gaussian Process Model for the Unknown Objective Function 39
4.3.2. Acquisition Functions 39
4.3.3. Optimizing the Acquisition Surface 40
4.4. Evaluation and Comparisons 41
4.4.1. Evaluation on a Stochastic Linear-Quadratic Regulation Task 41
4.4.2. Gait Optimization of a Bio-Inspired Biped 42
4.5. Summary 46

5. Robust Multi-Objective Bayesian Optimization **48**
5.1. Introduction 48
5.2. Background & Related Work 49
5.2.1. Single-objective Bayesian Optimization 49
5.2.2. Multi-objective Optimization 51
5.2.3. Related work 55
5.3. Robust Multi-Objective Optimization 58
5.3.1. A Taxonomy of Robustness 58
5.3.2. Robustness w.r.t. Stochasticity of the Objective Function 60
5.3.3. Robustness w.r.t. Parameter Uncertainty 61
5.3.4. Robust Multi-Objective Bayesian Optimization 64
5.4. Experimental Evaluation 64
5.4.1. Pareto Front Approximation using the Model of the Response Surface 64
5.4.2. Pareto Front Approximation in Presence of Stochastic Noise 65
5.4.3. Robustness w.r.t. Stochasticity of the Objective Function 66
5.4.4. Robustness w.r.t. Parameter Uncertainty 66
5.4.5. Parameter Uncertainty as Robustness to Model Uncertainty 67
5.5. Discussion 68
5.6. Summary 69

6. Conclusion **71**
6.1. Summary 71
6.2. Open Challenges 72
6.3. Publications 74

Bibliography **77**

A. Benchmark Test Functions **91**

B. Curriculum Vitae **95**

Figures and Tables

List of Figures

1.1. Contributions of this thesis. Each circle represent a different chapter and its topic. Bayesian modeling is the core topic of this thesis and is central to all the other topics. 2
1.2. Structure of this thesis and dependencies between the chapters. 3

2.1. Different regression settings to learn the function $f : \mathscr{X} \rightarrow \mathscr{Y}$. (a) Standard supervised regression. (b) Regression with an auxiliary latent space $\mathscr{L}$ that allows to simplify the task. In a full Bayesian framework, $\mathscr{L}$ would be integrated out, which is often analytically intractable. (c) Decomposition of the overall regression task f into discovering a feature space $\mathscr{H}$ using the map M and a subsequent (conditional) regression $G|M$. (d) Our MGP learns the mappings G and M *jointly*. 8
2.2. **Step Function**: (a) Predictive mean and 95% confidence bounds for a GP with SE-ARD covariance function (blue solid), a GP with NN covariance function (red dotted) and a log-sigmoid MGP (green dashed) on the step function of Equation (2.18). The discontinuity is captured better by an MGP than by a regular GP with either SE-ARD or NN covariance functions. (b) The 2D feature space $\mathscr{H}$ discovered by the non-linear mapping M as a function of the input $\mathscr{X}$. The discontinuity of the modeled function is already captured by the non-linear mapping M. Hence, the mapping from feature space $\mathscr{H}$ to the output $\mathscr{Y}$ is smooth and can be easily managed by the GP. 10
2.3. **Multiple Length-Scales**: Intensity map of (a) the considered function, (b) the learned feature space of the MGP with a linear activation function and (c) with a log-sigmoid activation. (d)–(f) The corresponding Spectrum for (d) the original function and the learned feature space for (e) MGP (identity) and (f) MGP (log-sigmoid). The spectral analysis of the original function shows the presence of multiple frequencies. The transformations learned by both variants of MGP focus the spectrum of the feature space towards a more compact frequencies support. 11
2.4. **Bipedal Robot Locomotion**: The bio-inspired bipedal walker *Fox* from which the dataset is generated. . . . 13
2.5. **Bipedal Robot Locomotion**: Predictive mean and 95% confidence bounds on the test set on real robot walking data for (a) GP with SE-ARD covariance function, (b) GP with NN covariance function and (c) MGP (log-sigmoid). The MGP (log-sigmoid) captures the structure of the data better compared to GPs with either SE-ARD or NN covariance functions. 14
2.6. **Bipedal Robot Locomotion**: The MGP learns a smooth feature space representation (the color indicates the phase during a single step). 15

3.1. The humanoid robot *iCub* used in our experiments. The blue artificial skin consists of over 2000 tactile sensors distributed across the whole body. 16
3.2. Illustration of the force/torque and tactile sensors during a contact of the robot arm with the environment. 17
3.3. Arrays of tactile sensors used on the *iCub*: each of the triangular array consists of 10 separate *taxels*. Typically, a lengthy spatial calibration is required after the sensors are mounted on the robot, to identify theirs exact position. 18
3.4. Our approach extends existing inverse dynamics without contacts by learning many contact models which serve as correction terms under different contacts type. The decision of which contact model to activate is taken by a gating network based on the skin measurements $\boldsymbol{s}$, the force torque sensors $\boldsymbol{F}$ and the current state $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$. 20
3.5. **Learning a single contact:** Effects of a contact (green curve) compared to the free movement (blue curve) for a simple PD controller without contact compensation. The contact influence both the joint position (a) and the torque measured by the joint torque sensor (b). 23
3.6. **Learning a single contact**: Comparison of the torques in presence of contact as measured at the elbow by the JTS, estimated by *iDyn*, and by our learned model (shown as mean ± 2 std). Our learned model better predicts the torque measured by JTS (a). Additionally, due to the identification of the noise in the model, its prediction is smoother compared to both the noisy JTS measurements and the prediction from *iDyn*. For visualization purposes we also show the predictions after filtering JTS and *iDyn* (b). 24

3.7. **Robustness of a Single Contact Model:** Effects of the contact on the task space and the torque for the three different environmental objects: far contact, medium contact and close contact. The task without contact is shown as reference (**black dashed curve**). 25

3.8. **Robustness of a Single Contact Model**. The three obstacles (far contact, medium contact and close contact) and the corresponding measurements from the forearm tactile sensors during the contacts. 26

3.9. **Learning Multiple Contacts**. The robot executes a circular trajectory with its left arm. The forearm collides alternatively with three objects causing the right contact, left contact or both contacts. 27

3.10. **Learning Multiple Contacts:** Prediction of torques with multiple contacts and the corresponding activation of the gating network. Our mixture-of-experts model combines the single-contact models into a multiple-contact model. 28

3.11. **Learning the Gating Network:** Classification accuracy (mean ± std) for different types of gating networks. The most accurate gating network was learned using an SVM having as input the full skin/proprioception data $[\boldsymbol{s}, \boldsymbol{F}, \boldsymbol{q}]$. 29

3.12. **Dimensionality Reduction**. Example of activation of the tactile sensors for the forearm and its reconstruction after reducing the dimensionality of the data to 15 using a SAE. Reducing the dimensionality of the data does not significantly influence the quality of the signals and the reconstructed data are indistinguishable from the original ones. 30

3.13. **Validation of the (Contact-free) Inverse Dynamics:** Average tracking error in the absence of contacts. Using the learned IDM plus PD (green curve) reduce the error compared to using the PD controller only (blue curve). We conclude that the learned IDM is meaningful. 31

3.14. **Validation of the (Contact-free) Inverse Dynamics:** Average torque contributed by the PD term $\boldsymbol{\tau}_{PD}$ in absence of obstacle (filtered for visualization purposes). $\boldsymbol{\tau}_{PD}$ drastically decrease when using the learned IDM, therefore suggesting that the learned IDM is accurate. 31

3.15. **Trajectory Tracking in Presence of Contacts:** Average tracking error in presence of obstacle. Our approach using PD + IDM + CM (red curve) outperformed both simple PD controller (blue curve) and PD + IDM (green curve). 31

3.16. **Trajectory Tracking in Presence of Contacts:** Average torque contributed by the contact model term $\boldsymbol{\tau}_{ext}$ in presence of obstacle (filtered for visualization purposes). 31

3.17. **Generalization of the Contact Model**. The three objects used as obstacles and their corresponding weight. The objects also present different tactile feature due to changes in size and material. Object 1 and 2 were used during the training, while Object 3 was used for testing. 32

3.18. **Generalization of the Contact Model:** Average tracking error in presence of a novel obstacle. Our approach using PD + IDM + CM (red curve) outperformed both simple PD controller (blue curve) and PD + ID model (green curve). 33

4.1. The bio-inspired dynamical bipedal walker *Fox*. Using Bayesian optimization, we found reliable and fast walking gaits with a velocity of up to 0.45m/s. 34

4.2. Related work in robot locomotion: Various optimization methods and the corresponding work in robot locomotion where they are applied. 37

4.3. Example of the Bayesian optimization process during the minimization of an unknown 1-D objective function f (red curve). The 95% confidence of the model prediction is represented by the blue area. The model is initialized with 5 previously evaluated parameters $\boldsymbol{\theta}$ and the corresponding function values $f(\boldsymbol{\theta})$. The location of the next parameter to be evaluated is represented by the vertical green dashed line. At each iteration, the model is updated using all the previously evaluated parameters (red dots). Bayesian optimization quickly found the global minimum of the unknown objective function, after a few iterations. . 38

4.4. Average over 50 experiments of best parameters found during the minimization process for a stochastic LQR using Bayesian optimization. The average objective value function (red curve) during the optimization process and the average analytical solution (green dashed line) are shown. 41

4.5. Example of Bayesian optimization for a stochastic LQR. The objective value function (red curve) and the 95% confidence of the model prediction (blue area) are shown during the optimization process, additionally, the analytical solution (green dashed line) is shown as a reference. 42

4.6. The *Fox* controller is a finite state machine with four states. Each of the four joints, left hip (LH), left knee (LK), right hip (RH) and right knee (RK), can perform one of three actions: flexion (Flex), extension (Ext) or holding (Hold). When a joint reaches the maximum extension or flexion, its state is changed to holding. The transition between the states and the control signals applied during flexion and extension are determined by the controller parameters $\boldsymbol{\theta}$. 43

4.7. Hip and knee angle reference frames (red dashed) and rotation bounds (blue solid). The hip joint angles' range lies between 135° forward and 205° backward. The knee angles range from 185° when fully extended to 60° when flexed backward. . . . 43
4.8. The maximum average walking speed of *Fox* evaluated during the gait optimization process. (a) Bayesian optimization performed better than both grid and random search. BO with the GP-UCB acquisition function performed best, achieving a fast and robust gait in less than 30 experiments. (b) Bayesian optimization of various acquisition functions are shown for fixed hyperparameters. Imprecise fixed hyperparameters led to sub-optimal solutions for all the acquisition functions. . . . 44
4.9. Maximum average walking speeds [m/s] found by the different optimization methods. BO using GP-UCB with automatic hyperparameter selection found the best maximum of all methods. The maximum obtained by PI is qualitatively similar. . . . 45
4.10. Two-dimensional slice of the mean of the response surface computed using all the experiments performed. Avoiding the local maxima demands global optimization. . . . 45
4.11. Example of slices of the response surface along each of the parameters. The response surface was trained with the data collected across all the experiments. Only a small zone of each slice leads to a walking gait. The symmetry between parameters 1 and 3 and between parameters 2 and 4 is clear. A shift of about 5 degrees in the optimal parameters is visible due to the constrained circular walk. . . . 46
4.12. Average walking speed during the gait optimization process of *Fox* using Bayesian optimization. For each evaluation the measurement of the objective function $f(\boldsymbol{\theta}^*)$ (red curve) and the corresponding 95% confidence of the model prediction $\hat{f}(\boldsymbol{\theta}^*)$ (blue area) are shown. Three evaluations are used to initialize Bayesian optimization and are not shown in the plot. After 80 evaluations, Bayesian optimization finds an optimum corresponding to a stable walking gait with an average speed of 0.45 m/s. . . . 47

5.1. This article fills a literature gap by providing a framework for robustness and reliability in multi-objective optimization of functions that are expensive to evaluate. For this purpose, we relate to multiple related fields. Classic MOO methods, require thousands of evaluations and do not consider robustness and reliability [Deb et al., 2002b, Zitzler et al., 2001]. Extensions to include robustness require even more evaluations [Singh et al., 2003]. Classic BO consider SOO of expensive evaluations [Jones, 2001]. MOBO such as ParEGO [Knowles, 2006] and EIHV [Emmerich et al., 2008] extended BO to the MOO case, but do not consider robustness. Kuindersma et al. [2013] extended BO to include robustness to measurement noise, but does not consider the presence of multiple objectives. . . . 49
5.2. Sketch of a multi-objective optimization task. The objective functions $f_i(\cdot)$ define (in absence of noise) a non-injective, non-surjective mapping, represented by the blue arrows, from the parameter space *(left)* to the objective function space *(right)*. . . . 51
5.3. **Scalarization vs Pareto optimization:** (a) The trade-off curve is used to visualize the performance of a set of parameters with regards to two objective functions. The (blue area) represents the possible solutions for the benchmark function MOP2 (see Appendix). (b) In scalarization, the problem is optimized by "sliding" the curve defined by the scalarization (green curve) toward its extremes. In this illustration, the PF is not convex, hence, linear scalarization can converge only to one of two possible solutions: The north-west or the south-east extremes. (c) The Pareto front, shown as the (red curve), allows instead a wider range of possible optimal trade-offs. . . . 52
5.4. Example of dominance. Both $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ dominate $\boldsymbol{\theta}_3$. Therefore, $\boldsymbol{\theta}_3$ is sub-optimal. However, neither $\boldsymbol{\theta}_1$ nor $\boldsymbol{\theta}_2$ are dominated by any parameter. Therefore, both $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$ belong to the Pareto front. . . . 54
5.5. The hypervolume is computed as the volume between the PF and the reference point $\mathscr{R}$. . . . 54
5.6. Example of input-dependent stochasticity of the objective function. To different parameters corresponds different levels of noise. . . . 58
5.7. Effects of stochasticity of the objective functions in a MOO task. In the objective function space, the presence of stochastic noise leads to the creation of an hyperrectangle dependent from the covariance matrix of the noise Σ. . . . 58
5.8. **Robustness w.r.t. Stochasticity of the Objective Function:** Between two sets of parameters with the same mean, the one with smallest variance is typically preferable. Depending from the circumstances, even sub-optimal parameters might be preferable to the optimal parameters, if their variance is smaller. . . 60
5.9. **Robustness w.r.t. Parameter Uncertainty:** in presence of perturbations of the parameters, between two sets of parameters with the same mean, the one with largest basin with similar performance is typically preferable. Depending from the circumstances, even sub-optimal parameters might be preferable to the optimal parameters if their basin is larger. . . . 60
5.10. **Robustness w.r.t. Parameter Uncertainty:** . . . 61
5.11. **Robustness w.r.t. Parameter Uncertainty:** . . . 62

5.12. **Pareto Front Approximation using the Model of the Response Surface:** Each plot shows for the MOP2 function the PF from the evaluated randomly chosen parameters (blue curve), the PF from the response surface learned using the evaluated parameters (green curve) and the real PF (red curve). (a) Using only a few evaluations leads to a poor approximation of the real PF. Using the response surface learned from the same evaluations already improves the quality of the approximated PF. (b,c) With more evaluations, the response surface accurately approximates the real PF. In contrast, the PF from the evaluations is still a poor and sparse approximation. 65
5.13. **Pareto Front Approximation using the Model of the Response Surface:** Mean ±2std of the HVR for the MOP2 and the ZDT3 functions varying the number of evaluations. The PFs computed using the plain evaluations (blue curve) even after 1000 evaluations are still not dense nor accurate. As a result, the estimated HVR do still have a visible difference to the ground truth. The use of models (green curve) allows to estimate the PF, and as a consequence the HVR, more accurately even with a small number of evaluations. 66
5.14. **Pareto Front Approximation in Presence of Stochastic Noise:** (a) Estimating the real Pareto front (red curve) exclusively from noisy evaluations leads to an over-optimistic Pareto front (blue curve). Using a response surface, we obtain a better approximation of the Pareto front (green curve) and estimate the 95% confidence of the noise (green area). (b) The poor approximation of the Pareto front from noisy measurements leads to sub-optimal parameters (blue dots). The parameters from the response surface (green dots) closely resemble the parameters of the real Pareto front (red dots). 67
5.15. **Pareto Front Approximation in Presence of Stochastic Noise:** Mean ±2std of the HVR for the MOP2 and ZDT3 functions varying the number of evaluations. The PFs computed using the noisy measurements (blue curve) over-estimate the true PF, and as a result the HVR. The use of a model (green curve) allows to reduce the effects of noise and to estimate the PF, and as a consequence the HVR, more accurately. 68
5.16. **Robustness w.r.t. Stochasticity of the Objective Function:** Trade-off curve and Pareto front (red curve) for the noisy Branin function. 68
5.17. **Robustness w.r.t. Stochasticity of the Objective Function:** Pareto front for the noisy Branin function in the parameters space. 69
5.18. **Robustness w.r.t. Parameter Uncertainty:** MOP2 function (a) Along the Pareto front only the solutions at the two extremities of the front are robust. A further area of stable solutions is visible in the North-East corner, but the values of the objective functions are sub-optimal. (b) The robust solutions along the Pareto front are the two small areas at [-0.7 -0.7] and [0.7 0.7]. The large portions of the parameter space on the corners corresponds to the sub-optimal solutions. 70
5.19. **Robustness w.r.t. Parameter Uncertainty:** RMTP3 function (a). (b) 70
5.20. Robustness to parameters uncertainty for the Branin function to compensate for uncertainty in the model parameters. 70

A.1. MOP2 function. 92
A.2. RMTP3 function. 92
A.3. Branin rcos 1 function. 93
A.4. The Branin function with heteroscedastic noise defined in Equation (A.14). The two surfaces visualize the mean ± std. 93
A.5. Trade-off curve for the Fox dataset from the 1909 evaluations performed on the real bipedal walker. The red datapoints belongs to the PF computed from the evaluations. 93
A.6. The bipedal walker “Fox” used to collect the dataset. 93

List of Tables

2.1. **Step Function**: Negative Log Marginal Likelihood (NLML) and Negative Log Predictive Probability (NLPP) per data point for the step function of Equation (2.18). The MGP (log-sigmoid) captures the nature of the underlying function better than a standard GP in both the training and test sets. 12
2.2. **Multiple Length-Scales**: NLML per data point for the training set and NLPP per data point for the test set. The MGP captures the nature of the underlying function better than a standard GP in both the training and test sets. 12
2.3. **Bipedal Robot Locomotion**: NLML per data point for the training set and NLPP per data point for the test set. The MGP captures the nature of the underlying function well in both the training and test sets. 14
2.4. **Smooth function**: NLML per data point for the training set and NLPP per data point for the test set. There is no relevant difference between MGP and standard GP in modeling smooth functions. 15

3.1. **Learning a single contact:** Mean and standard deviation of the mean for the RMSE of the test set for ground truth, predictions with the *iDyn* and our learned model. The learned model predicts the torques more accurately than *iDyn* for both the full trajectory and during the only contact phase. 24
3.2. **Robustness of a Single Contact Model:** Errors between the ground truth (JTS) and the predictions with either the *iDyn* and our learned model on the test set. A single expert is robust to small variations of the contact. 25
3.3. **Learning Multiple Contacts:** Root mean square error between the ground truth (JTS) and the predictions with the *iDyn* and our learned model on the test set. Our learned model predicts the torque more accurately than *iDyn*. 26
3.4. **Validation of the (Contact-free) Inverse Dynamics:** Tracking error (in degrees) in absence of contacts (mean ± std). The use of the learned IDM proved to be beneficial in reducing the error. 30
3.5. **Trajectory Tracking in Presence of Contacts:** Tracking error (in degrees) in presence of contact (mean ± std). 32
3.6. **Generalization of the Contact Model:** Tracking error (in degrees) in presence of an object never observed before (mean ± std). Even with novel objects our approach improve the performance of the two most used joints (shoulder roll and jaw) compared to simple PD and PD + IDM controllers. 33

4.1. Optimization methods in robotics: Properties of various optimization methods commonly used for optimization in robotics. As discussed in Section 4.2.1, the ideal optimizer for robotic applications should be global, zero-order, and assuming stochasticity. (*) Extensions exist for the stochastic case. (†) First or second order. 36
4.2. Performance of Bayesian optimization compared to the exact solution for the stochastic LQR problem. . . . 41

5.1. **Pareto Front Approximation in Presence of Stochastic Noise:** Mean ±2std of the HVR for the MOP2 and ZDT3 functions varying the number of evaluations. Using the noisy measurements over-estimate the true PF, and as a result the HVR. The use of a model allows to reduce the effects of noise and to estimate the PF, and as a consequence the HVR, more accurately. HVR values closer to 1 are generally better. 65

Abbreviations, Symbols and Operators

List of Abbreviations

ARD Automatic Relevance Determination.

BO Bayesian Optimization.

CM Contact Model.

DOF Degrees of Freedom.

e.g. *Exempli Gratia*.
EGO Efficient Global Optimization.
EI Expected Improvement.
EIHV Expected Improvement of the Hyper-Volume.

FSM Finite State Machine.
FTS Force/Torque Sensor.

GP Gaussian Process.

HGP Heteroscedastic Gaussian Process.
HVR Hyper-Volume Ratio.

ID Inverse Dynamics.
IDM Inverse Dynamics Model.
i.e. *Id Est*.
i.i.d. Independently and Identically Distributed.

JTS Joint Torque Sensor.

LQR Linear Quadratic Regulator.

MCMC Markov Chain Monte Carlo.
MDP Markov Decision Process.
MGP Manifold Gaussian Process.
MOBO Multi-Objective Bayesian Optimization.
MOO Multi-Objective Optimization.

NLML Negative Log Marginal Likelihood.
NLPP Negative Log Predictive Probability.

NN Neural Network.

PCA Principal Component Analysis.

PD Partial-Derivative (Controller).

PF Pareto Front.

PI Probability of Improvement.

RBD Rigid Body Dynamics.

RL Reinforcement Learning.

RMSE Root Mean Square Error.

SAE Stacked Auto-Encoders.

SOO Single-Objective Optimization.

SVM Support Vector Machine.

UCB Upper Confidence Bound.

VHGP Variational Heteroscedastic Gaussian Process.

List of Symbols

Notation	**Description**
C	Coriolis Force
$\boldsymbol{K}$	Covariance Matrix
$\boldsymbol{g}$	Gravity Force
∇	Gradient
$\boldsymbol{\omega}$	Hyperparameters of a (GP) Model
δ	Kronecker delta
$\boldsymbol{M}(\boldsymbol{q})$	Inertia Matrix
$\boldsymbol{\theta}$	Vector of Parameters to be Optimized
$\mathscr{P}$	Set of Pareto-Optimal Parameters
$\boldsymbol{q}$	Joints Position
$\dot{\boldsymbol{q}}$	Joints Velocity
$\ddot{\boldsymbol{q}}$	Joints Acceleration
$\mathbb{R}$	Real numbers
$\boldsymbol{\tau}$	Torques

List of Operators

Notation	Description	Operator
α	Acquisition Function	$\alpha(\bullet)$
k	Covariance Function	$k(\bullet,\bullet)$
$\mathcal{N}$	Gaussian Distribution	$\mathcal{N}(\bullet,\bullet)$
$\ln$	Natural Logarithm	$\ln(\bullet)$
σ	Transfer Function of a Neural Network Layer	$\sigma(\bullet)$
$\mathcal{U}$	Uniform Distribution	$\mathcal{U}(\bullet,\bullet)$

1 Introduction

> In these matters, the only certainty is that nothing is certain
>
> Pliny the Elder

1.1 Motivation

The goal of many researchers in the fields of robotic and artificial intelligence has been for decades to create agents capable of adapting and solving a wide-range of tasks autonomously. However, despite the tremendous scientific advances, the current robots and software agents still possess extremely limited adaptation capabilities. The flexible adaptability that defines our species is still out the reach of our artificial agents.

One reason that explains why we do not currently have robotic assistants in our homes is the increasing awareness of the reality that modeling complex physical systems is hard and time-consuming. Traditional robotics, which requires full analytical models of the system, is not capable of scaling with the innumerable variety of complex stimuli and environments. Even more, it is not capable of dealing with the intrinsic uncertainty that permeate our daily life. Creating models that perfectly represent the reality failed, as it is at least as hard as defining an hyperuranic world, or have an absolute understand of the law that guide our universe.

Throughout this thesis, two speculative philosophical threads have been used as inspiration. The first topic focuses on the benefits of using models for improving adaptability. Arguably, one intrinsic property of humans is the capability of performing predictions. If asked to predict the effects of opening a hand while holding a glass, most people will correctly predict that the glass would (follow the law of gravity,) drop to the floor, and possibly shatter. These predictive models that we possess are ubiquitous (but not necessarily correct) and encompass a large range of process and layers of abstractions, from simple mechanical concepts as cause-effect to predicting emotional response of other humans. While some of such predictive models that humans possess could be explained through innate processes, it is undeniable that most of such models are acquired throughout our life from experience. As an example, the effects of gravity are probably learned through hundreds of thousands of daily experiments that we perform since childhood. The first philosophical position in this thesis is hence that the learning and exploitation of learned models is a first step towards human-like predictive capabilities (and eventually adaptability). The second philosophical argumentation in this thesis is the importance of uncertainty. Throughout our daily life, we are capable of making decisions and performing tasks in presence of incomplete information and/or in stochastic environments. Whereas robots would succumb to such adverse situations, humans thrive and are capable of embracing, controlling and exploiting such property. An important aspect of this thesis has therefore been dedicated to the use of Bayesian approaches in order to capture and quantify the uncertainty in our models.

Going back to less philosophical and more concrete robotic topics, nowadays, in order to accomplish a task, robots still need to be instructed and programmed to do so. This programming heavily relies on the presence of an expert designer whose role is to conceptually disassemble and meticulously define the task. This *deus ex machina* expert is also often tasked with tuning and tweaking the parameters of the robot until it will successfully complete the task. However, when a new robot is built, there is limited information available and the time required to train an expert and subsequently make a robot usable are a real limitation for the fast deployment of robots in real-world application.

The efficiency of classical planning and control scheme, such as torque control and model predictive control [Camacho and Alba, 2013], rely on the existence of a dynamics model that accurately describe the system. A first challenge encountered to perform a task with a robot is the creation of a dynamics model. Traditionally, these dynamics models are created by using a system identification approach [Featherstone and Orin, 2008]. Although this approach works well for rigid industrial manipulators, it is harder to apply it to other robots, such as the cheap robots and the increasingly more common complaint robots. In fact, an analytical model often makes unrealistic assumptions, such as lack of elasticities or friction. Moreover, the system identification procedure necessary to identify the parameters of the analytical model can be very complex and time consuming, and in some cases even infeasible. This is undesirable considering that often even two copies of the same robot present slightly different dynamics due to imperfection in the manufacturing process. Similarly, the replacement of a broken component can significantly change the dynamics, and therefore require a new,

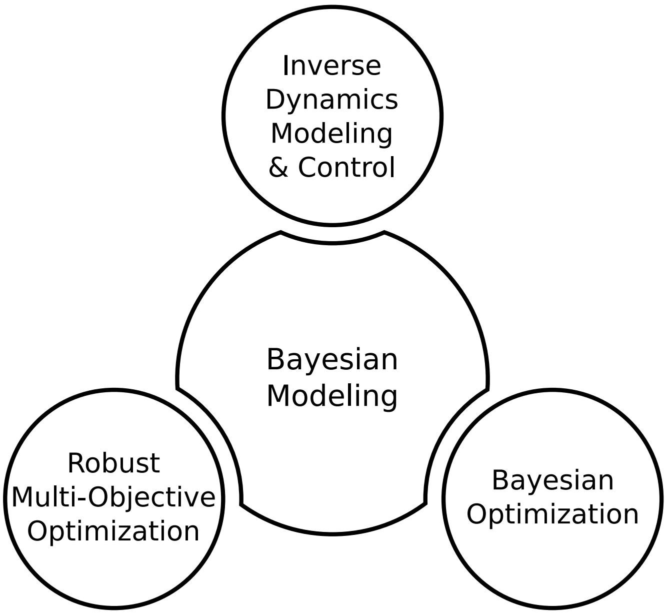

Inverse Dynamics Modeling & Control
Bayesian Modeling
Robust Multi-Objective Optimization
Bayesian Optimization

Figure 1.1.: Contributions of this thesis. Each circle represent a different chapter and its topic. Bayesian modeling is the core topic of this thesis and is central to all the other topics.

lengthy, system identification procedure. An alternative approach to traditional system identification is the use of non-parametric Bayesian models that can be directly learned from raw data [Rasmussen and Williams, 2006]. Learning such models can significantly reduce the time required to create a dynamic model and increase the accuracy of this model, compared to analytical model.

A second challenge encountered to perform a task with a robot is the tuning of the controller parameters. Given an existing controller that is capable of performing a task, it is necessary to find the best parameters to do so. Classically, given an accurate dynamics model these parameters can be computed analytically in closed form. However, for some robots, not all variables that define the full state of a robots are measurable. For these under-sensed robots, planning making use of dynamics model is not always possible, since no accurate full dynamics model can be defined. Therefore, even when a controller capable of performing the task already exists, it is generally not possible to optimize its parameters in closed form. In this case, the use of black-box optimization methods is a suitable alternative for learning to perform tasks. Without the need for a dynamics model, the parameters of the controller can be iteratively optimized with respect to the specified objective function. Crucial for practical black-box optimization methods in robotics is the use of a small number of evaluations. An attractive method for black-box optimization in robotics is Bayesian optimization [Brochu et al., 2010, Kushner, 1964]. In Bayesian optimization, at each iteration a Gaussian Process model is created from the tuple of past parameters evaluated and corresponding measured value of the objective function. Such a surrogate model can then be optimized without need to perform experiments on the robot, and only the optimal parameters are then evaluated on the real robot. Among other favorable properties, Bayesian optimization can naturally deal with stochastic observations. The use of black-box optimization is not limited to under-sensed robots, and it is widely used also for fully sensed robots due to its convenience and insensitivity to inaccuracies in the dynamics model.

A third challenge in robotics is the presence of multiple conflicting objective functions that have to be optimized at the same time. The deployment of most robotic systems requires design choices that will impact the trade-off between multiple objectives. Robotic examples encompass, speed vs energy consumption, tracking accuracy vs compliance, performance vs robustness, and so on. Finding the desired trade-offs without having extensive knowledge about the possible performance of the system is however daunting. It is therefore important to develop automatic tools that extend single black-box optimization methods to encompass the case of multiple objectives and that assist the designer to obtain the desired trade-offs. Of particular relevance to real-world robotic application is the objective of achieving robustness. With robustness, we here mean not only the capability of dealing with the stochastic nature of the task, but also the insensitivity of the performance to changes in the parameters of the system. To achieve these goals, we will make use of Bayesian modeling and of Bayesian optimization.

Tackling these three challenges is the scope of this thesis. Reducing the deployment time of robots in real-world applications, and improving the quality of their solutions is the prize for overcoming them.

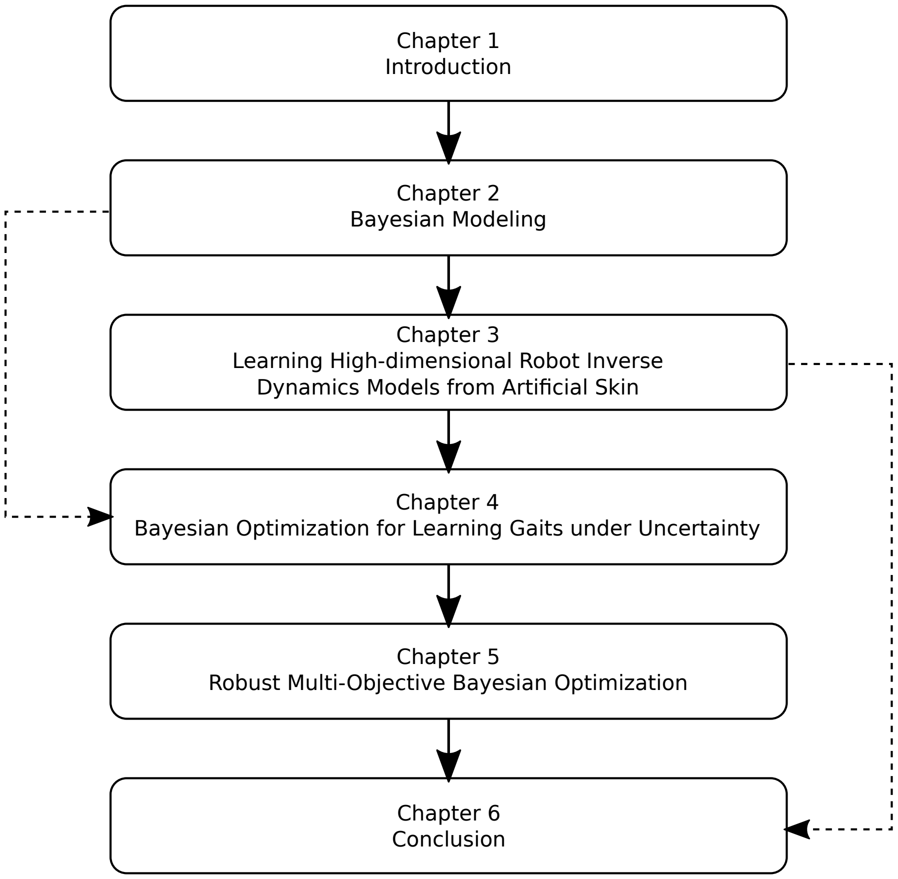

Chapter 1
Introduction
Chapter 2
Bayesian Modeling
Chapter 3
Learning High-dimensional Robot Inverse
Dynamics Models from Artificial Skin
Chapter 4
Bayesian Optimization for Learning Gaits under Uncertainty
Chapter 5
Robust Multi-Objective Bayesian Optimization
Chapter 6
Conclusion

Figure 1.2.: Structure of this thesis and dependencies between the chapters. Chapter 2 introduces Gaussian Process (GP), which is going to be used in all the following chapters, and our novel model the Manifold Gaussian Processes (MGP). Chapter 3 introduces a novel approach to learn inverse dynamics models in presence of contacts using high-dimensional artificial skin sensors. Chapter 4 discusses the use of Bayesian optimization (BO) for learning of complex robotic tasks where no dynamics model is available. In particular, we evaluate BO on automatic gait learning of a dynamic bipedal walker. Chapter 5 presents a framework for robust multi-objective optimization based on Bayesian optimization. Chapter 6 summarizes the contents of this thesis and discuss future research directions.

1.2 Contributions

In this thesis, we introduce new approaches and algorithms for Bayesian modeling tailored to control and optimize tasks in robotics application. The work presented in this thesis contributes to the state-of-the-art in machine learning and robotics. A visualization of the topics threated in this thesis is shown in Figure 1.1.

Manifold Gaussian Processes

Classical Gaussian Process modeling requires the specification of an appropriate covariance function. Often, such specification requires a high degree of prior knowledge about the particular function to be modeled. Additionally, the creation of non-standard covariance functions (e.g., with discontinuities) is challenging and time consuming. In Chapter 2, we propose a novel model, the Manifold Gaussian Processes (MGP), which jointly learn a feature representation (e.g., using a neural network) and a valid GP model. Through the supervised learning of a suitable feature representation, our approach is effectively capable of learning an appropriate covariance function by using standard Likelihood maximization. As a result, MGP can be used to ease the complexity of designing and choosing an appropriate covariance function, and to improve modeling accuracy of challenging highly-non linear functions (such as discontinuities).

Learning High-dimensional Robot Inverse Dynamics Models from Artificial Skin

Many control schema used in robotics requires the existence of accurate dynamics models of the robot. However, analytical models are often not sufficiently accurate, or simply not available. An alternative approach is based on the learning of such models directly from raw data [Nguyen-Tuong and Peters, 2011]. In Chapter 3, we introduce a novel mixture-of-contacts approach to learn inverse dynamics in presence of contacts. The difficulties of this task is twofold: On the one hand, the effects of contacts require modeling discontinuities (which could be possible using MGP models). On the other hand, identifying the presence of contacts requires the use of high-dimensional tactile sensors which is challenging due to the *Curse of Dimensionality*. To overcome these issues, we propose a novel approach based on a mixture-of-contacts model. A gating network uses the raw tactile inputs to identify the ongoing contact, and activates the corresponding model to compensate for the effects of the contact forces. Each of the single contact models is learned independently using GPs. We first validate this approach by comparing its predictive capabilities against analytical models. Following, we demonstrate that this approach improves the performance when used for the control of the humanoid robot *iCub* in presence of unknown contacts. Crucially, the proposed approach does not require a spatial calibration of the tactile sensors.

Bayesian Optimization for Learning Gaits under Uncertainty

Sometimes, a dynamic model of the robot cannot be learned (e.g., for under-sensed robots), or it is impractical to employ model-based control schema. In this case, it is still possible to improve the performance of a controller by tuning its parameters. Manually tuning the parameters is often more an art then a science, which requires expert knowledge about the system and a time consuming trial-and-error process. An alternative approach is the use of automatic optimization techniques. In Chapter 4, we focus on Bayesian optimization (BO), a model-based black-box optimization method [Kushner, 1964, Osborne et al., 2009]. At each iteration, BO fit a Bayesian model to the previous tuples of parameters and corresponding performance. This model is subsequently used to automatically select the next set of parameters according to an acquisition function. We demonstrate the use of BO for automatically tuning the parameters of the dynamic bipedal robot *Fox*. Moreover, we perform an extensive evaluation and comparison of different settings of BO, including multiple acquisition functions. Through the use of BO we eliminated on the *Fox* robot the need for expert knowledge and effectively reduced the tuning time from weeks to a single day of experiments. Additionally, the models that link parameters to performance provide insight into the specific properties of the robot and its controller.

Robust Multi-Objective Bayesian Optimization

The automatic tuning of the parameters of a controller is usually performed w.r.t. a single criteria, such as speed or energy efficiency. However, in most complex tasks, there exists multiple conflicting criteria that is desirable to optimize at the same time. Of these criteria, one of the most important ones for real-world robotic applications is robustness. However, measuring robustness on robotic systems is challenging and often requires an extensive number of experiments. In Chapter 5, we present a novel framework based on BO designed to optimize multiple objectives. We demonstrate that our approach, based on the careful use of Bayesian models, provide numerous advantages compared to the state-of-the-art in Multi-Objective Optimization (MOO). Additionally, we present two measure that exploit Bayesian modeling to estimate robustness without the need to perform additional extensive evaluations.

1.3 Outline

In Chapter 2, we formalize Gaussian processes, a Bayesian regression method used throughout this whole thesis. Following, we introduce MGP, a novel method for jointly learning features and GP models. We evaluate our approach modeling discontinuous functions, such as contacts during the gait of a bipedal robot. A preliminary version of some of the work in this chapter was shown in [Calandra et al., 2016]. The theory was joint work between myself and Marc Deisenroth, based on previous notes of Marc and Carl Rasmussen. I was responsible for the experimental tests. Jan Peters provided some initial problem motivation.

In Chapter 3, we present an approach for learning inverse dynamics models in presence of contacts through the use of tactile sensors. This approach is based on a mixture-of-contact model based on GPs and the superimposition of contacts. We demonstrate our approach on the humanoid robot *iCub*. This chapter is based on [Calandra et al., submitted to the IEEE Transactions on Robotics], and a preliminary version of some of the work in this chapter was shown in [Calandra et al., 2015a,b]. The theory was joint work between myself and Jan Peters. Serena Ivaldi helped with the data collection from *iCubParis01* and with useful tips on the iCub software architecture. I was responsible for the experimental evaluation. Marc Deisenroth provided useful advice about the experimental evaluation.

In Chapter 4, we give an overview on black-box optimization for robotic applications. Following, we focus on Bayesian optimization, an optimization approach based on Bayesian modeling. Without loss of generality, we demonstrate on a bipedal walker how BO can be used to efficiently optimize the parameters of a controller. We extensively compare BO in different configurations, including various acquisition functions. This chapter is based on [Calandra et al., 2015d], and a preliminary version of some of the work in this chapter was shown in [Calandra et al., 2014a,c]. The idea of evaluating BO on a real robot was provided by Marc Deisenroth, as well as using LQR as a optimization benchmark. Andre Seyfarth provided the robot *Fox* and very insightful discussion about bipedal locomotion. I implemented the necessary software for performing the optimization and the controlling the robot. I also performed the experimental evaluation on the robot. Nakul Gopalan provided the simulator for DMP optimization. Jan Peters provided the initial motivation for this work.

In Chapter 5, we extend single-objective Bayesian optimization to the case of multi-objective optimization, where multiple conflicting objective are optimized jointly. We demonstrate that the use of Bayesian modeling in multi-objective optimization improves the quality of the final solutions and reduces the number of evaluations required. Moreover, we develop a framework to measure robustness to stochastic noise and parameters perturbation and include them as additional objectives. The original motivation was proposed by myself. The theory was joint work between myself and Marc Deisenroth. Felix Unverzagt performed some preliminary evaluation regarding heteroscedastic models. I was responsible for the final experimental evaluation. Jan Peters provided useful advice.

In Chapter 6, we conclude with a summary of the contributions of this thesis. Additionally, we discuss open challenges and future research directions.

2 Bayesian Modeling

> All models are wrong, but some are useful
>
> George E. P. Box

A critical issue throughout this thesis is the capability of creating and training flexible Bayesian models. In Section 2.1, we briefly discuss the classical Gaussian Process (GP) model. In Section 2.2, we introduce the MGP, a novel model that extend GP by jointly learning a feature representation and regression model.

2.1 Gaussian Processes

GPs are a state-of-the-art probabilistic non-parametric regression method [Rasmussen and Williams, 2006]. A GP is a distribution over functions

$$f \sim \mathcal{GP}\left(m, k\right) \tag{2.1}$$

and fully defined by a mean function m (in our case $m \equiv \mathbf{0}$) and a covariance function k. The GP predictive distribution $f\left(\boldsymbol{x}_*\right)$ at a test input $\boldsymbol{x}_*$ is given by

$$\begin{aligned} p\left(f\left(\boldsymbol{x}_*\right)|\mathbb{D}, \boldsymbol{x}_*\right) &= \mathcal{N}\left(\mu(\boldsymbol{x}_*), \sigma^2(\boldsymbol{x}_*)\right), & (2.2)\\ \mu(\boldsymbol{x}_*) &= \boldsymbol{k}_*^T(\boldsymbol{K} + \sigma_w^2\boldsymbol{I})^{-1}\boldsymbol{Y}, & (2.3)\\ \sigma^2(\boldsymbol{x}_*) &= k_{**} - \boldsymbol{k}_*^T(\boldsymbol{K} + \sigma_w^2\boldsymbol{I})^{-1}\boldsymbol{k}_*, & (2.4) \end{aligned}$$

where $\mathbb{D} = \{\boldsymbol{X}, \boldsymbol{Y}\}$ is the training data, $\boldsymbol{K}$ is the kernel matrix with $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $k_{**} = k(\boldsymbol{x}_*, \boldsymbol{x}_*)$, $\boldsymbol{k}_* = k(\boldsymbol{X}, \boldsymbol{x}_*)$ and σ_w^2 is the measurement noise variance. In our experiments, we use different covariance functions k. Specifically, we use the squared exponential covariance function with Automatic Relevance Determination (ARD)

$$k_{\text{SE}}(\boldsymbol{x}_p, \boldsymbol{x}_q) = \sigma_f^2 \exp\left(-\tfrac{1}{2}(\boldsymbol{x}_p - \boldsymbol{x}_q)^T \boldsymbol{\Lambda}^{-1}(\boldsymbol{x}_p - \boldsymbol{x}_q)\right), \tag{2.5}$$

with $\boldsymbol{\Lambda} = \text{diag}([l_1^2, ..., l_D^2])$, where l_i are the characteristic length-scales, and σ_f^2 is the variance of the latent function f. Furthermore, we use the neural network covariance function

$$k_{\text{NN}}(\boldsymbol{x}_p, \boldsymbol{x}_q) = \sigma_f^2 \sin^{-1}\left(\frac{\boldsymbol{x}_p^T \boldsymbol{P} \boldsymbol{x}_q}{\sqrt{(1+\boldsymbol{x}_p^T \boldsymbol{P} \boldsymbol{x}_p)(1+\boldsymbol{x}_q^T \boldsymbol{P} \boldsymbol{x}_q)}}\right), \tag{2.6}$$

where $\boldsymbol{P}$ is a weight matrix. Each covariance function possesses various hyperparameters $\boldsymbol{\omega}$ to be selected. This selection is performed by minimizing the Negative Log Marginal Likelihood (NLML)

$$\begin{aligned} \text{NLML}(\boldsymbol{\omega}) &= -\log p(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{\omega}) & (2.7)\\ &\doteq \tfrac{1}{2}\boldsymbol{Y}^T(\boldsymbol{K}_{\boldsymbol{\omega}} + \sigma_w^2\boldsymbol{I})^{-1}\boldsymbol{Y} + \tfrac{1}{2}\log|\boldsymbol{K}_{\boldsymbol{\omega}} + \sigma_w^2\boldsymbol{I}| - \tfrac{n}{2}\log 2\pi \end{aligned}$$

Using the chain-rule, the corresponding gradient can be computed analytically as

$$\frac{\partial \text{NLML}(\boldsymbol{\omega})}{\partial \boldsymbol{\omega}} = \color{red}{\frac{\partial \text{NLML}(\boldsymbol{\omega})}{\partial \boldsymbol{K}_{\boldsymbol{\omega}}}} \color{blue}{\frac{\partial \boldsymbol{K}_{\boldsymbol{\omega}}}{\partial \boldsymbol{\omega}}}, \tag{2.8}$$

which allows us to optimize the hyperparameters using Quasi-Newton optimization, e.g., L-BFGS [Liu and Nocedal, 1989].

2.2 Manifold Gaussian Processes

Gaussian Processes (GPs) are a powerful state-of-the-art nonparametric Bayesian regression method. The covariance function of a GP implicitly encodes high-level assumptions about the underlying function to be modeled, e.g., smoothness or periodicity. Hence, the choice of a suitable covariance function for a specific data set is crucial. A standard choice is the squared exponential (Gaussian) covariance function, which implies assumptions, such as smoothness and stationarity. Although the squared exponential can be applied to a great range of problems, generic covariance functions may also be inadequate to model a variety of functions where the common smoothness assumptions are violated, such as ground contacts in robot locomotion.

Two common approaches can overcome the limitations of standard covariance functions. The first approach combines multiple standard covariance functions to form a new covariance function [Duvenaud et al., 2013, Rasmussen and Williams, 2006, Wilson and Adams, 2013]. This approach allows to automatically design relatively complex covariance functions. However, the resulting covariance function is still limited by the properties of the combined covariance functions. The second approach is based on data transformation (or pre-processing), after which the data can be modeled with standard covariance functions. One way to implement this second approach is to transform the output space as in the Warped GP [Snelson et al., 2004]. An alternative is to transform the input space. Transforming the input space and subsequently applying GP regression with a standard covariance function is equivalent to GP regression with a new covariance function that explicitly depends on the transformation [MacKay, 1998]. One example is the stationary periodic covariance function [HajiGhassemi and Deisenroth, 2014, MacKay, 1998], which effectively is the squared exponential covariance function applied to a complex representation of the input variables. Common transformations of the inputs include data normalization and dimensionality reduction, e.g., PCA [Pearson, 1901]. Generally, these input transformations are good heuristics or optimize an unsupervised objective. However, they may be suboptimal for the overall regression task.

In this paper, we propose the *Manifold Gaussian Process* (MGP), which is based on MacKay's ideas to devise flexible covariance functions for GPs. Our GP model is equivalent to jointly learning a data transformation into a feature space followed by a GP regression with off-the-shelf covariance functions from feature space to observed space. The model profits from standard GP properties, such as a straightforward incorporation of a prior mean function and a faithful representation of model uncertainty.

Multiple related approaches in the literature attempt joint supervised learning of features and regression/classification. In Salakhutdinov and Hinton [2007], pre-training of the input transformation makes use of computationally expensive unsupervised learning that requires thousands of data points. **?** combined both unsupervised and supervised objectives for the optimization of an input transformation in a classification task. Unlike these approaches, the MGP is motivated by the need of a stronger (i.e., supervised) guidance to discover suitable transformations for regression problems, while remaining within a Bayesian framework. Damianou and Lawrence [2013] proposed the Deep GP, which stacks multiple layers of GP-LVMs, similarly to a neural network. This model exhibits great flexibility in supervised and unsupervised settings, but the resulting model is not a full GP. Snelson and Ghahramani [2006] proposed a supervised dimensionality reduction by jointly learning a liner transformation of the input and a GP. Snoek et al. [2014] transformed the input data using a Beta distribution whose parameters were learned jointly with the GP. However, the purpose of this transformation is to account for skewness in the data, while MGP allows for a more general class of transformations.

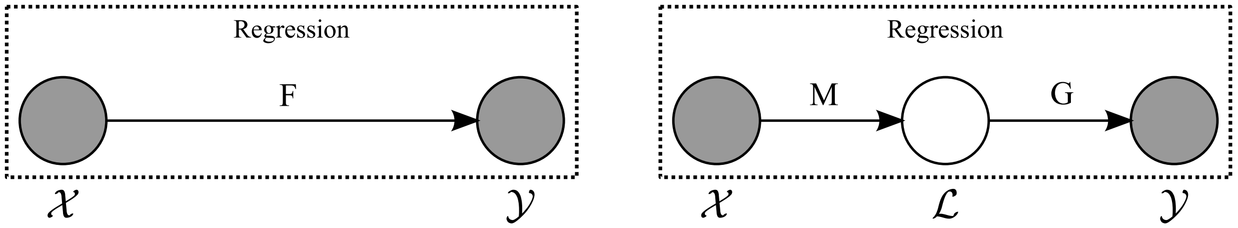
2.2.1 Jointly Learning Feature Space and Gaussian Processes

In the following, we review methods for regression, which may use latent or feature spaces. Following, we introduce the Manifold Gaussian Processes, our novel approach to jointly learning a regression model and a suitable feature representation of the data.

Regression with Learned Features

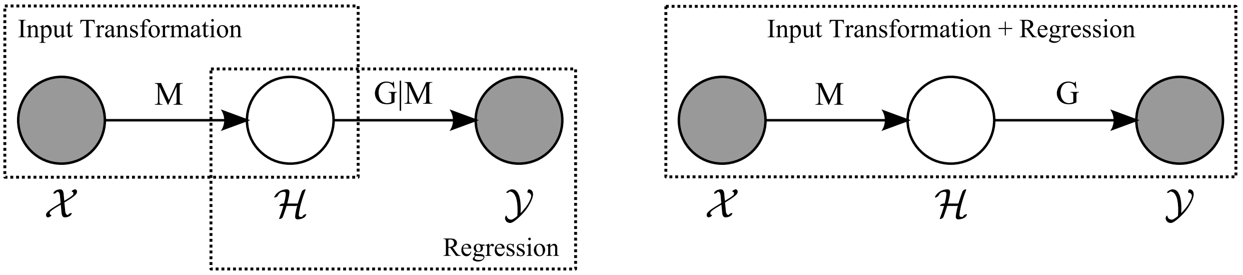
We assume N training inputs $\boldsymbol{x}_n \in \mathscr{X} \subseteq \mathbb{R}^D$ and respective outputs $y_n \in \mathscr{Y} \subseteq \mathbb{R}$, where $y_n = f(\boldsymbol{x}_n) + w$, $w \sim \mathcal{N}\left(0, \sigma_w^2\right)$, $n = 1, \ldots, N$. The training data is denoted by $\boldsymbol{X}$ and $\boldsymbol{Y}$ for the inputs and targets, respectively. We consider the task of learning a regression function $f : \mathscr{X} \to \mathscr{Y}$. The corresponding setting is given in Figure 2.1a. Discovering the regression function f is often challenging for nonlinear functions. A typical way to simplify and distribute the complexity of the regression problem is to introduce an auxiliary latent space $\mathscr{L}$. The function f can then be decomposed into $f = G \circ M$, where $M : \mathscr{X} \to \mathscr{L}$ and $G : \mathscr{L} \to \mathscr{Y}$, as shown in Figure 2.1b. In a full Bayesian framework, the latent space $\mathscr{L}$ is integrated out to solve the regression task f, which is often analytically unfeasible [Schmidt and O'Hagan, 2003].

A common approximation to the full Bayesian framework is to introduce a deterministic feature space $\mathscr{H}$, and to find the mappings M and G in two consecutive steps. First, M is determined by means of unsupervised feature learning.



(a) Supervised learning of a regression function.

(b) Supervised learning integrating out a latent space is often *intractable*.



(c) Unsupervisedly learned input transformation M followed by a conditional regression $G|M$.

(d) Manifold GP: joint supervised learning of the input transformation M and the regression task G.

Figure 2.1.: Different regression settings to learn the function $f : \mathscr{X} \to \mathscr{Y}$. (a) Standard supervised regression. (b) Regression with an auxiliary latent space $\mathscr{L}$ that allows to simplify the task. In a full Bayesian framework, $\mathscr{L}$ would be integrated out, which is often analytically intractable. (c) Decomposition of the overall regression task f into discovering a feature space $\mathscr{H}$ using the map M and a subsequent (conditional) regression $G|M$. (d) Our MGP learns the mappings G and M *jointly*.

Second, the regression G is learned supervisedly as a conditional model $G|M$, see Figure 2.1c. The use of this feature space can reduce the complexity of the learning problem. For example, for complicated non-linear functions a higher-dimensional (overcomplete) representation $\mathscr{H}$ allows learning a simpler mapping $G : \mathscr{H} \to \mathscr{Y}$. For high-dimensional inputs, the data often lies on a lower-dimensional manifold $\mathscr{H}$, e.g., due to non-discriminant or strongly correlated covariates. The lower-dimensional feature space $\mathscr{H}$ reduces the effect of the curse of dimensionality. In this paper, we focus on modeling complex functions with a relatively low-dimensional input space, which, nonetheless, cannot be well modeled by off-the-shelf GP covariance functions.

Typically, unsupervised feature learning methods determine the mapping M by optimizing an unsupervised objective, independent from the objective of the overall regression f. Examples of such unsupervised objectives are the minimization of the input reconstruction error (auto-encoders [Vincent et al., 2008]), maximization of the variance (PCA [Pearson, 1901]), maximization of the statistical independence (ICA [Hyvärinen and Oja, 2000]), or the preservation of the distances between data (isomap [Tenenbaum et al., 2000] or LLE [Roweis and Saul, 2000]). In the context of regression, an unsupervised approach for feature learning can be insufficient as the learned data representation $\mathscr{H}$ might not suit the overall regression task f [Wahlström et al., 2015]: Unsupervised and supervised learning optimize different objectives, which do not necessarily match, e.g., minimizing the reconstruction error as unsupervised objective and maximizing the marginal likelihood as supervised objective. An approach where feature learning is performed in a supervised manner can instead guide learning the feature mapping M toward representations that are useful for the overall regression $f = G \circ M$. This intuition is the key insight of our Manifold Gaussian Processes, where the feature mapping M and the GP G are learned jointly using the same supervised objective as depicted in Figure 2.1d.

Manifold Gaussian Processes

In this section, we describe the MGP model and its parameters $\boldsymbol{\omega}_{\text{MGP}}$ itself, and relate it to standard GP regression. Furthermore, we detail training and prediction with the MGP.

Model

As shown in Figure 2.1d, the MGP considers the overall regression as a composition of functions

$$f = G \circ M \,. \tag{2.9}$$

The two functions M and G are learned jointly to accomplish the overall regression objective function, i.e., the marginal likelihood in Equation (2.7). In this paper, we assume that M is a deterministic, parametrized function that maps the input space $\mathscr{X}$ into the feature space $\mathscr{H} \subseteq \mathbb{R}^Q$, which serves as the domain for the GP regression $G : \mathscr{H} \to \mathscr{Y}$. Performing this transformation of the input data corresponds to training a GP G having $\boldsymbol{H} = M(\boldsymbol{X})$ as input. Therefore, the MGP is equivalent to a GP for a function $f : \mathscr{X} \to \mathscr{Y}^1$ with a covariance function $\tilde{k}$ defined as

$$\tilde{k}(\boldsymbol{x}_p, \boldsymbol{x}_q) = k\left(M(\boldsymbol{x}_p), M(\boldsymbol{x}_q)\right), \tag{2.10}$$

i.e., the kernel operates on the Q-dimensional feature space $\mathscr{H} = M(\mathscr{X})$. According to MacKay [1998], a function defined as in Equation (2.10) is a valid covariance function and, therefore, the MGP is a valid GP.

The predictive distribution for the MGP at a test input $\boldsymbol{x}_*$ can then be derived from the predictive distribution of a standard GP in Equation (5.6) as

$$\begin{aligned} p\left(f\left(\boldsymbol{x}_*\right)|\mathbb{D}, \boldsymbol{x}_*\right) &= p\left((G \circ M)(\boldsymbol{x}_*)|\mathbb{D}, \boldsymbol{x}_*\right) \\ &= \mathscr{N}\left(\mu(M(\boldsymbol{x}_*)), \sigma^2(M(\boldsymbol{x}_*))\right), \qquad (2.11) \\ \mu(M(\boldsymbol{x}_*)) &= \tilde{\boldsymbol{k}}_*^T(\tilde{\boldsymbol{K}} + \sigma_w^2\boldsymbol{I})^{-1}\boldsymbol{Y}, \qquad (2.12) \\ \sigma^2(M(\boldsymbol{x}_*)) &= \tilde{k}_{**} - \tilde{\boldsymbol{k}}_*^T(\tilde{\boldsymbol{K}} + \sigma_w^2\boldsymbol{I})^{-1}\tilde{\boldsymbol{k}}_*, \qquad (2.13) \end{aligned}$$

where $\tilde{\boldsymbol{K}}$ is the kernel matrix constructed as $\tilde{K}_{ij} = \tilde{k}(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $\tilde{k}_{**} = \tilde{k}(\boldsymbol{x}_*, \boldsymbol{x}_*)$, $\tilde{\boldsymbol{k}}_* = \tilde{k}(\boldsymbol{X}, \boldsymbol{x}_*)$, and $\tilde{k}$ is the covariance function from Equation (2.10). In our experiments, we used the squared exponential covariance function from Equation (2.5) for the kernel k in Equation (2.10).

Training

We train the MGP by jointly optimizing the parameters $\boldsymbol{\omega}_M$ of the transformation M and the GP hyperparameters $\boldsymbol{\omega}_G$. For learning the parameters $\boldsymbol{\omega}_{\mathrm{MGP}} = [\boldsymbol{\omega}_M, \boldsymbol{\omega}_G]$, we minimize the NLML as in the standard GP regression. Considering the composition of the mapping $f = G \circ M$, the NLML becomes

$$\begin{aligned} \mathrm{NLML}(\boldsymbol{\omega}_{\mathrm{MGP}}) &= -\log p\left(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{\omega}_{\mathrm{MGP}}\right) \\ &\doteq \tfrac{1}{2}\boldsymbol{Y}^T(\tilde{\boldsymbol{K}}_{\boldsymbol{\omega}_{\mathrm{MGP}}} + \sigma_w^2\boldsymbol{I})^{-1}\boldsymbol{Y} + \tfrac{1}{2}\log|\tilde{\boldsymbol{K}}_{\boldsymbol{\omega}_{\mathrm{MGP}}} + \sigma_w^2\boldsymbol{I}|. \end{aligned}$$

Note that $\tilde{\boldsymbol{K}}_{\boldsymbol{\omega}_{\mathrm{MGP}}}$ depends on both $\boldsymbol{\omega}_G$ and $\boldsymbol{\omega}_M$, unlike $\boldsymbol{K}_{\boldsymbol{\omega}}$ from Equation (2.7), which depends only on $\boldsymbol{\omega}_G$. The analytic gradients $\partial \mathrm{NLML}/\partial \boldsymbol{\omega}_G$ of the objective in Equation (2.14) with respect to the parameters $\boldsymbol{\omega}_G$ are computed as in the standard GP, i.e.,

$$\frac{\partial \mathrm{NLML}(\boldsymbol{\omega}_{\mathrm{MGP}})}{\partial \boldsymbol{\omega}_G} = \frac{\partial \mathrm{NLML}(\boldsymbol{\omega}_{\mathrm{MGP}})}{\partial \boldsymbol{K}_{\boldsymbol{\omega}_{\mathrm{MGP}}}} \frac{\partial \boldsymbol{K}_{\boldsymbol{\omega}_{\mathrm{MGP}}}}{\partial \boldsymbol{\omega}_G}. \tag{2.14}$$

The gradients of the parameters $\boldsymbol{\omega}_M$ of the feature mapping are computed by applying the chain-rule

$$\frac{\partial \mathrm{NLML}(\boldsymbol{\omega}_{\mathrm{MGP}})}{\partial \boldsymbol{\omega}_M} = \frac{\partial \mathrm{NLML}(\boldsymbol{\omega}_{\mathrm{MGP}})}{\partial \boldsymbol{K}_{\boldsymbol{\omega}_{\mathrm{MGP}}}} \frac{\partial \boldsymbol{K}_{\boldsymbol{\omega}_{\mathrm{MGP}}}}{\partial \boldsymbol{H}} \frac{\partial \boldsymbol{H}}{\partial \boldsymbol{\omega}_M}, \tag{2.15}$$

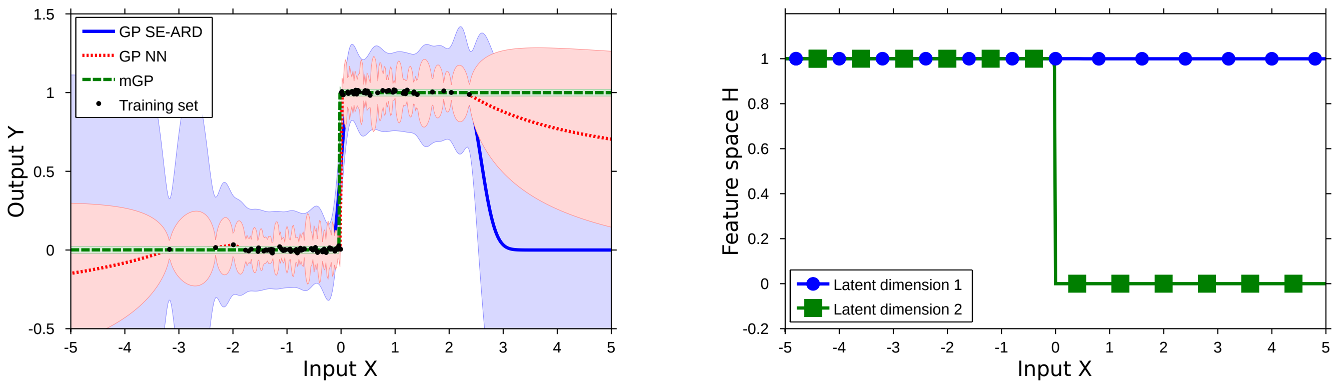
where only $\partial \boldsymbol{H}/\partial \boldsymbol{\omega}_M$ depends on the chosen input transformation M, while $\partial \boldsymbol{K}_{\boldsymbol{\omega}_{\mathrm{MGP}}}/\partial \boldsymbol{H}$ is the gradient of the kernel matrix with respect to the Q-dimensional GP training inputs $\boldsymbol{H} = M(\boldsymbol{X})$. Similarly to standard GP, the parameters $\boldsymbol{\omega}_{\mathrm{MGP}}$ in the MGP can be obtained using off-the-shelf optimization methods.

Input Transformation

Our approach can use any deterministic parametric data transformation M. We focus on multi-layer neural networks and define their structure as $[q_1 - \ldots - q_l]$ where l is the number of layers, and q_i is the number of neurons of the i^{th} layer. Each layer $i = 1, \ldots, l$ of the neural network performs the transformation

$$\mathbb{T}_i(\boldsymbol{Z}) = \sigma\left(\boldsymbol{W}_i\boldsymbol{Z} + \boldsymbol{B}_i\right), \tag{2.16}$$

where $\boldsymbol{Z}$ is the input of the layer, σ is the transfer function, and $\boldsymbol{W}_i$ and $\boldsymbol{B}_i$ are the weights and the bias of the layer, respectively. Therefore, the input transformation M of Equation (2.9) is $M(\boldsymbol{X}) = (\mathbb{T}_l \circ \ldots \circ \mathbb{T}_1)(\boldsymbol{X})$. The parameters $\boldsymbol{\omega}_M$ of the neural network M are the weights and biases of the whole network, so that $\boldsymbol{\omega}_M = [\boldsymbol{W}_1, \boldsymbol{B}_1, \ldots, \boldsymbol{W}_l, \boldsymbol{B}_l]$. The gradients $\partial \boldsymbol{H}/\partial \boldsymbol{\omega}_M$ in Equation (2.15) are computed by repeated application of the chain-rule (backpropagation).



(a) GP prediction.

(b) Learned mapping M using MGP (log-sigmoid).

Figure 2.2.: Step Function: (a) Predictive mean and 95% confidence bounds for a GP with SE-ARD covariance function (blue solid), a GP with NN covariance function (red dotted) and a log-sigmoid MGP (green dashed) on the step function of Equation (2.18). The discontinuity is captured better by an MGP than by a regular GP with either SE-ARD or NN covariance functions. (b) The 2D feature space $\mathscr{H}$ discovered by the non-linear mapping M as a function of the input $\mathscr{X}$. The discontinuity of the modeled function is already captured by the non-linear mapping M. Hence, the mapping from feature space $\mathscr{H}$ to the output $\mathscr{Y}$ is smooth and can be easily managed by the GP.

2.2.2 Experimental Results

To demonstrate the efficiency of our proposed approach, we apply the MGP to challenging benchmark problems and a real-world regression task. First, we demonstrate that MGPs can be successfully applied to learning discontinuous functions, a daunting undertaking with an off-the-shelf covariance function, due to its underlying smoothness assumptions. Second, we evaluate MGPs on a function with multiple natural length-scales. Third, we assess MGPs on real data from a walking bipedal robot. The locomotion data set is highly challenging due to ground contacts, which cause the regression function to violate standard smoothness assumptions.

To evaluate the goodness of the different models on the training set, we consider the NLML previously introduced in Equation (2.7) and Equation (2.14). Additionally, for the test set, we make use of the Negative Log Predictive Probability (NLPP)

$$\mathrm{NLPP} = -\log p(\boldsymbol{y} = \boldsymbol{y}_* | \boldsymbol{X}, \boldsymbol{x}_*, \boldsymbol{Y}, \boldsymbol{\omega}) , \tag{2.17}$$

where the $\boldsymbol{y}_*$ is the test target for the input $\boldsymbol{x}_*$ as computed for the standard GP in Equation (5.6) and Equation (2.11) for the MGP model.

We compare our MGP approach with GPs using the SE-ARD and NN covariance functions, which implement the model in Figure 2.1a. Moreover, we evaluate two unsupervised feature extraction methods, Random Embeddings and PCA, followed by a GP SE-ARD, which implements the model in Figure 2.1c.[2] For the model in Figure 2.1d, we consider two variants of MGP with the log-sigmoid $\sigma(x) = 1/(1+e^{-x})$ and the identity $\sigma(x) = x$ transfer functions. These two transfer functions lead to a non-linear and a linear transformation M, respectively.

Step Function

In the following, we consider the step function

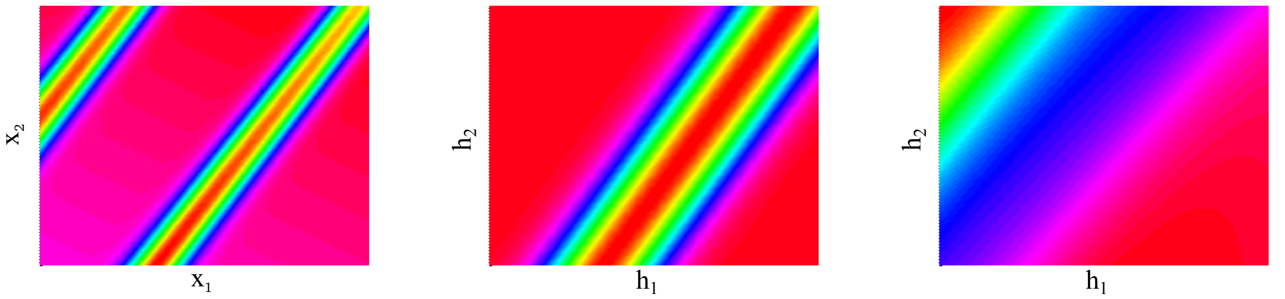
$$\begin{aligned} y &= f(x) + w, \quad w \sim \mathcal{N}\left(0, 0.01^2\right), \\ f(x) &= \begin{cases} 0 & \text{if } x \leq 0 \\ 1 & \text{if } x > 0 \end{cases} . \end{aligned} \tag{2.18}$$

For training, 100 inputs points are sampled from $\mathcal{N}(0,1)$ while the test set is composed of 500 data points uniformly distributed between -5 and $+5$. The MGP uses a multi-layer neural network of [1-6-2] neurons (such that the feature

---

[1] As long as the feature space and kernel are Euclidean.

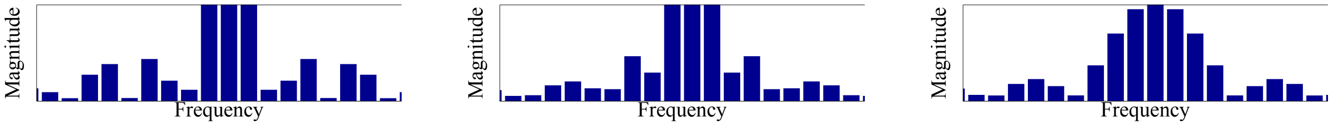
[2] The random embedding is computed as the transformation $\boldsymbol{H} = \boldsymbol{\alpha}\boldsymbol{X}$, where the elements of $\boldsymbol{\alpha}$ are randomly sampled from a normal distribution.



(a) Intensity map of the function.

(b) Intensity map of the learned feature space for the MGP (identity).

(c) Intensity map of the learned feature space for the MGP (log-sigmoid).



(d) Spectrum of the function.

(e) Spectrum of the learned feature space for the MGP (identity).

(f) Spectrum of the learned feature space for the MGP (log-sigmoid).

Figure 2.3.: Multiple Length-Scales: Intensity map of (a) the considered function, (b) the learned feature space of the MGP with a linear activation function and (c) with a log-sigmoid activation. (d)–(f) The corresponding Spectrum for (d) the original function and the learned feature space for (e) MGP (identity) and (f) MGP (log-sigmoid). The spectral analysis of the original function shows the presence of multiple frequencies. The transformations learned by both variants of MGP focus the spectrum of the feature space towards a more compact frequencies support.

space $\mathscr{H} \subseteq \mathbb{R}^2$) for the mapping M and a standard SE-ARD covariance function for the GP regression G. Values of the NLML per data point for the training and NLPP per data point for the test set are reported in Table 2.1. In both performance measures, the MGP using a non-linear transformation outperforms the other models. An example of the resulting predictive mean and the 95% confidence bounds for three models is shown in Figure 2.2a. Due to the implicit assumptions employed by the SE-ARD and NN covariance functions on the mapping f, neither of them appropriately captures the discontinuous nature of the underlying function or its correct noise level. The GP model applied to the random embedding and MGP (identity) perform similar to a standard GP with SE-ARD covariance function as their linear transformations do not substantially change the function. Compared to these models, the MGP (log-sigmoid) captures the discontinuities of the function better, thanks to its non-linear transformation, while the uncertainty remains small over the whole function's domain.

Note that the MGP still assumes smoothness in the regression G, which requires the transformation M to take care of the discontinuity. This effect can be observed in Figure 2.2b, where an example of the 2D learned feature space $\mathscr{H}$ is shown. The discontinuity is already encoded in the feature space. Hence, it is easier for the GP to learn the mapping G. Learning the discontinuity in the feature space is a direct result from jointly training M and G as feature learning is embedded in the overall regression f.

Multiple Length-Scales

In the following, we demonstrate that the MGP can be used to model functions that possess multiple intrinsic length-scales. For this purpose, we rotate the function

$$y = 1 - \mathscr{N}\left(x_2|3, 0.5^2\right) - \mathscr{N}\left(x_2|-3, 0.5^2\right) + \tfrac{x_1}{100} \tag{2.19}$$

anti-clockwise by 45°. The intensity map of the resulting function is shown in Figure 2.3a. By itself (i.e., without rotating the function), Equation (2.19) is a fairly simple function. However, when rotated, the correlation between the covariates substantially complicates modeling. If we consider a horizontal slice of the rotated function, we can see how different spectral frequencies are present in the function, see Figure 2.3d. The presence of different frequencies is problematic for covariance functions, such as the SE-ARD, which assume a single frequency. When learning the hyperparameters, the length-scale needs to trade off different frequencies. Typically, the hyperparameter optimization gives a preference to shorter length-scales. However, such a trade-off greatly reduces the generalization capabilities of the model.

We compare the performances of a standard GP using SE-ARD and NN covariance functions and random embeddings followed by a GP using the SE-ARD covariance function, and our proposed MGP. We train these models with 400 data

Method	Training set		Test set	
	NLML	RMSE	NLPP	RMSE
GP SE-ARD	-0.68	1.00×10^{-2}	$+0.50 \times 10^{-3}$	0.58
GP NN	-1.49	$\mathbf{0.57 \times 10^{-2}}$	$+0.02 \times 10^{-3}$	0.14
MGP (log-sigmoid)	$\mathbf{-2.84}$	1.06×10^{-2}	$\mathbf{-6.34 \times 10^{-3}}$	**0.02**
MGP (identity)	-0.68	1.00×10^{-2}	$+0.50 \times 10^{-3}$	0.58
RandEmb + GP SE-ARD	-0.77	5.26×10^{-2}	$+0.51 \times 10^{-3}$	0.52

Table 2.1.: Step Function: Negative Log Marginal Likelihood (NLML) and Negative Log Predictive Probability (NLPP) per data point for the step function of Equation (2.18). The MGP (log-sigmoid) captures the nature of the underlying function better than a standard GP in both the training and test sets.

Method	Training set		Test set	
	NLML	RMSE	NLPP	RMSE
GP SE-ARD	-2.46	0.40×10^{-3}	-4.34	1.51×10^{-2}
GP NN	-1.57	1.52×10^{-3}	-2.53	6.32×10^{-2}
MGP (log-sigmoid)	$\mathbf{-6.61}$	$\mathbf{0.37 \times 10^{-4}}$	$\mathbf{-7.37}$	$\mathbf{0.58 \times 10^{-4}}$
MGP (identity)	-5.60	0.79×10^{-4}	-6.63	2.36×10^{-3}
RandEmb + GP SE-ARD	-0.47	6.84×10^{-3}	-1.29	1.19×10^{-1}

Table 2.2.: Multiple Length-Scales: NLML per data point for the training set and NLPP per data point for the test set. The MGP captures the nature of the underlying function better than a standard GP in both the training and test sets.

points, randomly sampled from a uniform distribution in the intervals $x_1 = [0, 10]$, $x_2 = [0, 10]$. As a test set we use 2500 data points distributed on a regular grid in the same intervals. For the MGP with both the log-sigmoid and the identity transfer functions, we use a neural network of [2-10-3] neurons. The NLML and the NLPP per data point are shown in Table 2.2. The MGP outperforms all other methods evaluated. We believe that this is due to the mapping M, which transforms the input space so as to have a single natural frequency. Figure 2.3b shows the intensity map of the feature space after the MGP transformed the inputs using a neural network with the identity transfer function. Figure 2.3c shows the intensity map of the feature when the log-sigmoid transfer function is used. Both transformations tend to make the feature space smother compared to the initial input space. This effect is the result of the transformations, which aim to equalize the natural frequencies of the original function in order to capture them more efficiently with a single length-scale. The effects of these transformations are clearly visible in the spectrogram of the MGP (identity) in Figure 2.3e and of the MGP (log-sigmoid) in Figure 2.3f. The smaller support of the spectrum, obtained through the non-linear transformations performed by MGP using the log-sigmoid transfer function, translates into superior prediction performance.

Bipedal Robot Locomotion

Modeling data from real robots can be challenging when the robot has physical interactions with the environment. Especially in bipedal locomotion, we lack good contact force and friction models. Thus, we evaluate our MGP approach on modeling data from the bio-inspired bipedal walker *Fox* [Renjewski, 2012] shown in Figure 2.4. The data set consists of measurements of six covariates recorded at regular intervals of 0.0125 sec. The covariates are the angles of the right and left hip joints, the angles of the right and left knee joints and two binary signals from the ground contact sensors. We consider the regression task where the left knee joint is the prediction target $\mathcal{Y}$ and the remaining five covariates are the inputs $\mathcal{X}$. For training we extract 400 consecutive data points, while we test on the following 500 data points. The MGP uses a network structure [1-30-3].

Table 2.3 shows that the MGP models the data better than the other models. The standard GPs with SE-ARD or NN covariance function predict the knee angle relatively well.

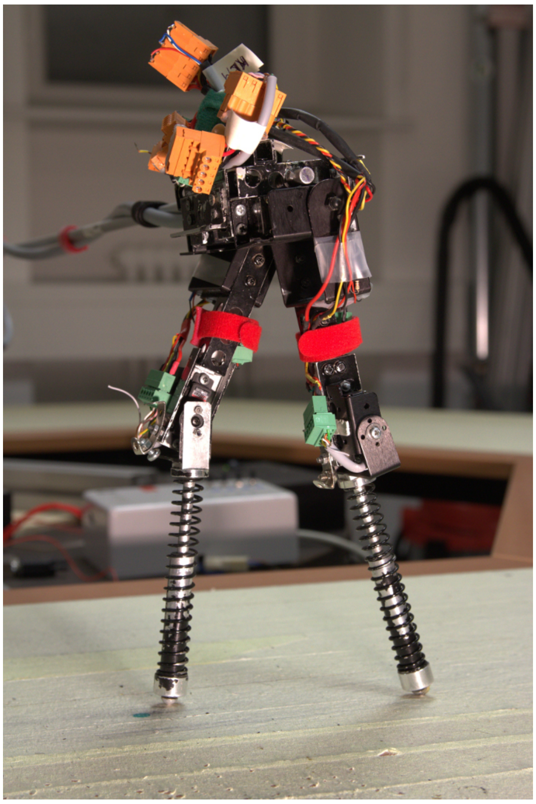


Figure 2.4.: Bipedal Robot Locomotion: The bio-inspired bipedal walker *Fox* from which the dataset is generated.

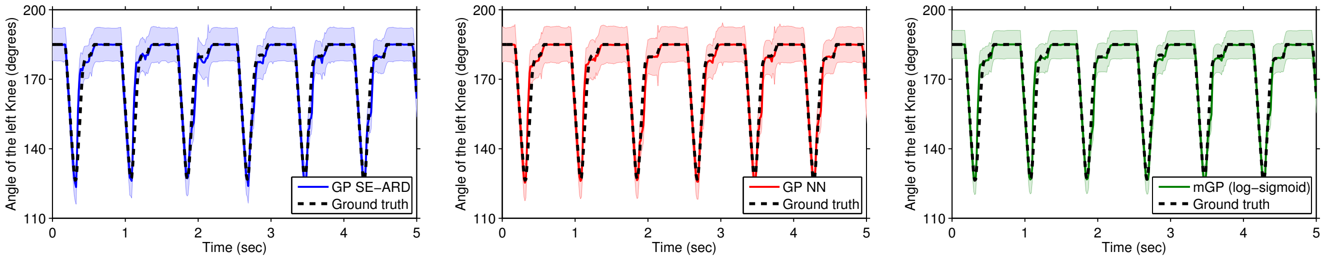
Figure 2.5 shows that the MGP has larger variance of the prediction for areas where fast movement occurs due to leg swinging. However, it captures the structure and regularity of the data better, such as the mechanically enforced upper bound at 185 degrees. The uncertainty of about 20 degrees is reasonable for the fast changes in the knee angle during the swinging phase. However, the same uncertainty of noise is unrealistic once the knee is fully extended at 185 degrees. Therefore, for control purposes, using the MGP model would be preferable. This is a positive sign of the potential of MGP to learn representations that are meaningful for the overall regression task. Figure 2.6 visualizes two dimensions of the learned feature space in which the walking trajectory is smoothly embedded.

2.2.3 Discussion

Unlike neural networks, which have been successfully used to extract complex features, MacKay [1998] argued that GPs are unsuited for feature learning. However, with growing complexity of the regression problem, the discovery of useful data representations (i.e., features) is often necessary. Despite similarities between neural networks and GPs [Neal, 1995], it is still unclear how to exploit the best of both worlds. Inspired by deep neural networks, deep GPs stack multiple layers of GP latent variable models [Damianou and Lawrence, 2013]. This method can be used also in a supervised regression framework, which renders it similar to our proposed MGP. The main differences between Deep GPs and the MGP is that (a) Deep GPs integrate out the latent functions connecting the individual layers and do not require to explicitly define a deterministic transformation structure and, (b) unlike the MGP, the Deep GP is not a full GP. Our MGP model extends a standard GP by learning corresponding useful data representations for the regression problem at hand. In particular, it can discover feature representations that comply with the implicit assumptions of the GP covariance function employed.

One of the main challenges of training MGPs using neural networks as mapping M is the unwieldy joint optimization of the parameters $\boldsymbol{\omega}_{\mathrm{MGP}}$. The difficulty resides in the non-convexity of the NLML, leading to multiple local optima. Depending on the number of parameters $\boldsymbol{\omega}_M$ of the feature map M, the problem of local optima can be severe. This problem is well-known for neural networks, and there may be feasible alternatives to L-BFGS, such as the Hessian-free optimization proposed by Martens [2010]. Additionally, sparsity and low-rank approximations in $\boldsymbol{W}$ and $\boldsymbol{B}$ can be beneficial to reduce the complexity of the optimization.

The extreme expressiveness of the MGP does not prevent the model from solving “easy” regression tasks. For a proof-of-concept, we applied the MGP to modeling a sinusoidal function, which is very easy to model with a standard GP. The results in Table 2.4 suggest that even for simple functions the MGP performs as good as a standard GP.



(a) GP with SE-ARD

(b) GP with NN

(c) MGP (log-sigmoid)

Figure 2.5.: Bipedal Robot Locomotion: Predictive mean and 95% confidence bounds on the test set on real robot walking data for (a) GP with SE-ARD covariance function, (b) GP with NN covariance function and (c) MGP (log-sigmoid). The MGP (log-sigmoid) captures the structure of the data better compared to GPs with either SE-ARD or NN covariance functions.

Method	Training set		Test set	
	NLML	RMSE	NLPP	RMSE
GP SE-ARD	−0.01	0.18	−0.13	0.20
GP NN	0.04	**0.17**	−0.13	0.20
MGP (log-sigmoid)	**−0.28**	**0.17**	**−0.18**	**0.19**
MGP (identity)	0.97	0.03	0.86	0.66
PCA + GP SE-ARD	0.01	0.18	−0.12	0.20
RandEmb + GP SE-ARD	0.16	0.18	−0.09	0.20

Table 2.3.: Bipedal Robot Locomotion: NLML per data point for the training set and NLPP per data point for the test set. The MGP captures the nature of the underlying function well in both the training and test sets.

Increasing the number of parameters of the mapping M intuitively leads to an increased flexibility in the learned covariance function. However, when the number of parameters exceeds the size of data set, the model is prone to over-fitting. For example, during experimental validation of the step function, we noticed the undesirable effect that the MGP could model discontinuities at locations slightly offset from their actual locations. In these cases, training data was sparse around the locations of discontinuity. This observed effect is due to over-fitting of the deterministic transformation M. Ideally, we replace this deterministic mapping with a probabilistic one, which would describe the uncertainty about the location of the discontinuity. In a fully Bayesian framework, we would average over possible models of the discontinuity. However, the use of such a probabilistic mapping in the context of GP regression is analytically intractable in closed form [Schmidt and O'Hagan, 2003] and would require to train GPs with uncertain inputs. This kind of GP training is also analytically intractable, although approximations exist [Lawrence, 2005, McHutchon and Rasmussen, 2011, Titsias and Lawrence, 2010, Wang et al., 2008].

2.3 Summary

The quality of a Gaussian process model strongly depends on an appropriate covariance function. However, designing such a covariance function is challenging for some classes of functions, e.g., highly non-linear functions. To model such complex functions we introduced Manifold Gaussian Processes. The key idea is to decompose the overall regression into learning a feature space mapping and a GP regression that maps from this feature space to the observed space. Both the input transformation and the GP regression are learned jointly and supervisedly by maximizing the marginal likelihood. The MGP is a valid GP for the overall regression task using a more expressive covariance function.

The MGP successfully modeled highly non-liner functions, e.g., step functions or effects of ground contacts in robot locomotion, where standard GPs fail. Applications that profit from the enhanced modeling capabilities of the MGP include robot modeling (e.g., contact and stiction modeling), reinforcement learning, and Bayesian optimization.

We have now detailed the foundations of Bayesian modeling used in the remaining of this thesis. In the next chapters, we will describe how such modeling techniques can be applied and tailored to solve real-world robotics problems.

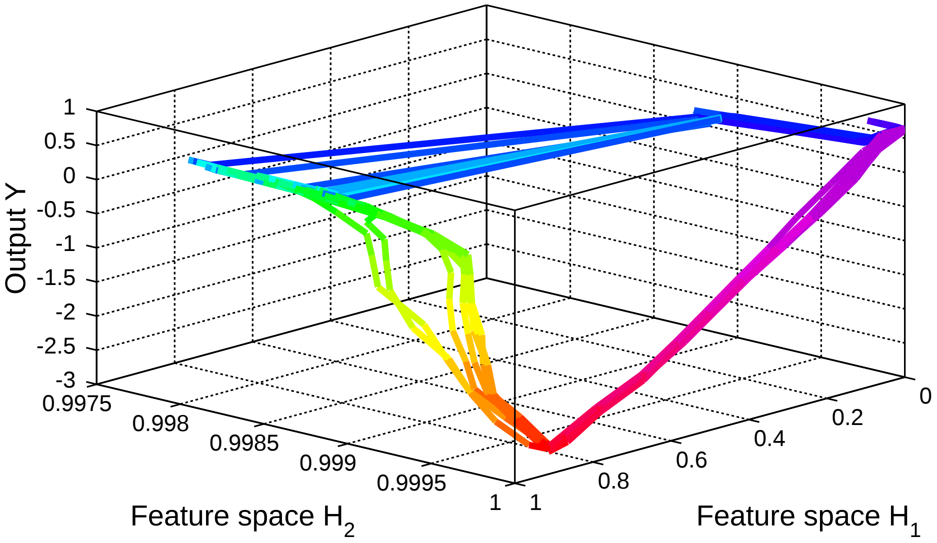


Figure 2.6.: Bipedal Robot Locomotion: The MGP learns a smooth feature space representation (the color indicates the phase during a single step).

Method	Training set		Test set	
	NLML	RMSE	NLPP	RMSE
GP SE-ARD	−4.30	2.78×10^{-3}	−4.42	2.91×10^{-3}
MGP (log-sigmoid)	−4.31	2.76×10^{-3}	−4.42	2.90×10^{-3}

Table 2.4.: Smooth function: NLML per data point for the training set and NLPP per data point for the test set. There is no relevant difference between MGP and standard GP in modeling smooth functions.

3 Learning High-dimensional Robot Inverse Dynamics Models from Artificial Skin

> Time present and time past
> Are both perhaps present in time future,
> And time future contained in time past
>
> T. S. Eliot

Creating analytical dynamics models of complex robots is a challenging and time-consuming process. Our first application of Bayesian modeling is therefore the learning of such dynamics models. In this chapter in particular, we study the learning of inverse dynamics model for the control of a humanoid robot in presence of contacts. Crucial challenge to the use of traditional Bayesian models is the presence of a high dimensional tactile skin that has to be integrated in the model in order to receive information about the location of the contacts.

3.1 Introduction

A key challenge for torque-controlled humanoid robots is to accurately control their dynamics in presence of contacts with the environment Ivaldi et al. [2016], for example during manipulation in clutter [Jain et al., 2013], locomotion [Calandra et al., 2015d] and whole-body movements [Nori et al., 2015]. Accomplishing this feat would allow a larger choice of possible interactions with the environment and even with humans. In such complex situations, the dynamics of the system can be used to compute the required joint torques necessary to realize the desired trajectories of the joints.

Being able to compute a model of the dynamics of the system is at the base of multiple control schema such as inverse dynamics control [Erez and Todorov, 2012, Nguyen-Tuong et al., 2009], computed torque control [Siciliano et al., 2009] and model predictive control [Camacho and Alba, 2013, Maciejowski, 2002]. Estimating the influence of multiple contacts onto the dynamics is a crucial step to use inverse dynamics for control. However, this step is very challenging [Todorov, 2014] as in real scenarios the contact properties may vary significantly and be difficult to consider in the estimation of the whole-body dynamics. At the same time, rigid or compliant contacts should be considered in the design of the robot control [Azad and Mistry, 2015, Kolev and Todorov, 2015].

In principle, there are two ways of estimating the dynamics in presence of multiple contacts: 1) we could try to measure all torques at joint level and detect external torques from the difference between the measured torques and a contact-free inverse dynamics model, and 2) predict the external joint torques jointly with the inverse dynamics. The first approach requires an unrealistically exact analytical model [Nguyen-Tuong and Peters, 2011] and extremely good torque sensing at joint level – not even available on our platform.

Inaccurate joint torques estimation can deteriorate the performances of inverse dynamics control and tracking of the desired trajectories, which can ultimately cause severe problems when controlling the robot motion in presence of uncertainties, disturbances and unexpected contacts with the environment. In presence of contacts, an accurate estimation of the joint torques and the external forces is difficult even with dedicated sensors, such as force/torque sensors, in the

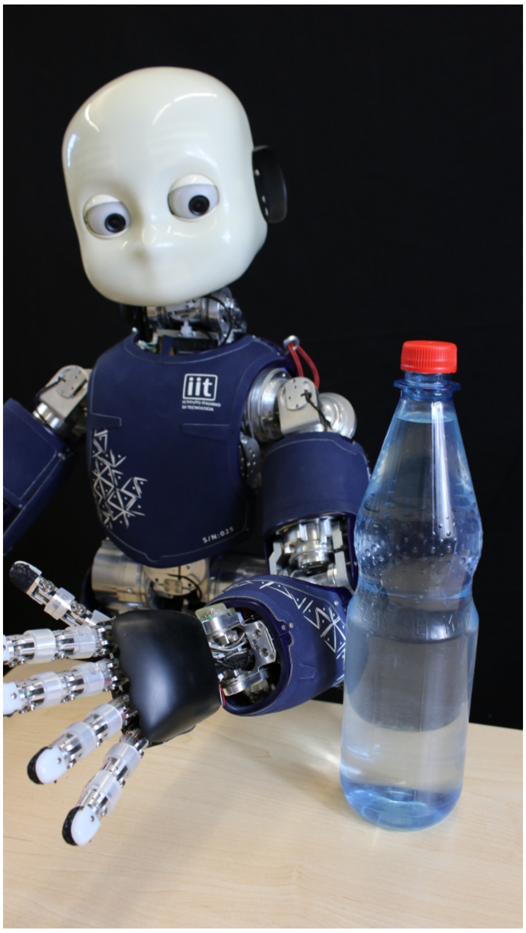


Figure 3.1.: The humanoid robot *iCub* used in our experiments. The blue artificial skin consists of over 2000 tactile sensors distributed across the whole body.

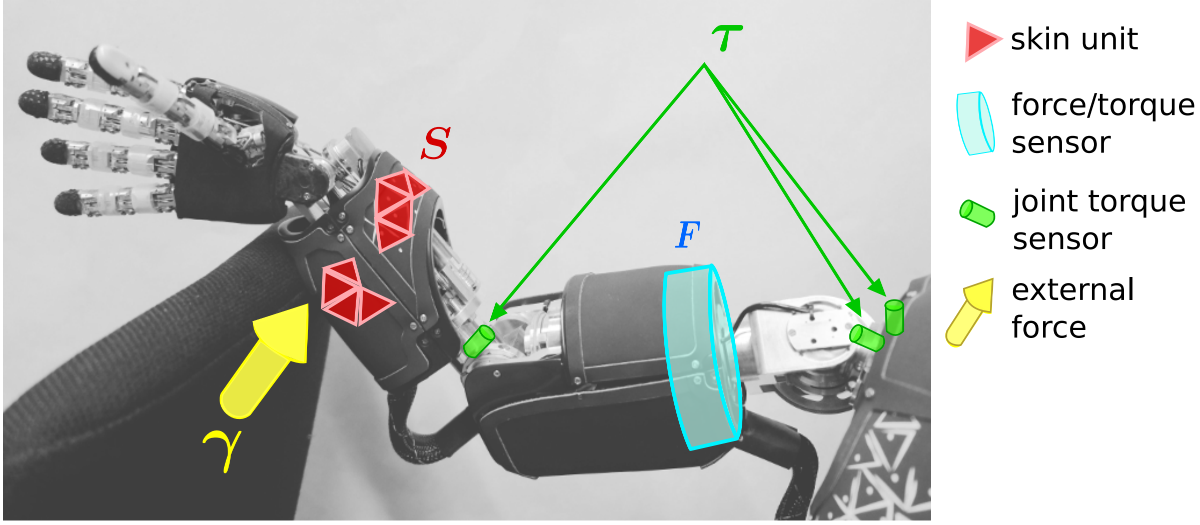


Figure 3.2.: Illustration of the force/torque and tactile sensors during a contact of the robot arm with the environment.

joints or in the links Fumagalli et al. [2012]. Moreover, employing a force/torque sensor in each possible contact location is frequently neither economically cost-effective nor mechanically feasible due to the size and weight of available sensors. As an alternative for contact detection, force/torque sensors can be replaced by cheaper and lighter arrays of tactile sensors. Tactile sensors can be used to compute the contact location and estimate the contact force (the quality of the estimation, in that case, depends on the technology used to fabricate the sensors). For more information about current tactile sensor and robot skin technology see Dahiya et al. [2010, 2013], Yogeswaran et al. [2015].

Recently, some humanoid robots have started being equipped with whole-body distributed tactile sensors, in form of a "skin" [Cannata et al., 2008, Mittendorfer et al., 2015]. The key procedure enabling the use of these sensors for measuring the contact location is the spatial calibration, where the geometric location of each tactile unit (called *taxel*) is identified with respect to the frames of the robot links hosting the sensors. The spatial calibration of the skin is a tedious and lengthy procedure prone to errors, which must be repeated for each individual robot since it depends on the sensor placement; it must be repeated even among different robots of the same type, since there can be small mechanical differences in the exact placement of the tactile sensors on the robotic structure, or electronics differences in the sensors [Del Prete et al., 2011]. Additionally, the calibration of the skin requires the activation of each single sensor and therefore applying this approach to a high number of sensors can be very time consuming. Despite the calibration difficulties, the robotic skin is appealing to solve the contact location estimation problem. However, when using whole-body distributed tactile sensing, small errors in the estimation of the contact location can substantially deteriorate the quality of the torque estimation [Del Prete et al., 2012].

Two main lines of research are currently addressing accurate estimation of the dynamics. On the one hand, there are model-based approaches which rely on identification techniques to improve the dynamics parameters of the analytic models [Ogawa et al., 2014, Traversaro et al., 2015]. On the other hand, we have machine learning approaches, where non-parametric models of the robot dynamics can be learned from data collected on the robot [Jamone et al., 2014, Nguyen-Tuong et al., 2008, **?**]. The main advantages of the latter are that they do not usually require a prior parametric identification, kinematics/dynamics calibration of the robot [Yamane, 2011] and can use raw sensor input without pre-processing. As a result, machine learning approaches are also less prone to model errors, and can easily deal with noise and varying contexts [Fumagalli et al., 2010, Jamone et al., 2014].

The advantages of the machine learning approaches are especially valuable for tasks that rely on computing the dynamics from high-dimensional raw tactile sensors, since there is no need to perform spatial calibration of the tactile sensors [Pugach et al., 2015]. Therefore, in this chapter, we follow the machine learning approach and focus on the task of learning the inverse dynamics model of a humanoid robot in presence of contacts, which are *a priori* unknown (i.e., no information about the position of the environment, or its properties) and that are dynamically created and destroyed.

3.1.1 Contributions

This chapter extends our previous work [Calandra et al., 2015a,b]. In Calandra et al. [2015b], we proposed a data-driven mixture-of-experts learning approach based on Gaussian Processes, which predicts joint torques from joint encoders,

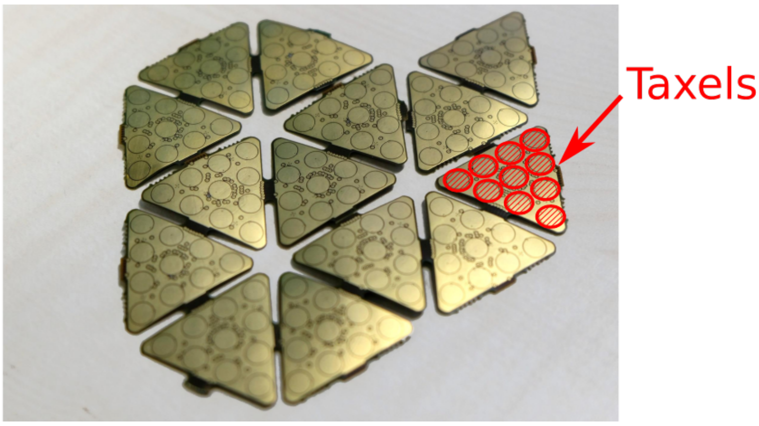


Figure 3.3.: Arrays of tactile sensors used on the *iCub*: each of the triangular array consists of 10 separate *taxels*. Typically, a lengthy spatial calibration is required after the sensors are mounted on the robot, to identify theirs exact position.

tactile and force/torque sensors data[1]. We applied such a model to learn the joint torques of the arm of *iCub* in presence of contacts, showing that the learned model was more accurate than analytic dynamics models using the same sensors data [Ivaldi et al., 2011]. However, the proposed learning approach required the availability of joint torque sensors to supervise the learning process and provide the ground truth. This requirement limited the application of the proposed approach to few robots, in particular excluding almost all *iCubs*[2]. In Calandra et al. [2015a], we generalized the previous approach, relaxing the need for joint torque sensors on the robot. Most importantly, we demonstrated in an active control task with the humanoid robot *iCub* that the proposed model can be efficiently used for computed torque control in presence of contacts with the environment. With the learned model we could decrease the tracking error in presence of known or novel contacts.

In the current chapter, we bring the overall approach to its full maturity and extend [Calandra et al., 2015a,b] by addressing critical issues in learning from high-dimensional inputs, such as the artificial skin, using deep learning techniques. Learning from high-dimensional inputs is crucial for estimating torques from the distributed tactile elements that cover most of the robot's body. In particular, since *iCub* is getting gradually more and more body parts covered with tactile skin, it is critical to be able to estimate the dynamics in presence of multiple contacts captured by the entire skin. We demonstrate here that it is possible to considerably reduce the dimensionality of the skin data preserving the information content of the contact position by using stacked autoencoders. Reducing the dimensionality of the skin helps to increase the classification accuracy when learning the gating network tasked with identifying the contact position. Finally, we thoroughly discuss the current limitations of our approach and propose possible future extensions.

3.1.2 Outline of the Chapter

The remainder of this chapter is structured as follows: In Section 3.2, we formalize inverse dynamics and discuss common model learning and system identification methods, and related literature. In Section 3.3, we define inverse dynamics as a learning task and we present our approach for learning an inverse dynamics that uses tactile sensors as input. In Section 3.4, we present the experimental evaluation of the dimensionality reduction technique and validate our control approach on the humanoid robot *iCub*. In Section 3.5, we discuss the limitations of our approach and of the current technology of tactile sensors, and conclude with final remarks and future applications.

3.2 Background & Related Work

The inverse dynamics (ID) of a robot with m degrees of freedom without contacts can be generally described by

$$\tau = \underbrace{\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}})}_{\tau_{RBD}} + \epsilon(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}}), \tag{3.1}$$

---

[1] In that study the force/torque sensor was placed in the middle of the robotic arm proximally with respect to the end-effector and contact location.

[2] Only one iCub, *iCubParis02* is equipped with joint torque sensors - and not in all the joints.

where $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$ are the joint positions, velocities and accelerations, respectively, $\boldsymbol{M}(\boldsymbol{q})$ is the inertia matrix and

$$\boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}) = C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + g(\boldsymbol{q}) + F_v\dot{\boldsymbol{q}} + F_s\,\mathrm{sgn}(\dot{\boldsymbol{q}})$$

is the matrix combining the contributions from Coriolis and centripetal, friction (viscous and static) and gravity torques. The term $\epsilon(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})$ in Equation (3.1) captures the errors of the analytical model, such as unmodeled dynamics (e.g., elastic elements and Stribeck friction), inaccuracy of the dynamics parameters (e.g., masses, inertia), vibrations, couplings, and sensor noise. With a set $\mathscr{C} = \{c_1 \dots c_n\}$ of contacts c_i occurring between the robot and the environment, Equation (3.1) becomes

$$\boldsymbol{\tau} = \underbrace{\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \epsilon(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})}_{\boldsymbol{\tau}_{ID}} + \underbrace{\sum_{c_i \in \mathscr{C}} \boldsymbol{J}_{c_i}^T(\boldsymbol{q})\boldsymbol{\gamma}_i}_{\boldsymbol{\tau}_{ext}}, \tag{3.2}$$

where the term $\boldsymbol{\tau}_{ext}$ accounts for the additive effect of the external wrenches (forces and moments) $\boldsymbol{\gamma}_i$ applied at the contact location c_i, and $\boldsymbol{J}_{c_i}(\boldsymbol{q})$ is the contact Jacobian. The contact location c_i is not necessarily fixed as the contacts may occur on the whole robotic structure and not exclusively at the end-effectors. Moreover, contacts may be dynamically started and ended as a consequence of the robot's or environment's motion. In such a case, the unknown contact location c_i must be estimated through distributed tactile sensors. To compute the contact Jacobian $\boldsymbol{J}_{c_i}(\boldsymbol{q})$, we need the position of the contact point with respect to the reference frame of the link Fumagalli et al. [2012]. The inverse dynamics control schema employed once the torques estimates are available is detailed in Section 3.3.4.

3.2.1 Contacts and Artificial Skins

The knowledge of the contact location, if retrieved by an artificial skin, requires a kinematic calibration of the skin as explained by Del Prete et al. Del Prete et al. [2011]. If such information is not available, it may still be possible to make (strong) assumptions on the contact location, as done in Ivaldi et al. [2011], for example by assuming that contacts occur at the end-effectors.

As discussed in Cannata et al. [2010], the skin calibration problem is essentially a 6D maximum-likelihood mapping problem, where the position and orientation of each tactile element (*taxel*) with respect to the robot base frame must be estimated. The skin calibration problem has not received much attention in the literature, because of the lack of large-scale tactile sensors. However, in the last years many robots have been equipped with artificial tactile skins: for example, the RoboSkin on *Fox* Cannata et al. [2008] and the CellularSkin on the UR Bergner et al. [2015]. Del Prete et al. Del Prete et al. [2011] performed a first kinematic calibration of the *Fox*'s skin through a long procedure where each taxel had to be manually identified and excited. Roncone et al. Roncone et al. [2014] proposed an automatic calibration method for the arm skin, based on robot self touch. While this procedure is appealing, it requires that the robot can touch each sensor and measure the force applied by the fingertip, and is not immune to imprecision. Generally, not having to rely on such calibration phase for locating the contacts would be highly desirable.

3.2.2 Classical Model-based Approaches for Computing the Robot Dynamics

Classical approaches for computing the torques $\boldsymbol{\tau}$ rely on dynamics models with known or identified kinematics and dynamics parameters Ivaldi et al. [2014]. The torques $\boldsymbol{\tau}_{RBD} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}})$ can be computed analytically through the rigid body dynamics model of the robot (assuming perfect rigid bodies and knowledge of the parameters), resulting in a standard parametric description of the robot Featherstone and Orin [2008]. The term $\epsilon(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})$ is often neglected, or implicitly taken into account by considering a perturbation in the dynamics parameters of $\boldsymbol{\tau}_{RBD}$, which need to be identified accurately.

The difficulty of parameter identification for robot arms is often underestimate Nakanishi et al. [2008]. Nevertheless, for some arms, sufficiently accurate estimates have been obtained Nakanishi et al. [2008], Ting et al. [2006]. The procedure for floating-base robots, such as humanoids, is not straightforward because of two main issues: 1) It is challenging the generation of sufficiently large accelerations for the identification while maintaining the robot balance and the control of contacts. This issue was well explained in [Yamane, 2011], which proposed a technique to identify the mass and the local center of mass of the links in a humanoid robot with fixed feet at the ground and slow joint trajectories. 2) It might not be straightforward to measure the external forces $\boldsymbol{\gamma}_i$ exerted on the robot as it is not possible to cover the robot body with 6-axis force/torque sensors to measure the force exerted on every possible contact location c_i. With few exceptions Mittendorfer and Cheng [2012], such sensors are usually big, heavy and expensive. Thus, they are carefully placed where the external forces are critical for the main tasks, for example at the end-effectors for manipulation and

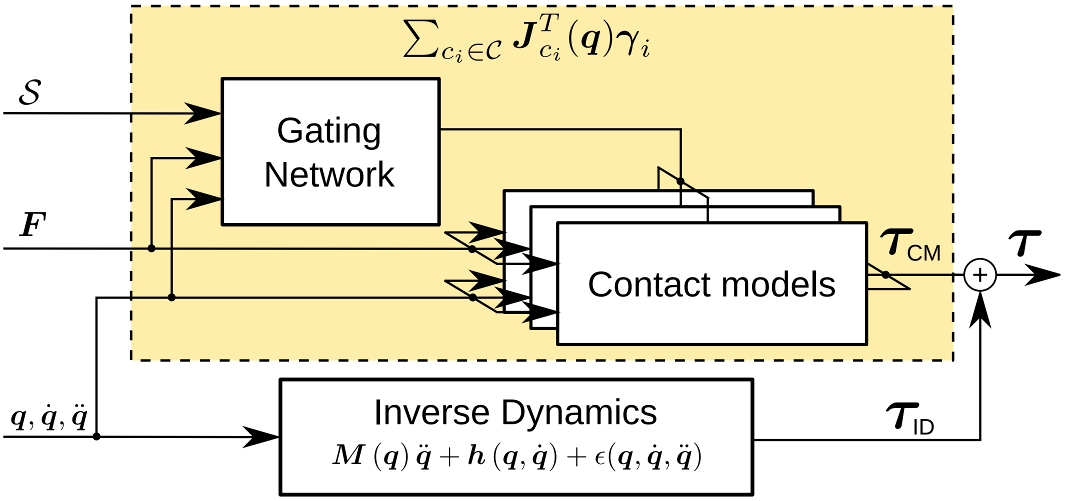

$\sum_{c_i \in \mathcal{C}} \boldsymbol{J}_{c_i}^T(\boldsymbol{q})\gamma_i$
$\mathcal{S}$
Gating Network
$\boldsymbol{F}$
Contact models
$\boldsymbol{\tau}_{\text{CM}}$
+
$\boldsymbol{\tau}$
$\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$
Inverse Dynamics
$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \epsilon(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$
$\boldsymbol{\tau}_{\text{ID}}$

Figure 3.4.: Our approach extends existing inverse dynamics without contacts by learning many contact models which serve as correction terms under different contacts type. The decision of which contact model to activate is taken by a gating network based on the skin measurements $\boldsymbol{s}$, the force torque sensors $\boldsymbol{F}$ and the current state $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$.

at the feet for balancing. In such cases, it has been demonstrated that it is principally possible to identify the dynamics parameters while balancing and walking without additional contacts [Ogawa et al., 2014]. When force/torque sensors are placed proximally, such as in the *Fox* arms [Fumagalli et al., 2012], some of the dynamics parameters can be identified albeit in the absence of contacts [Traversaro et al., 2013].

When multiple contacts are exerted on the robot structure at other locations than the classical end-effectors, it may still be possible to compute an accurate inverse dynamics estimate, but this approach requires both pervasive joint torque sensing (as in *Toro* [Ogawa et al., 2014]) and additional force/torque and tactile sensing (as in the *Fox* Ivaldi et al. [2011]). Moreover, it requires the precise knowledge of the contact locations detected by the tactile sensors, which can only be obtained by a spatial calibration of the skin [Del Prete et al., 2011]. This procedure is prone to errors, and it has been shown that small errors in the kinematics calibration of the taxels (i.e., the tactile units) can induce non-negligible errors in the estimation of the contact forces [Del Prete et al., 2012].

Overall, these model-based approaches have three main limitations: 1) It is hard to identify couplings, elasticity, friction and other unmodeled nonlinear dynamics, which are required for sufficient accuracy. 2) The performance of the data-driven identification strongly depends on the experimental setting (with/without contacts) and the exciting trajectories [Pedrocchi et al., 2014]. 3) They make strong assumptions regarding the contact model and its position.

3.2.3 Learning the Inverse Dynamics

An appealing alternative approach to analytic dynamics computation is to use machine learning methods to learn the dynamics model of a robot Deisenroth et al. [2012], Nguyen-Tuong and Peters [2011], Sigaud et al. [2011], Vijayakumar and Schaal [2000]. Learning the dynamics model does not (necessarily) require a parametric model. Therefore, without the need for compensating for insufficient feature functions, a learned dynamics model can improve the tracking and control performance of a robot Nguyen-Tuong et al. [2009], Schaal et al. [2002]. Overcoming the limitations of traditional model-based approaches is an advantage of learning the inverse dynamics. These include difficulty of modeling complex nonlinear dynamics, restrictive assumptions regarding contacts and sensors, prior accurate kinematics calibration of the tactile sensors. However, previous machine learning techniques used to learn dynamics models often do not scale sufficiently with the increase of the dimensionality of the input, and cannot hence be applied to the problem of learning the dynamics exploiting hundreds of tactile sensors. Additionally, with rare exceptions Calandra et al. [2016], existing machine learning techniques are not designed to model discontinuous functions such as contacts. In the next section, we introduce a novel approach designed to overcome these limitations.

3.3 Mixture-of-Contacts using Gaussian Processes

In this section, we describe our approach for learning ID models with contacts. We first formalize the problem as learning a mixture-of-experts model. Subsequently, we discuss the design and the automatically learning of the gating network.

We then detail how to use Gaussian processes as the corresponding experts. To complete the approach, we discuss the employed control schema.

3.3.1 Learning a Mixture-of-Contacts

When learning the inverse dynamics model (IDM) with contacts of Equation (3.2), we assume that the (contact-free) inverse dynamics $\boldsymbol{\tau}_{ID}$ can be computed accurately, either from an analytic model or from a learned model Nguyen-Tuong and Peters [2011]. As a result, only the model of the additional term due to contact forces $\boldsymbol{\tau}_{ext}$ has to be learned. In this paper, we consider a robot that is provided with skin measurements $\boldsymbol{s}$ from the tactile sensors, and force measurements $\boldsymbol{F}$ from the force torque sensors (FTS). A visual representation of these relevant components is shown in Figure 3.2. Predicting the contact forces $\boldsymbol{\tau}_{ext}$ can be formalized as a regression problem

$$\boldsymbol{Y} = f(\boldsymbol{\theta}) + \epsilon\,, \tag{3.3}$$

with outputs $\boldsymbol{Y} = \boldsymbol{\tau}_{ext} = \sum_{i\in\mathscr{C}} \boldsymbol{J}_i(\boldsymbol{q})^T \boldsymbol{\gamma}_i$ and inputs $\boldsymbol{\theta} = [\boldsymbol{q}, \boldsymbol{s}, \boldsymbol{F}]$. Additionally, the i.i.d. Gaussian measurement noise ϵ has mean 0 and variance σ_n^2. Therefore, our regression problem becomes

$$\boldsymbol{\tau}_{ext} = \sum_{i\in\mathscr{C}} \boldsymbol{J}_i(\boldsymbol{q})^T \boldsymbol{\gamma}_i = f([\boldsymbol{q}, \boldsymbol{s}, \boldsymbol{F}]) + \epsilon\,. \tag{3.4}$$

The skin $\boldsymbol{s}$ is part of the input since contacts with different parts of the body lead to different effects in the dynamics: Intuitively, $\boldsymbol{s}$ is required to identify the position of the contact. The force/torque measurements $\boldsymbol{F}$ could be avoided if we were interested in learning contacts that do not change between training and test time, which would restrict us to static objects, such as a rigid floor, walls or stationary obstacles. However, this assumption is too limiting in our context, so we decided to include the force/torque measurements $\boldsymbol{F}$ in our model.

The resulting regression of Equation (3.4) is challenging due to nonlinearity and the high dimensionality of the input $\boldsymbol{\theta}$ (the skin measurements $\boldsymbol{s}$ alone account for hundreds or thousands of dimensions). Observing that the contact forces term $\boldsymbol{\tau}_{ext}$ is a sum over multiple contacts, we can tackle this challenging regression by exploiting the superposition of the contacts and, therefore, decomposing the $\boldsymbol{\tau}_{ext}$ into a sum of multiple regression problems, where each regression problem is a single contact. Since each contact is uniquely defined by its position, we can make the torque contribution of each single contact model independent of the skin, and use the skin only to decide which of the contact models is currently active. Hence, we can rephrase the regression of Equation (3.4) as the learning of the mixture-of-experts model ("mixture of contacts" in our case)

$$\boldsymbol{\tau}_{ext} = \sum_{i\in\mathscr{C}} \boldsymbol{J}_i(\boldsymbol{q})^T \boldsymbol{\gamma}_i = \sum_{j\in\mathscr{J}} f_j([\boldsymbol{q}, \boldsymbol{F}]) + \epsilon\,, \tag{3.5}$$

where $\mathscr{J}$ is the set of active experts f_j identified by a gating network. The purpose of the gating network is to select the experts (i.e., contacts) that are currently active, based on the skin input $\boldsymbol{s}$, and to add their contributions[3]. An overview of our proposed approach is illustrated in Figure 3.4.

Note that, in Equation (3.5), the skin input $\boldsymbol{s}$ is no longer explicitly part of the inputs of the experts since it is now only used in the gating network. As a result, each single expert f_j is now sufficiently low-dimensional to be modeled independently but at the same time the possibility of summing the contribution of each contact allows to account for complex behaviors. We model each separate expert f_j as a Gaussian process (GP) mapping $[\boldsymbol{q}, \boldsymbol{F}] \mapsto \boldsymbol{J}_j(\boldsymbol{q})^T \boldsymbol{\gamma}_j$. The design of the gating network is discussed in Section 3.3.2 while detailed information regarding GP models and their training are given in Section 3.3.3. An important design choice is the number of experts (i.e., contact models) to use. Generally, the more contact models the better, as each one would specialize in its own particular contact class (i.e., contacts that results is similar effects on the dynamics) resulting in more accurate dynamics model, at the expense of computational cost. However, it is important to notice that already with a single contact model, there are limited generalization capabilities as demonstrated in Section 3.4.4. Moreover, since we treat all contact models as independent, it is always possible to train additional contact models on-demand.

3.3.2 Learning the Gating Network

The gating network can be considered a classifier $\mathscr{J} = g(\boldsymbol{q}, \boldsymbol{s}, \boldsymbol{F})$ that selects which contact is currently active. For simple tasks (e.g., with a low number of possible contact locations), this gating network can be manually designed (e.g., using

---

[3] In our experimental evaluation we consider, without loss of generality, the case of a single contact active at any given time.

thresholds on the raw activation of the taxels). Alternatively, for tasks that produce rich and complex tactile inputs an approach based on machine learning is more suitable. In Section 3.4.3, we discuss and evaluate an approach for the automatic learning of such gating networks by using support vector machine (SVM) Cortes and Vapnik [1995].

The tactile data used to train a classifier is high-dimensional, but inherently structured: We know that the data is generated from a 3-dimensional process since the activation of the skin directly depends on the x-y-z coordinates of the contacts. Therefore, we hypothesize the possibility of reducing the dimensionality of the problem closer to its intrinsic dimensionality. For this purpose, in Section 3.4.3, we explore the use of Deep Learning techniques to perform dimensionality reduction on the raw tactile inputs before training a classifier. In particular, the models that we employ are the Stacked AutoEncoders (SAE) Vincent et al. [2010], consisting of a sequence of autoencoders trained layer-by-layer. An autoencoder is defined as a two-layer neural network that transforms the input data $\boldsymbol{\theta} \in \mathbb{R}^n$ to a latent space $\boldsymbol{H} \in \mathbb{R}^m$ (typically lower dimensional $m < n$) and then re-projects the latent features back to the data space $\hat{\boldsymbol{\theta}}$ according to

$$\boldsymbol{H} = \sigma\left(\boldsymbol{W}\boldsymbol{\theta} + \boldsymbol{B}\right), \tag{3.6}$$

$$\hat{\boldsymbol{\theta}} = \sigma\left(\boldsymbol{W}^T\boldsymbol{H} + \boldsymbol{Bh}\right), \tag{3.7}$$

where $\sigma(\cdot)$ is the transfer function, $\boldsymbol{W}$ the weights, and $\boldsymbol{B}$ and $\boldsymbol{Bh}$ the biases. Such a network is trained unsupervisedly by minimizing a reconstruction loss function $\mathscr{L}$, e.g., the mean squared reconstruction error

$$\mathscr{L} = \frac{1}{n}\sum_{i=1}^{n}\left\|\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}\right\|^2, \tag{3.8}$$

where $\hat{\boldsymbol{\theta}}$ is the reconstructed data for the input $\boldsymbol{\theta}$. Once a AE is learned, it is possible to map each input data to the lower-dimensional latent space $\boldsymbol{H}$ and subsequently train the next layer using the latent space as input. Once all the layers of the SAE are trained, we can generate the low-dimensional representation of the training set by projecting trough all the layers, and use it to learn a standard classifier (e.g. an SVM). Similarly, at running time, we can first compute the projections of the input data and subsequently use this low-dimensional representation as input of the classifier to obtain the active contacts. In our experiments, we used as transfer function $\sigma(\cdot)$ the rectified linear units (ReLu) Nair and Hinton [2010], and as optimizer RPROP Riedmiller and Braun [1993].

3.3.3 Gaussian Processes as Expert Models

Once the gating network activates one of the expert models to compensate for a specific family of contacts, the expert model has to predict the torque contribution for the ongoing contact. To learn such experts models we use Gaussian Processes due to their being probabilistic models and the possibility of incorporating physically grounded priors (e.g., from analytical contact models). Gaussian Processes Rasmussen and Williams [2006] are a state-of-the-art regression method also used in robotics to learn dynamics models Deisenroth et al. [2012] and for control **?**. In the context of this chapter, a GP is a distribution over inverse dynamics models

$$f \sim \mathscr{GP}\left(m_f, k_f\right), \tag{3.9}$$

fully defined by a prior mean m_f and a covariance function k_f. When learning the contact models $\boldsymbol{\tau}_{ext}$, we choose as prior mean $m_f \equiv \boldsymbol{\tau}_{ID}$ and as covariance function k_f the squared exponential with automatic relevance determination and Gaussian noise:

$$k(\boldsymbol{x}_p, \boldsymbol{x}_q) = \sigma_f^2 \exp\left(-\tfrac{1}{2}(\boldsymbol{x}_p - \boldsymbol{x}_q)^T \boldsymbol{\Lambda}^{-1}(\boldsymbol{x}_p - \boldsymbol{x}_q)\right) + \sigma_w^2 \delta_{pq} \tag{3.10}$$

where $\boldsymbol{\Lambda} = \mathrm{diag}([l_1^2, \ldots, l_D^2])$ and δ_{pq} is the Kronecker delta (which is one if $p = q$ and zero otherwise). Here, l_i are the characteristic length-scales, σ_f^2 is the variance of the latent function $f(\cdot)$ and σ_w^2 the noise variance. The purpose of $\sigma_w^2\delta_{pq}$ is to model (and identify) the presence of the Gaussian noise ϵ. In our experiments, when learning contact models, the input is defined as $\boldsymbol{\theta} = [\boldsymbol{q}, \boldsymbol{F}]$ and the outputs (observations) are the torques, such that $\boldsymbol{y} = \boldsymbol{\tau}$. Hence, given n training inputs $\boldsymbol{X} = [\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n]$ and corresponding training targets $\boldsymbol{Y} = [y_1, \ldots, y_n]$, we define the training data set $\mathscr{D} = \{\boldsymbol{X}, \boldsymbol{Y}\}$. Training the GP corresponds to finding the good hyperparameters $\boldsymbol{\omega} = [l_i, \sigma_f, \sigma_w]$, which can be done by the standard procedure of maximizing the marginal likelihood Rasmussen and Williams [2006].

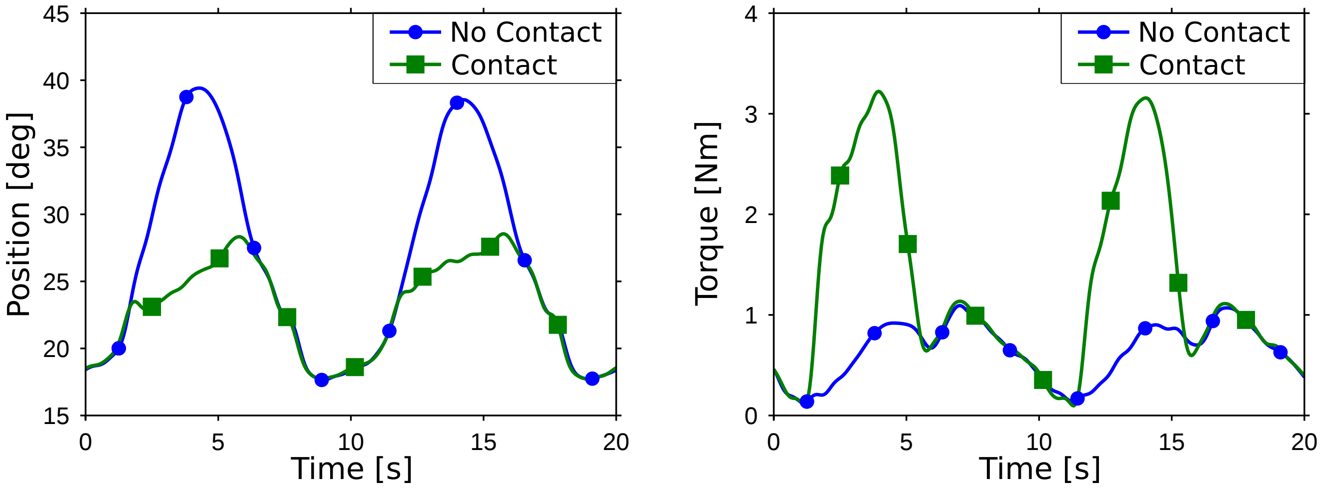
The GP yields the predictive distribution over torques for a new input $\boldsymbol{x}_* = [\boldsymbol{q}_*, \boldsymbol{F}_*]$

$$p(y|\mathscr{D}, \boldsymbol{\theta}_*) = \mathscr{N}\left(\mu(\boldsymbol{\theta}_*), \sigma^2(\boldsymbol{\theta}_*)\right), \tag{3.11}$$

where the mean $\mu(\boldsymbol{\theta}_*)$ and the variance $\sigma^2(\boldsymbol{\theta}_*)$ are

$$\mu(\boldsymbol{\theta}_*) = \boldsymbol{k}_*^T\boldsymbol{K}^{-1}\boldsymbol{Y}, \quad \sigma^2(\boldsymbol{\theta}_*) = k_{**} - \boldsymbol{k}_*^T\boldsymbol{K}^{-1}\boldsymbol{k}_*. \tag{3.12}$$

The entries of the matrix $\boldsymbol{K}$ are $K_{ij} = k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$, and we define $k_{**} = k(\boldsymbol{\theta}, \boldsymbol{\theta})$ and $\boldsymbol{k}_* = k(\boldsymbol{X}, \boldsymbol{\theta})$.



(a) Joint position **(b)** Torque

Figure 3.5.: Learning a single contact: Effects of a contact (green curve) compared to the free movement (blue curve) for a simple PD controller without contact compensation. The contact influence both the joint position (a) and the torque measured by the joint torque sensor (b).

3.3.4 Control in Presence of Contacts

In absence of contacts $\mathscr{C} = \{\emptyset\}$, and with a perfect model, we can define the task-space nonlinear feed-forward control

$$\boldsymbol{u} = \boldsymbol{\tau}_{ID}\,, \tag{3.13}$$

where the $\boldsymbol{u}$ are the torques given as motor command and $\boldsymbol{\tau}_{ID}$ are the torques computed from the inverse dynamics (or a learned model of it). To compensate for any noise or inaccuracies in the feed-forward dynamics, an additional feedback controller (impedance controller or PD controller) is added, such that

$$\boldsymbol{u} = \boldsymbol{\tau}_{ID} + \underbrace{\boldsymbol{K}_P\left(\boldsymbol{q}^{\text{des}} - \boldsymbol{q}\right) + \boldsymbol{K}_D\left(\dot{\boldsymbol{q}}^{\text{des}} - \dot{\boldsymbol{q}}\right)}_{\boldsymbol{\tau}_{PD}}\,. \tag{3.14}$$

where $\boldsymbol{K}_P$ and $\boldsymbol{K}_D$ are the gains matrices. Intuitively, the magnitude of the control signals coming from the PD controller $\boldsymbol{\tau}_{PD}$ can be used to measure the accuracy of the inverse dynamics model. Accurate inverse dynamics models will only need small corrections by the feedback controller during tracking of a desired trajectory, while inaccurate models will rely more heavily on it. In case of inaccurate models, increasing the PD gains can still lead to acceptable tracking performance, but often at the expense of safety for both robot and environment, and compliance. Additionally, high gains may require very large forces that are often infeasible on real robots, and that may destabilize the system.

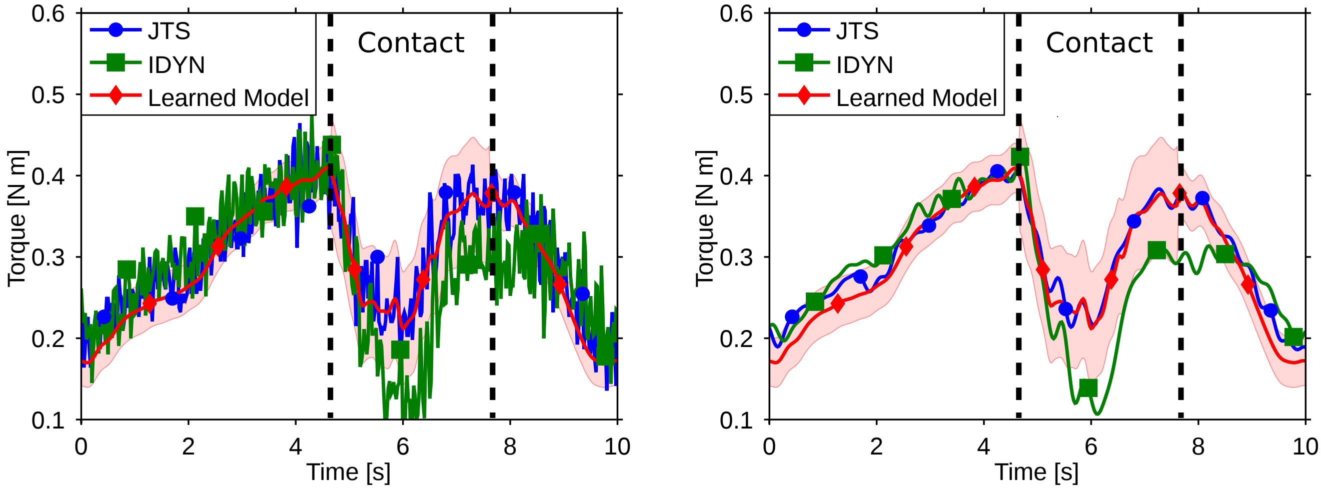
For contacts, we can further generalize the controller of Equation (3.14) to

$$\boldsymbol{u} = \boldsymbol{\tau}_{ID} + \boldsymbol{\tau}_{PD} + \boldsymbol{\tau}_{ext}\,, \tag{3.15}$$

where $\boldsymbol{\tau}_{ext}$ is the contribution from the contacts estimated using the learned model from Equation (3.5). In the experimental evaluations presented in Section 3.4.4 we will compare the performance of all these three control schemes.

3.4 Experimental Results

In this section, we present the experimental results obtained on the humanoid robot *iCub*. In Section 3.4.1, we describe the experimental setting including the robot *iCub* and its artificial skin. In Section 3.4.2, we demonstrate the modeling capabilities of our mixture-of-contacts approach on a number of experiments. In particular, we show that 1) We can accurately learn a single-contact model; 2) A single learned model (i.e., an expert) is robust to small changes in the position of the contact; 3) Our approach generalizes to multiple contacts by combining models of single contacts. In Section 3.4.3, we discuss the automatic learning of the gating network and the use of dimensionality reduction techniques to learn from raw skin measurements. In Section 3.4.4, we demonstrate that our approach reduces the tracking error when applied to control tasks in presence of unknown dynamics obstacles.



(a) Real data. **(b)** Filtered visualization.

Figure 3.6.: Learning a single contact: Comparison of the torques in presence of contact as measured at the elbow by the JTS, estimated by *iDyn*, and by our learned model (shown as mean ± 2 std). Our learned model better predicts the torque measured by JTS (a). Additionally, due to the identification of the noise in the model, its prediction is smoother compared to both the noisy JTS measurements and the prediction from *iDyn*. For visualization purposes we also show the predictions after filtering JTS and *iDyn* (b).

	Method	Shoulder 1 [Nm]	Shoulder 2 [Nm]	Elbow [Nm]
Full trajectory	*iDyn*	$0.09 \pm 1.1 \times 10^{-3}$	$0.16 \pm 1.8 \times 10^{-3}$	$0.05 \pm 7.4 \times 10^{-4}$
	Our model	$\mathbf{0.04 \pm 5.6 \times 10^{-4}}$	$\mathbf{0.07 \pm 9.8 \times 10^{-4}}$	$\mathbf{0.02 \pm 3.1 \times 10^{-4}}$
Contact only	*iDyn*	$0.07 \pm 3.1 \times 10^{-3}$	$0.13 \pm 5.7 \times 10^{-3}$	$0.08 \pm 3.0 \times 10^{-3}$
	Our model	$\mathbf{0.03 \pm 1.5 \times 10^{-3}}$	$0.12 \pm 5.9 \times 10^{-3}$	$\mathbf{0.03 \pm 1.3 \times 10^{-3}}$

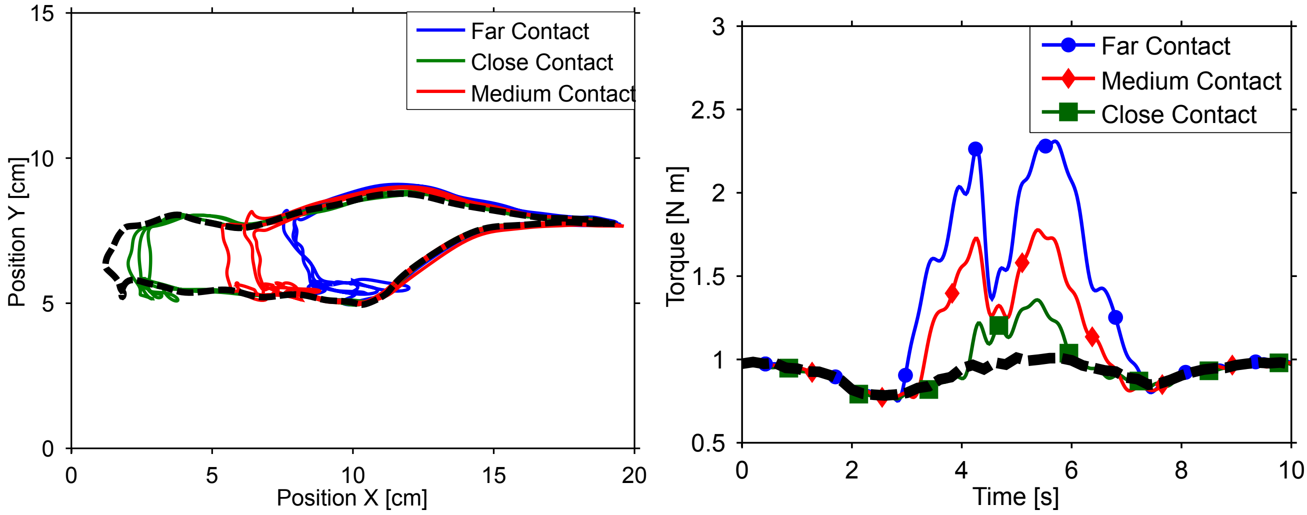
Table 3.1.: Learning a single contact: Mean and standard deviation of the mean for the RMSE of the test set for ground truth, predictions with the *iDyn* and our learned model. The learned model predicts the torques more accurately than *iDyn* for both the full trajectory and during the only contact phase.

3.4.1 Experimental Setting

We used the humanoid robot *iCub* as our experimental platform. The *iCub* has 53 degrees of freedom, is 104 cm tall and weighs about 24 kg Natale et al. [2013]. The robot is equipped with an inertial sensor in the head, and four 6-axis force/torque sensors placed proximally in the middle of legs and arms. Additionally, some *iCubs* are provided with an artificial skin, such as the *iCubDarmstadt01* shown in Figure 3.1. This artificial skin consists of many distributed tactile elements mounted on the robot covers for a total of over 2000 tactile sensors (*taxels*). An array of these tactile sensors is shown in Figure 3.3. Each tactile sensor provides a signal (in the interval $\in [0, 255]$) correlated to the vertical force applied to sensor, with some minor distortion due to the round shape of the contact surface and the dielectric foam layer covering the sensors. For more information about the artificial skin mounted on the *iCub* please refer to Cannata et al. [2008]. The standard version of the *iCub* is not equipped with joint/torque sensors. Only one of the *iCubs*, the *iCubParis02*, has been modified to mount partial joint torque sensing.

In our experiments, we control 4 DoF of the *iCub* arm: shoulder pitch, roll and yaw, and elbow. The skin input s from the forearm consists of 384 taxels. In the *iCub*, the joint torques are not estimated by the motors currents, but via an analytic model with identified dynamics parameters Traversaro et al. [2015], which updates the estimation based on the current inertial, tactile and force/torque measurements Ivaldi et al. [2011]. In the following, τ_{IDYN} denotes the joint torques estimated by the *iDyn* library used on the *Fox*, which we use as analytical model for comparison. For more detail on its contact detection and taxel calibration, we refer to Del Prete et al. [2011, 2012].

A low-level motor current controller implemented on the motor boards takes care of tracking the desired joint torques, sent through the robot interfaces. Two different *iCubs* have been used in our experiments: In Section 3.4.2, we used the *iCubParis02* since it is equipped with special joint torque sensors. In Section 3.4.4, we used the *iCubDarmstadt01*, which is the most recent version (v. 2.5) of the *iCub*, but not equipped with torque sensing.



(a) Task space **(b)** Torque

Figure 3.7.: Robustness of a Single Contact Model: Effects of the contact on the task space and the torque for the three different environmental objects: far contact, medium contact and close contact. The task without contact is shown as reference (**black dashed curve**).

	Method	Shoulder 1 [Nm]	Shoulder 2 [Nm]	Elbow [Nm]
Far contact	*iDyn*	$0.13 \pm 3.9 \times 10^{-3}$	$0.40 \pm 9.7 \times 10^{-3}$	$0.06 \pm 1.9 \times 10^{-3}$
	Our model	$\mathbf{0.06 \pm 1.9 \times 10^{-3}}$	$\mathbf{0.08 \pm 2.9 \times 10^{-3}}$	$\mathbf{0.03 \pm 8.0 \times 10^{-4}}$
Close contact	*iDyn*	$0.09 \pm 2.2 \times 10^{-3}$	$0.22 \pm 4.5 \times 10^{-3}$	$0.04 \pm 0.9 \times 10^{-3}$
	Our model	$\mathbf{0.06 \pm 1.4 \times 10^{-3}}$	$\mathbf{0.06 \pm 1.4 \times 10^{-3}}$	$\mathbf{0.02 \pm 6.3 \times 10^{-4}}$
Medium contact	*iDyn*	$0.10 \pm 2.8 \times 10^{-3}$	$0.32 \pm 6.7 \times 10^{-3}$	$0.05 \pm 1.3 \times 10^{-3}$
	Our model	$\mathbf{0.06 \pm 1.7 \times 10^{-3}}$	$\mathbf{0.12 \pm 4.7 \times 10^{-3}}$	$0.05 \pm 1.7 \times 10^{-3}$

Table 3.2.: Robustness of a Single Contact Model: Errors between the ground truth (JTS) and the predictions with either the *iDyn* and our learned model on the test set. A single expert is robust to small variations of the contact.

3.4.2 Modeling in presence of JTS

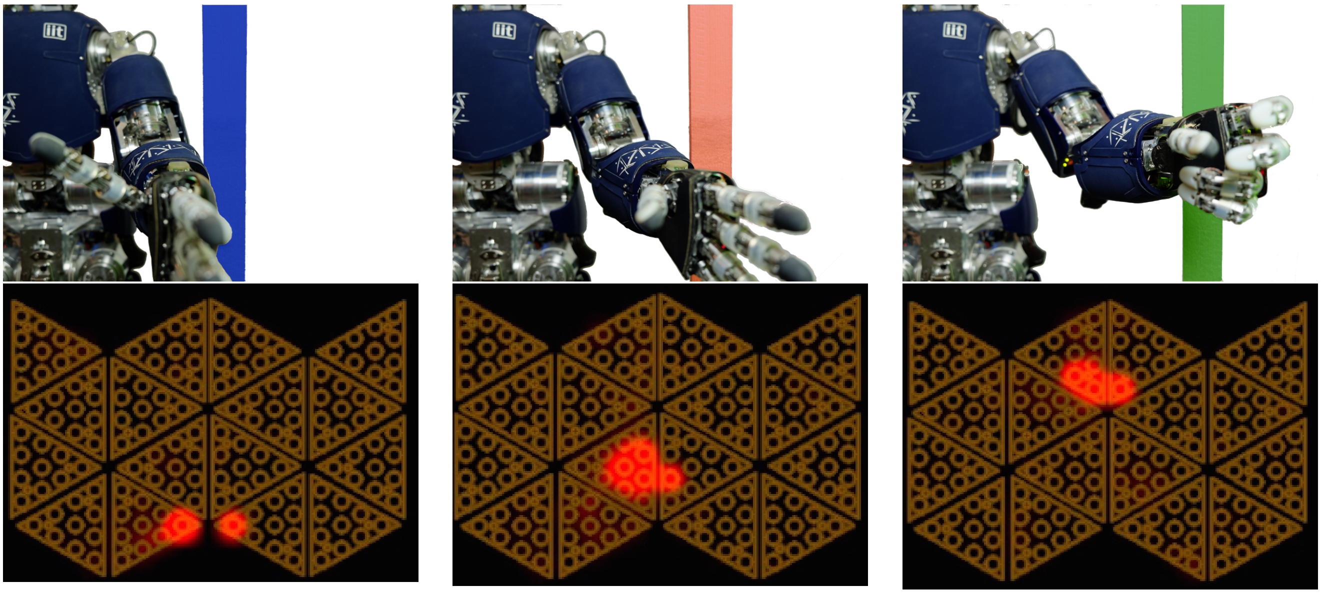
In the following experiments, we focus exclusively on the learning of the contact models. The aim is to demonstrate the improved accuracy of our learned models compared to the use of analytical models. For this purpose, in the following experiments we use the *iCubParis02* which is equipped with three additional Joint Torque Sensors (JTSs), two in the shoulder and one in the elbow. These JTSs are calibrated by computing the offset and gain trough least-square regression with respect to the output of *iDyn* in absence of contacts[4]. We consider these calibrated JTSs as ground truth measurements of the joint torques τ.

Learning a single contact

In this experiment, we consider the *iCub* making contact with a single object. The evaluation is performed on a simple tracking task with the *iCub*'s end-effector moving along a circular trajectory. We repeat the task twice: first without any contact and then with a contact at a fixed position. Figure 3.5 shows the effects of the contact in terms of position and torque during the tracking task. When the contact occurs, the position error increases considerably. As a result, the torque is increased to compensate for the contact. We collected ten repetitions of the trajectory with the contact and used eight of them to train the model. The remaining trajectories are used as test set to evaluate the predictive performances of our

---

[4] Calibrating the JTS on *iDyn* is the most favorable condition for *iDyn*. Different real-world calibration procedure would introduce modeling inaccuracies and reduce the performance of *iDyn*.



(a) Far contact

(b) Medium contact

(c) Close contact

Figure 3.8.: Robustness of a Single Contact Model. The three obstacles (far contact, medium contact and close contact) and the corresponding measurements from the forearm tactile sensors during the contacts.

	Method	Shoulder 1 [Nm]	Shoulder 2 [Nm]	Elbow [Nm]
Right contact	*iDyn*	$0.10 \pm 1.3 \times 10^{-3}$	$0.13 \pm 1.6 \times 10^{-3}$	$0.06 \pm 8.1 \times 10^{-4}$
	Our model	$\mathbf{0.04 \pm 6.3 \times 10^{-4}}$	$\mathbf{0.07 \pm 1.2 \times 10^{-3}}$	$\mathbf{0.02 \pm 2.7 \times 10^{-4}}$
Left contact	*iDyn*	$0.08 \pm 1.2 \times 10^{-3}$	$0.16 \pm 2.0 \times 10^{-3}$	$0.05 \pm 8.2 \times 10^{-4}$
	Our model	$\mathbf{0.03 \pm 5.7 \times 10^{-4}}$	$\mathbf{0.07 \pm 9.6 \times 10^{-4}}$	$\mathbf{0.02 \pm 2.8 \times 10^{-4}}$
Both contacts	*iDyn*	$0.10 \pm 1.3 \times 10^{-3}$	$0.11 \pm 1.4 \times 10^{-3}$	$0.07 \pm 8.4 \times 10^{-4}$
	Our model	$\mathbf{0.05 \pm 8.3 \times 10^{-4}}$	$\mathbf{0.10 \pm 1.6 \times 10^{-3}}$	$\mathbf{0.03 \pm 4.0 \times 10^{-4}}$

Table 3.3.: Learning Multiple Contacts: Root mean square error between the ground truth (JTS) and the predictions with the *iDyn* and our learned model on the test set. Our learned model predicts the torque more accurately than *iDyn*.

learned model. For this experiment we consider a single expert (the gating network still decides whether to activate the expert or not).

We compare the baseline joint torques (i.e., the JTS) to the estimation by the analytic model *iDyn* and the torques τ_{ID} predicted by our learned model. In Table 3.1, we report the root mean square error (RMSE) and the standard deviation of the mean of *iDyn* and our learned model for all the three joints. Additionally, we report both the error of the learned models (learned RBD plus learned contact model) during the full trajectory and exclusively *during* the contact. In five out of six cases, the learned model performs better than the analytic model. In the sixth case (contact only, shoulder 2), the performance of the learned model is similar to the analytic model. However, increasing the amount of data used for training may further increase the performance of the learned model. The predictions of the test set for the elbow joints are shown in Figure 3.6.

This experiment provides evidence that the classical rigid-body dynamics model $\tau_{RBD}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$ and the *iDyn* estimation (that also exploits proximal force/torque sensing) fail to sufficiently accurately estimate the joint torques when the robot is in contact with the environment. Moreover, we show that the learned contact model, when combined with the RBD model, provides a better approximation of the joint torques.

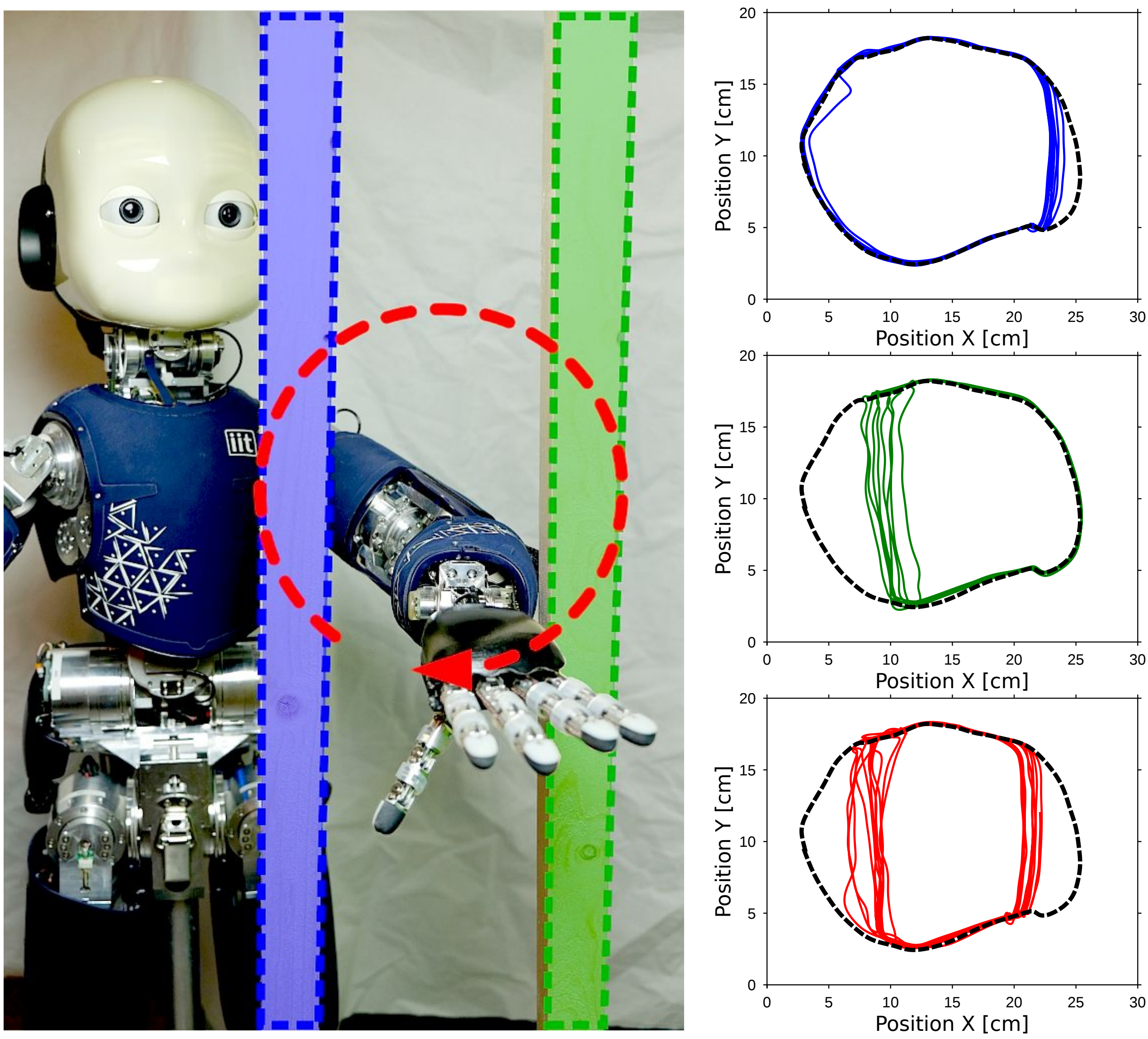


Figure 3.9.: Learning Multiple Contacts. The robot executes a circular trajectory with its left arm. The forearm collides alternatively with three objects causing the right contact, left contact or both contacts.

Robustness of a Single Contact Model

In the following, we show that the prediction performance of each GP expert is robust to small variations in the contact position. This robustness is important since the exact position of the obstacle does not need to be known in advance (within a single expert f_j). As in the previous experiment we consider a tracking task along a circular trajectory. However, this time the obstacle is placed at one of three different positions along the trajectory: close, medium and far. Each of these obstacles is shifted 2 cm along the horizontal axis. Obstacles at different positions along the trajectory lead to different effects in terms of both joint positions and torques signal, as clearly visible in Figure 3.7. Note that the skin input $\boldsymbol{s}$ is also affected, as shown in Figure 3.12. Hence, we could potentially learn a separate expert for each contact. However, we only consider a single expert here, as we want to demonstrate its generalization capabilities.

The contact model is learned using the data collected from Contact 1 and Contact 3 (far and close contacts) and as validation the data set generated from the *unseen* Contact 2 (medium) is used. In Table 5.1, the RMSE for all three contacts are reported for *iDyn* and our learned model, respectively. The results show that the learned model is robust to unseen contacts and performs equally well or better than the analytic model *iDyn*.

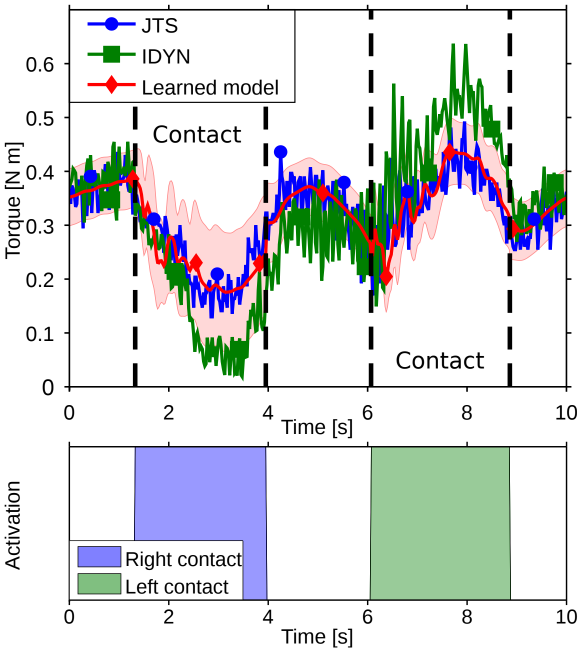


Figure 3.10.: Learning Multiple Contacts: Prediction of torques with multiple contacts and the corresponding activation of the gating network. Our mixture-of-experts model combines the single-contact models into a multiple-contact model.

Learning Multiple Contacts

After learning single contacts, we now show how to combine the learned models to adapt to unseen and more complex environments with multiple sequential contacts. We consider a scenario where the *iCub* performs a circular motion with its left arm. We initially performed two experiments with an object in collision either on the left and on the right of the reference trajectory (see Figure 3.9). With the data collected in these two contact cases, we trained two independent expert models f_1, f_2, one for each contact. We repeated the experiment, but this time with both left and right contacts and used this last unseen case to validate our models. Figure 3.10 shows an example of the prediction and the corresponding activation of the two contact models. During both the right and the left contact, the corresponding experts are activated by the gating network at the right time, and can therefore compensate for the presence of contacts. This experiment demonstrate that our approach can successfully combine the contributions of single contact models learned to generalize to unseen cases with multiple contacts. Table 3.3 reports the RMSE for the predictions. We notice that even in this experiment the experts accurately learn the effects of single contacts. Moreover, the gating network allows us to combine the experts to generalize to unseen environments, such as in the case of both contacts.

3.4.3 Learning the Gating Network

In the previous experiments, we employed a manually designed gating network to select the active experts f_j. However, designing such a gating network is not the goal of this paper, and can be time-consuming and require significant expert knowledge. In the following, we consider instead the automatic learning of such gating network and compare several approaches.

Learning the Gating Network using SVMs

In this experiment, we show that it is possible to automatically learn an SVM gating network capable of recognizing the presence of contacts better than a manually-tuned gating network. As ground truth to evaluate the performances, as well as for training the classifier, we labeled the data collected with one of the following labels: no contact, left contact, right contact. The manually-tuned gating network is based on thresholds of the activation of the skin input $\boldsymbol{s}$ and the force torque sensors $\boldsymbol{F}$. When training the SVM classifier we compare three different input combinations: 1) the *Full* dataset including both skin input $\boldsymbol{s}$ and proprioceptive data from the force torque sensors $\boldsymbol{F}$ and joint positions $\boldsymbol{q}$. 2) the *Skin* dataset consisting exclusively of the skin input $\boldsymbol{s}$. 3) the *Reduced* dataset consisting only of the proprioceptive data from the force torque sensors $\boldsymbol{F}$ and joint positions $\boldsymbol{q}$. Using k-fold cross validation with $k = 5$, we trained one SVM classifier

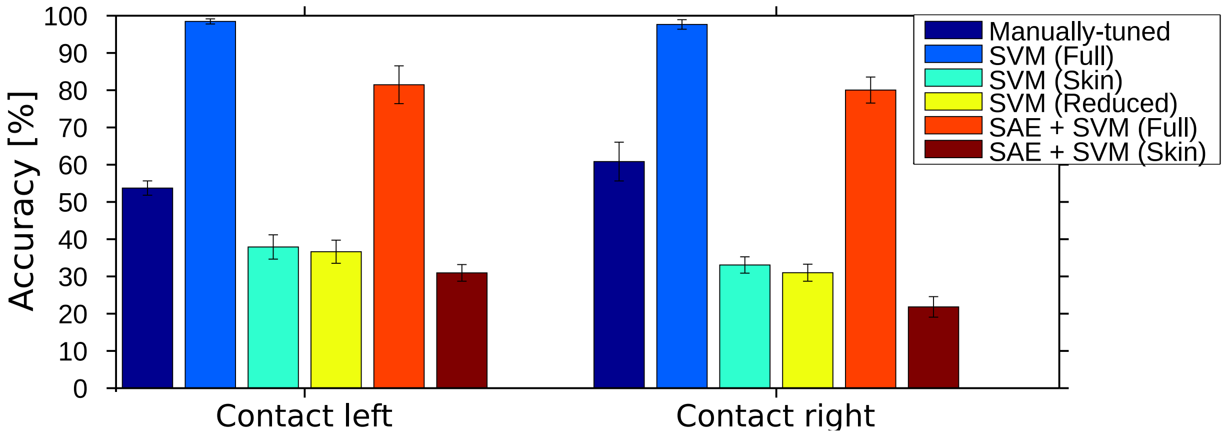


Figure 3.11.: Learning the Gating Network: Classification accuracy (mean ± std) for different types of gating networks. The most accurate gating network was learned using an SVM having as input the full skin/proprioception data $[s, F, q]$.

for each of these three datasets, having as output the contact labels (none, left, right). Figure 3.11 shows that the learned SVM using the full input state achieved an almost perfect classification accuracy and significantly outperformed the manually-tuned gating network. Interestingly, both using only the skin input and using only the proprioceptive input seem to be insufficient to learn an accurate classifier of contacts. This result was to be expected when using the proprioceptive input, since in our experiments the robot does not generate such high forces to unequivocally indicate the presence of contacts. For the skin input, this result is more surprising and, in Section 3.5, we discuss one possible motivation for such result. Overall, training the gating network (i.e., training the SVM classifier) required considerably less expert knowledge compared to manually designing and tuning the gating network, and obtained better performance. Therefore, we believe that data-driven approaches are generally more suitable than manually designing a gating network.

Dimensionality Reduction

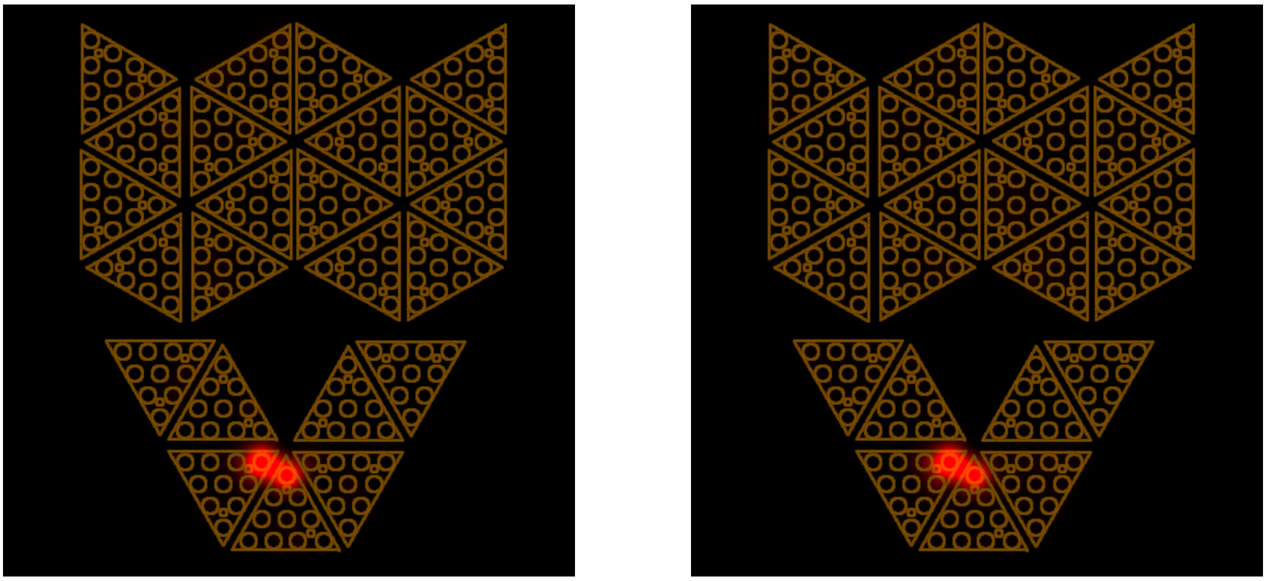
In this experiment, we demonstrate that it is possible to reduce the dimensionality of the skin data training unsupervisedly the stacked autoencoders described in Section 3.3.2 and still obtain good performance when automatically learning the gating network. We train an SAE having 3 layers with [600, 200, 15] neurons respectively using the data collected from the raw tactile sensors. To validate our model, we analyzed the reconstruction of the tactile data. In Figure 3.12 is shown an example of the reconstructed data from the low-dimensional representation. The similarity between the original data and the reconstructed suggests that the low-dimensional representation learned captured the relevant information content. Subsequently, we trained an SVM classifier from the low-dimensional features learned by the SAE. In Figure 3.11, it can be seen that although the performance of the SAE+SVM classifier trained on the Full dataset do not outperform the standard SVM, SAE+SVM outperforms the manually-tuned gating network. Similarly to the results obtained when training the simple SVM, using exclusively the tactile data prove insufficient to learn an accurate classifier of contacts. We hypothesize that the superiority of the SVM classifier in our specific tasks depends from the relatively small number of tactile sensors available on the forearm (384). However, it is important to note that the use of SVM classifier would not scale gracefully to thousands or millions of inputs. As such, we believe that the use of dimensionality reduction is in the long term crucial for the learning from thousands or millions of distributed tactile sensors.

3.4.4 Control without JTS

In this section, we present the experimental results obtained from a tracking task in the presence of contacts. First, we demonstrate that an accurate (contact-free) ID model can be learned. This IDM will be used as base for comparing the tracking performance in presence of contacts. Second, we demonstrate that our proposed model can be used to compensate for an unknown contact and achieve better tracking performance. Finally, we demonstrate the generalization capabilities of our learned model by testing it on a novel object.

Validation of the (Contact-free) Inverse Dynamics

Before learning the contact models, a pre-requisite of our approach is the existence of a (contact-free) rigid-body inverse dynamics model. In our case, we decided to learn such ID model from scratch using Gaussian Processes. The reason for



(a) Original data.

(b) Reconstructed data.

Figure 3.12.: Dimensionality Reduction. Example of activation of the tactile sensors for the forearm and its reconstruction after reducing the dimensionality of the data to 15 using a SAE. Reducing the dimensionality of the data does not significantly influence the quality of the signals and the reconstructed data are indistinguishable from the original ones.

Controller	Shoulder Pitch	Shoulder Roll	Shoulder Jaw	Elbow
PD	0.72 ± 0.60	4.64 ± 2.07	3.68 ± 1.73	2.37 ± 0.11
PD + IDM	$\mathbf{0.46 \pm 0.29}$	$\mathbf{2.74 \pm 1.10}$	$\mathbf{1.38 \pm 1.42}$	$\mathbf{0.59 \pm 0.18}$

Table 3.4.: Validation of the (Contact-free) Inverse Dynamics: Tracking error (in degrees) in absence of contacts (mean ± std). The use of the learned IDM proved to be beneficial in reducing the error.

this choice are the unmodeled dynamics $\epsilon(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$, which introduce substantial errors even in absence of contacts. In this subsection, we present an experimental validation of the learned (contact-free) inverse dynamics. To validate the learned IDM we firstly compared the tracking error, in absence of obstacles. The task used for the experimental validation of the proposed approach consists of a tracking task with the left arm of the *iCub* where each of the four joints of the arm should follow a pre-designed trajectory. In Table 3.4, we collected the tracking errors obtained averaging over 20 repetitions. The results shows that the learned IDM reduces the tracking error, compared to the simple PD controller. The visualization of Figure 3.13 shows the average error tracking for the shoulder jaw. As a second comparison, we analyzed the contribution to the desired torque from the PD controller τ_{PD}. In Figure 3.14 it is visible that when using the learned ID model, τ_{PD} is drastically reduced. Both results suggests that the learned IDM is accurate and useful for the desired tracking task.

Trajectory Tracking in Presence of Contacts

For the validation of the inverse dynamics with contacts models, we consider the same trajectory tracking task used in the previous evaluation. However, in these experiments we included the presence of a object along the trajectory, specifically a filled 1-liter bottle of water. For classical controllers, when an contact occurs, the rigid body inverse dynamics does not account for this variation. As a result, the tracking error increases, and the contribution of the PD feedback controller increases to compensate for this tracking error. In this scenario we demonstrate that it is possible to use a learned contact model to improve the tracking accuracy when unforeseen objects are encountered along the path.

We first performed the tracking experiment (with the obstacle) once using as controller the PD + IDM, in order to collect training data. Using these data collected, we trained the contact model using as target the difference between the desired torque and the inverse dynamics model contribution τ_{ID} (we here assume a single contact model). Following, we

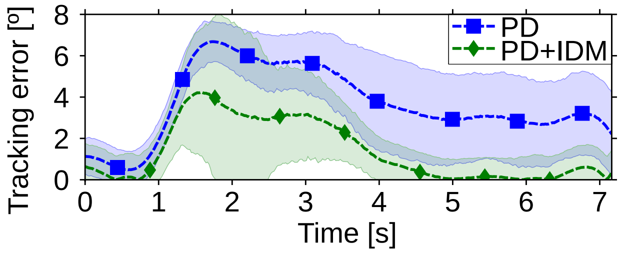


Figure 3.13.: Validation of the (Contact-free) Inverse Dynamics: Average tracking error in the absence of contacts. Using the learned IDM plus PD (green curve) reduce the error compared to using the PD controller only (blue curve). We conclude that the learned IDM is meaningful.

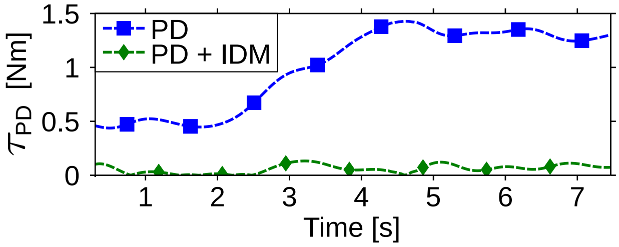


Figure 3.14.: Validation of the (Contact-free) Inverse Dynamics: Average torque contributed by the PD term τ_{PD} in absence of obstacle (filtered for visualization purposes). τ_{PD} drastically decrease when using the learned IDM, therefore suggesting that the learned IDM is accurate.

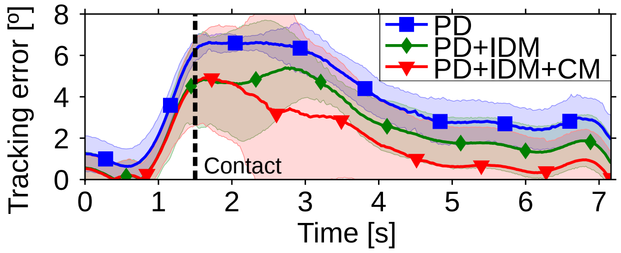


Figure 3.15.: Trajectory Tracking in Presence of Contacts: Average tracking error in presence of obstacle. Our approach using PD + IDM + CM (red curve) outperformed both simple PD controller (blue curve) and PD + IDM (green curve).

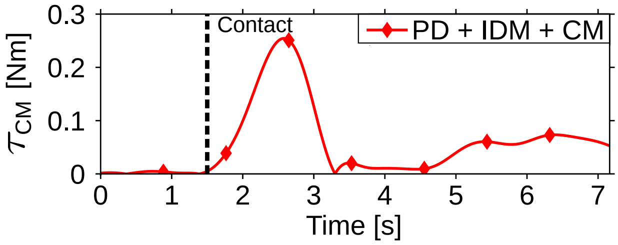


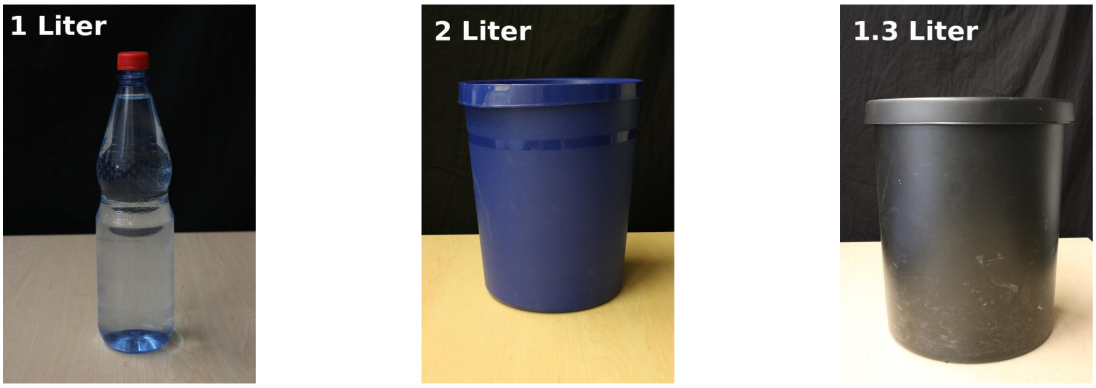
Figure 3.16.: Trajectory Tracking in Presence of Contacts: Average torque contributed by the contact model term τ_{ext} in presence of obstacle (filtered for visualization purposes).

repeated the experiment comparing the full controller PD + IDM + CM, the simple PD, and PD + IDM. In Table 3.5 the tracking errors for the three control schema are shown for all the four joint, averaged over 20 repetitions. The presence of the contact model proved beneficial in terms of tracking error for three out of four joints. For the elbow, the PD + IDM performed slightly better. However, it is important to note that the joints that are most involved in the trajectory are the shoulder pitch and shoulder jaw. Figure 3.15 shows the average tracking error (for the shoulder jaw) for the considered control modes. It is clearly visible that after about 1 second from the contact with the obstacle, the contact model reduce the tracking error. From these results we can conclude that using the additional learned contact model is beneficial. The profile of the average torque contributed by τ_{ext} is shown in Figure 3.16.

One limitation of the current results is that the gating network does not always recognize the presence of contacts. This is due to hardware limitations of the *iCub* skin. In fact, the current skin presents two main issues: 1) Noise and electrical glitches due to the fabrication process, that sometimes make one isolated taxel to saturate the signal as if was solicited by a contact. 2) Mechanical compliance of the skin and low sensitivity of the sensors. When applying a small force, the soft skin layer acts as a filter, causing a delay in the contact detection and a reduced spatial detection of the contact over many taxels. The sum of these two issues resulted in the introduction of small errors and delays for the identification of contacts and therefore the use of the additional contact model. This delay, which we estimate at around 500 ms, is visible in both Figure 3.15 and Figure 3.16. Often even for a human expert it is hard to distinguish between measurement noise and a real contact when observing short (i.e., less than 100 ms) windows of data. A second issue that we encountered during the training of both inverse dynamics model and contact models, is that the torques applied from the basic PD controller implemented on the *iCub* suffer from a high noise. As a result, we had to filter the desired torques. However, filtering the desired torques inevitably introduces small errors in the learned models. We expect that smoother desired torques would result in a more accurate learned model and even lower tracking error.

Controller	Shoulder Pitch	Shoulder Roll	Shoulder Jaw	Elbow
PD	0.50 ± 0.40	4.67 ± 2.52	3.78 ± 1.91	2.04 ± 0.11
PD + IDM	0.49 ± 0.38	3.96 ± 2.28	2.63 ± 1.65	$\mathbf{0.25 \pm 0.18}$
PD + IDM + CM	$\mathbf{0.46 \pm 0.31}$	$\mathbf{3.34 \pm 1.68}$	$\mathbf{1.81 \pm 1.55}$	0.46 ± 0.19

Table 3.5.: Trajectory Tracking in Presence of Contacts: Tracking error (in degrees) in presence of contact (mean ± std).



(a) Object 1 **(b)** Object 2 **(c)** Object 3

Figure 3.17.: Generalization of the Contact Model. The three objects used as obstacles and their corresponding weight. The objects also present different tactile feature due to changes in size and material. Object 1 and 2 were used during the training, while Object 3 was used for testing.

Generalization of the Contact Model

To analyze the generalization capabilities of a single contact model, we experimented with the use of obstacles not included in the training data. In a similar manner to the previous experiment we collected data using the PD + IDM on two different obstacles: a 1-liter bottle of water and a bin filled with 2 liters of water (see Figure 3.17). Using the data collected from both these obstacles we trained the contact model. We tested the resulting learned model on a new obstacle, a bin filled with 1.3 liters of water. Table 3.6 report the tracking error (mean and standard deviation) measured over 25 experiments. It can be observed that for the two joints with the highest usage during the task (i.e., shoulder roll and jaw), the contact model considerably improves the tracking accuracy (respectively of 2.5° and 1.8°). A visualization of the tracking error for the shoulder roll is shown in Figure 3.18. However, for the other two joints we observed a small ($< 0.25°$) decrease of the tracking performance. Overall, these results suggest that the contact model is capable of generalizing to novel obstacles with different properties (e.g., weight and tactile footprint).

3.5 Discussion

A limitation of our current approach is the need to have ground truth torques to train the contact models. This limitation implies that some way to measure the torques applied is necessary, either using JTS as ground truth, or with an analytical model. It is not the case for our *Fox*, but in other robots the ground truth may come from the current measures. This limitation is shared with any method that tries to learn the inverse dynamics in a supervised way, with or without contacts Nguyen-Tuong and Peters [2011]. To overcome this limitation it is possible to control the motors directly in voltage and/or current, rather then in torques. By doing so, the voltage/current could be used to directly learn the inverse dynamics model, since these measures are at the same time controllable and observable. Unfortunately, neither of these low-level control modalities are currently available on the *iCub*.

A second limitation that we experienced during our experiments lies in the hardware capability of the tactile sensors employed on the *iCub*. The current technology is mainly limited by the measurement of the orthogonal component of the force applied to a taxel and by the low sensitivity of the sensors due to mechanical compliance and measurement noise. We believe that using better sensors capable of measuring the complete vector of the contact forces would allow to remove the need of FTS sensors for modeling dynamical objects. Moreover, improving the sensitivity of the sensors would lead to more accurate models and less delay in the identification of the contact, resulting overall in further improvements of the tracking performance.

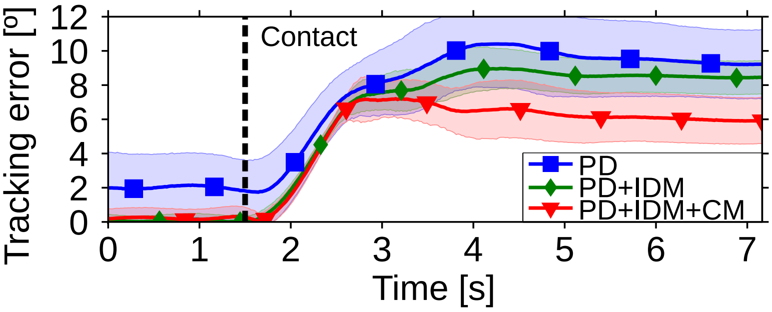


Figure 3.18.: Generalization of the Contact Model: Average tracking error in presence of a novel obstacle. Our approach using PD + IDM + CM (red curve) outperformed both simple PD controller (blue curve) and PD + ID model (green curve).

Controller	Shoulder Pitch	Shoulder Roll	Shoulder Jaw	Elbow
PD	**0.60 ± 0.54**	6.95 ± 3.36	4.97 ± 2.37	1.74 ± 0.10
PD + IDM	0.72 ± 0.31	5.69 ± 3.69	3.90 ± 1.75	**0.37 ± 0.18**
PD + IDM + CM	0.67 ± 0.42	**4.46 ± 2.77**	**3.13 ± 1.91**	0.59 ± 0.23

Table 3.6.: Generalization of the Contact Model: Tracking error (in degrees) in presence of an object never observed before (mean ± std). Even with novel objects our approach improve the performance of the two most used joints (shoulder roll and jaw) compared to simple PD and PD + IDM controllers.

3.6 Summary

Whole-body control strategies that exploit contacts are crucial for balancing and stabilization, and to increase the number of potential actions that the robot is able to execute, e.g., creating a contact in a cluttered environment to reach for distant objects. However, such control strategies need accurate models of the system's inverse dynamics incorporating contacts, which is challenging in presence of unknown and dynamic contacts and contacts with an environment of unknown properties. The use of force/torque sensors at the contact locations can be unfeasible, while the use of tactile sensors typically requires a spatial calibration to obtain the position of each individual sensor.

We introduced a novel data-driven mixture-of-experts approach based on Gaussian processes for learning inverse dynamics models in presence of contacts. We evaluated our model on the *Fox* humanoid robot using tactile sensors and force/torque sensors as model inputs. We showed that the model accurately predicts contact forces and outperforms a state-of-the-art analytical approach used to estimate the joint torques in the *Fox*. The estimation from the learned model does not rely on dynamic parameters, but it is completely data-driven and based on tactile sensors and force/torque sensors. Importantly, our approach does not require spatially calibrated models of the skin Del Prete et al. [2011, 2012] and is able to deal with high-dimensional artificial skin with hundreds or thousands of *taxels*.

Our work is a first step towards such approaches where it will be possible to take a handful of miniaturized tactile sensors, spread them on the structure of the robot, and quickly start learning from them, without having to know where each one of the millions of individual sensors is located.

4 Bayesian Optimization for Learning Gaits under Uncertainty

> Imagination is more important than knowledge. For knowledge is limited, whereas imagination embraces the entire world, stimulating progress, giving birth to evolution
>
> Albert Eintstein

In the previous chapter, we discussed how to learn dynamics model and use them for control. However, not for all the robots it is possible to learn a dynamic model. For robots that are under-sensed (i.e., robots which do not have sufficient sensors to fully capture the state of the system), it is thus more convenient to employ control schema which do not rely on the existence of a dynamic model. Nonetheless, even for such robots, it is still possible to improve the performance of the controller by appropriately tune its parameters. In this chapter, we study the use of Bayesian optimization for automatic black-box optimization of the controller's parameters. In particular, we evaluate BO on an under-sensed dynamic bipedal walker.

4.1 Introduction

Bipedal walking and running are versatile and fast locomotion gaits. Despite its high mobility, bipedal locomotion is rarely used in real-world robotic applications. Key challenges in bipedal locomotion include balance control, foot placement, and gait optimization. In this chapter, we focus on **gait optimization**, i.e., finding good (controller) parameters for the gait of a robotic biped.

Due to the partially unpredictable effects of and interactions among the gait parameters, gait optimization is often an empirical, time-consuming and strongly robot-specific process. In practice, gait parameter optimization often translates into a trial-and-error process, which requires an educated guess by a human expert or a systematic but time-consuming parameter search. As a result, gait optimization may require considerable expert knowledge, engineering effort and time-consuming experiments. Additionally, the effectiveness of the resulting gait strongly depends on the assumptions made during the controller design process: A change in these conditions, often requires searching for new, more appropriate, gait parameters. Examples of possible change of these assumptions are a change in the environment (e.g., different surfaces), a variation in the hardware response (e.g., hardware wear and tear, replacement of a motor or differences in the calibration) or the choice of a performance criterion (e.g., walking speed, energy efficiency, robustness). Hence, to deploy walking robots in the real world, it is essential to reduce the dependence on expert knowledge and automate the gait optimization process.

The search for gait parameters can be formulated as an optimization problem. Such a problem formulation, in conjunction with an appropriate optimization method, allows to automate the search for optimal gait parameters and reduces the need for engineering expert knowledge. To date, automatic gait optimization methods have been used for designing efficient gaits in locomotion Chernova and Veloso [2004], Gibbons et al. [2009], Kulk and Welsh [2011]. Commonly, these methods only find locally optimal solutions and do not take sources of uncertainty (e.g., measurement noise) into account.

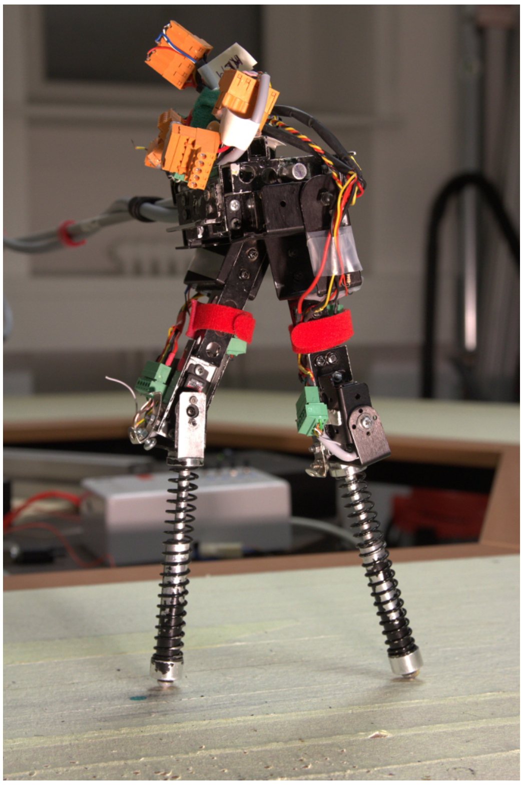


Figure 4.1.: The bio-inspired dynamical bipedal walker *Fox*. Using Bayesian optimization, we found reliable and fast walking gaits with a velocity of up to 0.45m/s.

Moreover, many optimization methods require a high number of function evaluations to find a good solution. Since each function evaluation requires an experiment with the robot, standard optimization methods are time-consuming and will eventually cause severe wear and tear on the robot, rendering these methods economically impractical. In practice, it is often essential to keep the number of robot experiments as small as possible.

To overcome this practical limitation on the number of possible interactions with the robot, we propose to use **Bayesian optimization** for efficient bipedal gait optimization. Bayesian optimization is a state-of-the-art global optimization method Jones [2001], Kushner [1964], Osborne et al. [2009] that can be applied to problems where it is vital to optimize a performance criterion while keeping the number of evaluations of the system small, e.g., when an evaluation requires an expensive interaction with a robot. Bayesian optimization makes efficient use of past interactions (experiments) by learning a probabilistic surrogate model of the function to optimize. Subsequently, the learned surrogate model is used for finding optimal parameters *without* the need to evaluate the expensive (true) function. By exploiting the learned model, Bayesian optimization, therefore, often requires fewer interactions (i.e., evaluations of the true objective function) than other optimization methods Jones [2001]. Bayesian optimization can also make good use of prior knowledge, such as expert knowledge or data from related environments or hardware, by directly integrating it into the prior of the learned surrogate model. Moreover, unlike most optimization methods, it can re-use any collected interaction data set, e.g., whenever we want to change the performance criterion.

Bayesian optimization has been successfully applied to sensor-set selection Garnett et al. [2010], gait optimization for quadrupeds Lizotte et al. [2007] and snake robots Tesch et al. [2011] and automatic algorithm configuration Hutter et al. [2011], Snoek et al. [2012a].

This chapter substantially extends our previous work on Bayesian optimization for robotics Calandra et al. [2014], Calandra et al. [2014c]. In Calandra et al. [2014], Bayesian optimization was applied to gait optimization of a bipedal robot. Three acquisition functions and the effect of fixed versus automatic hyperparameter selection were analyzed. Our extensive evaluation with more than 1,800 experiments with the robot shown in Figure 4.1 highlights the practicality and exposes strengths and weaknesses of commonly used acquisition functions. In Calandra et al. [2014c], we considered a more challenging set-up with a higher number of parameters. We successfully applied Bayesian optimization for automatic gait optimization of up to eight parameters on a bipedal robotic walker.

In this chapter, we additionally formalize the problem of automatic gait optimization and discuss the practicality of commonly used optimization methods. Furthermore, we analyze *a posteriori* the quality of the models learned. The results of this analysis motivates the need for an efficient global optimization algorithm and give insights into the effects and correlations between the parameters.

4.2 Parameter Optimization under Uncertainty in Robotics

In this section, we formalize automatic parameter optimization under uncertainty in the context of robotics. Moreover, we discuss classical optimization methods and related work in the context of gait optimization.

4.2.1 Parameter Optimization in Robotics

The search for good controller parameters $\boldsymbol{\theta}^*$ can be formulated as an optimization problem, such as the minimization

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta} \in \mathbb{R}^d}{\arg\min}\, f(\boldsymbol{\theta}) \,, \tag{4.1}$$

of an objective function $f(\cdot)$ with respect to the controller parameters $\boldsymbol{\theta} \in \mathbb{R}^d$[1].

In robotics, the objective function f often encodes a single performance criterion, such as precision, speed, energy efficiency, robustness or a mixture of them, while $\boldsymbol{\theta}$ are the parameters of an existing gait. In gait optimization, the relevant criteria typically are walking speed, energy efficiency, robustness or a mixture of them. Optimizing analytically the objective function f is typically infeasible since the relation between the controller parameters and the objective function is unknown. Hence, we need to use numerical black-box optimization where evaluating the objective function f for a given set of parameters requires a physical interaction with the robot.

The general parameter optimization problem in robotics (as well as our considered gait optimization task) possesses the following properties:

- **Zero-order objective function.** Each evaluation of the objective function f returns the value of the function $f(\boldsymbol{\theta})$, but no information about the gradient $\nabla_{\boldsymbol{\theta}} f = \mathrm{d}f(\boldsymbol{\theta})/\mathrm{d}\boldsymbol{\theta}$ with respect to the parameters $\boldsymbol{\theta}$.

---

[1] In this thesis, we consider only continuous parameters. Bayesian optimization does however also allow the use of discrete variables [Hutter et al., 2011]

Method	Order optimizer	Stochasticity assumption	Global optimizer	Re-usability evaluations
Grid Search	Zero-order	No*	Global	Limited
Pure Random Search	Zero-order	No*	Global	Yes
Gradient-descent family	First-order†	No*	Local	No
Bayesian Optimization	Zero-order	Yes	Global	Yes
Evolutionary Algorithms	Zero-order	No*	Global	No
Particle Swarm	Zero-order	No*	Global	No

Table 4.1.: Optimization methods in robotics: Properties of various optimization methods commonly used for optimization in robotics. As discussed in Section 4.2.1, the ideal optimizer for robotic applications should be global, zero-order, and assuming stochasticity.
(*) Extensions exist for the stochastic case.
(†) First or second order.

- **Stochastic objective function.** The evaluation of the objective function is inherently stochastic due to noisy measurements, variable initial conditions and system uncertainties (e.g., slack). Therefore, any suitable optimization method needs to account for the fact that two evaluations of the same parameters can yield two different values.

- **Global optimization.** No assumption can be made about the number of local minima or the convexity of the objective function f. However, ideally we seek the global minimum of the objective function.

These characteristics render this family of problems a challenging optimization task. Additionally, in the context of robotics the number of experiments that can be performed on a real system is small. Each experiment can be costly, require a long time, and it inevitably contributes to the wear and tear of the robot's hardware. Therefore, the optimizer must be as experimentally-efficient as possible. As result, the capability of re-using past experiments (e.g., experiments with random parameters) is a desirable property to keep the number of experiments small.

4.2.2 Optimization Methods in Robotics

Commonly used algorithms in robotics include grid search, gradient descent, evolutionary algorithms and others. In the following, we present some of the most common optimization methods and we discuss the main limitations that make these algorithms unsuitable for robotic applications. Table 4.1 shows a summary of the methods discussed.

Due to its ease of use, the first and most used optimization method in robotics is grid search. Its main limitation is that it is an exhaustive search method and, thus, requires many experiments. In fact, the number of experiments n grows exponentially with the number of parameters d as $n = p^d$ where p is the number of experiments along each parameter. A valid alternative to grid search is pure random search Brooks [1958], which possesses statistical guarantees of convergence and often outperforms grid search, e.g., in high-dimensional problems where many irrelevant dimensions exist Bergstra and Bengio [2012]. Nonetheless, even pure random search requires a number of experiments that is impractical in many robotic applications.

Another family of optimization methods commonly used in robotics are first-order methods, such as gradient descent. The use of first-order optimization methods, which make use of gradient information, is generally desirable in optimization as they lead to faster convergence than zero-order methods. Thus, it is common in the case of zero-order objective functions to approximate the gradient using finite differences. However, finite differences requires evaluating the objective function f multiple times. Since each evaluation requires interactions with the robot, the number of robot experiments quickly becomes excessive, rendering also first-order methods (e.g., the whole family of gradient descent) unsuitable for our task.

Particle swarm and evolutionary algorithms are two common global optimization methods. Both make use a populations of particles (or individuals), which explore the parameter space. However, both methods typically need thousands or tens of thousands of experiments to find good solutions. Hence, they are not easily applicable to real robots.

Method	Locomotion
Grid Search	Gibbons et al. [2009], Yamane [2013]
Pure Random Search	-
Gradient-descent family	Kulk and Welsh [2011], Tedrake et al. [2004]
Bayesian Optimization	Calandra et al. [2014], Lizotte et al. [2007], Tesch et al. [2011]
Evolutionary Algorithms	Chernova and Veloso [2004], Tang et al. [2005]
Particle Swarm	Niehaus et al. [2007]

Figure 4.2.: Related work in robot locomotion: Various optimization methods and the corresponding work in robot locomotion where they are applied.

4.2.3 Related Work in Robot Locomotion

To date, various automatic gait optimization methods have been used in locomotion to design gaits including: gradient descent methods Kulk and Welsh [2011], Tedrake et al. [2004], evolutionary algorithms Chernova and Veloso [2004], Tang et al. [2005], particle swarm optimization Niehaus et al. [2007] and many others Geng et al. [2006], Gibbons et al. [2009], Hemker et al. [2009], Yamane [2013].

We now discuss the works that use surrogate models to optimize robot locomotion. In Hemker et al. [2009], a surrogate model optimization is performed on a bipedal robot. This approach is very similar in spirit to Bayesian optimization, however, the model used is not probabilistic. Hence, the acquisition function can only use a deterministic prediction and, therefore, will choose the next point to evaluate greedily. This choosing greedily typically leads to local and suboptimal solutions. In robot locomotion, Bayesian optimization has been applied to quadrupedal robot Lizotte et al. [2007], snake robots Tesch et al. [2011] and bipedal walkers Calandra et al. [2014]. In Lizotte et al. [2007], two gait optimization criteria are considered for a Sony AIBO ERS-7 quadrupedal robot: once with respect to the maximum walking speed and once for the maximum gait smoothness. As acquisition function the authors used Probability of improvement (which we discuss in Section 4.3.2), and as a model a standard Gaussian process. The hyperparameters of the GP were manually selected by a human expert at the beginning of the optimization. Correctly fixing the hyperparameters generally simplifies the optimization process and, therefore, speeds up the optimization Lizotte et al. [2012]. Nonetheless, it requires a deep knowledge of the optimization task, which is typically an unrealistic assumption. In Tesch et al. [2011], Bayesian optimization is used to optimized the gait of a snake robot. The authors used expected improvement as acquisition function (which we discuss in Section 4.3.2).

4.3 Introduction to Bayesian Optimization

Bayesian optimization (BO) is a global optimization method Brochu et al. [2010], Kushner [1964], Osborne et al. [2009] based on response surface (i.e., surrogate model). Bayesian optimization has been re-discovered multiple times by different communities and is also referred to as efficient global optimization (EGO) Jones et al. [1998] and sequential kriging optimization (SKO) Huang et al. [2006].

Response surface-based optimization methods iteratively create a data set $\mathcal{D} = \{\boldsymbol{\theta}, f(\boldsymbol{\theta})\}$ of evaluated parameters $\boldsymbol{\theta}$ and the corresponding function evaluations $f(\boldsymbol{\theta})$ Jones [2001]. This data set is used to build a model $\hat{f}(\cdot) : \boldsymbol{\theta} \mapsto f(\boldsymbol{\theta})$, the **response surface**, that maps parameters $\boldsymbol{\theta}$ to corresponding function evaluations $f(\boldsymbol{\theta})$. The response surface is subsequently used to replace the optimization of Equation (5.1) with a “virtual” optimization process

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta} \in \mathbb{R}^d} \hat{f}(\boldsymbol{\theta}) \,. \tag{4.2}$$

In this context, “virtual” indicates that optimizing the response surface $\hat{f}(\cdot)$ with respect to the parameters $\boldsymbol{\theta}$ does only requires to evaluate the learned model, but not the true objective function f. Only when a new set of parameters $\boldsymbol{\theta}^*$ has been determined by means of the virtual optimization process of the response surface $\hat{f}$, it is eventually evaluated on the real objective function f.

A variety of different models, such as linear functions or splines Jones [2001], have been used in the past to create the response surface that model f. In Bayesian optimization, probabilistic models are used. The use of a probabilistic model allows us to model noisy observations and to explicitly take the uncertainty about the model itself into account, which makes the probabilistic model more robust to the effect of model errors. A probabilistic framework also allows to use

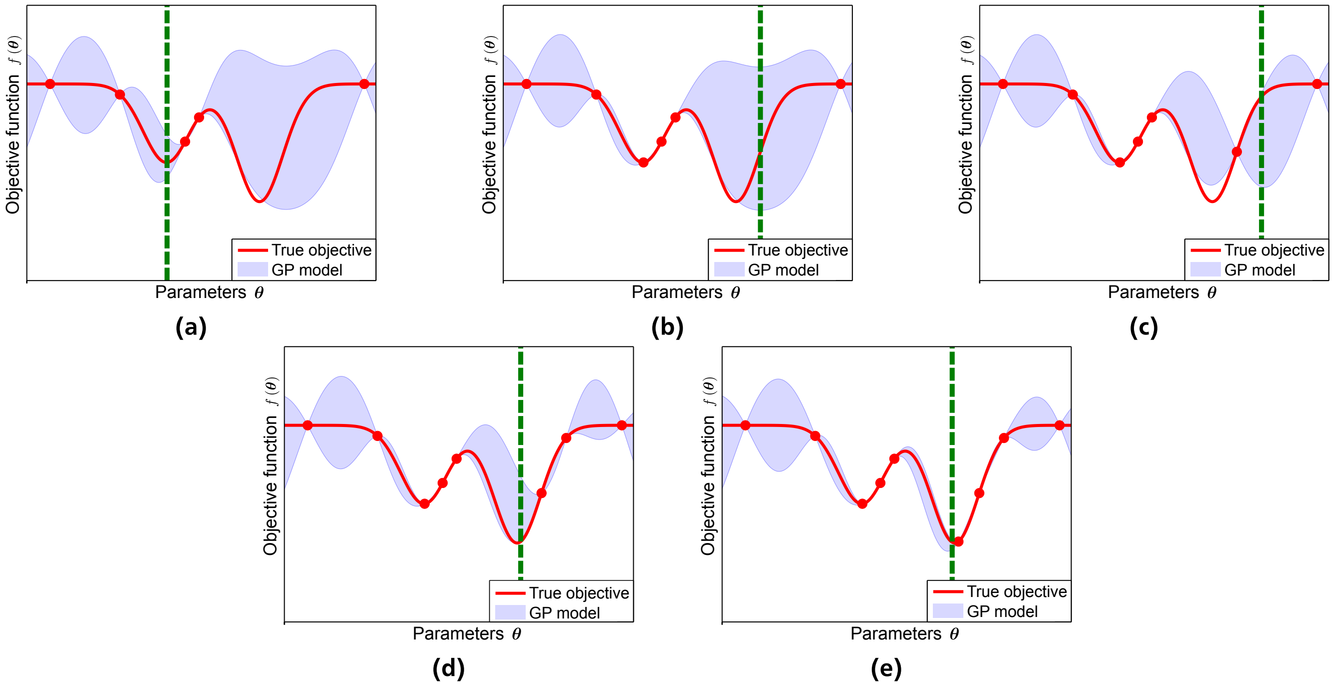


Figure 4.3.: Example of the Bayesian optimization process during the minimization of an unknown 1-D objective function f (red curve). The 95% confidence of the model prediction is represented by the blue area. The model is initialized with 5 previously evaluated parameters $\boldsymbol{\theta}$ and the corresponding function values $f(\boldsymbol{\theta})$. The location of the next parameter to be evaluated is represented by the vertical green dashed line. At each iteration, the model is updated using all the previously evaluated parameters (red dots). Bayesian optimization quickly found the global minimum of the unknown objective function, after a few iterations.

priors that encode available expert knowledge or information from related systems in a principled way. In the context of a walking robot, this knowledge could be a prior on the optimal parameters after a motor was replaced or the walking surface changed. The most common probabilistic model used in Bayesian optimization, and the one that we consider in this article, is a Gaussian process (GP) Rasmussen and Williams [2006]. Nonetheless, other probabilistic models are possible, such as random forests Hutter et al. [2011].

When using a probabilistic model, the response surface $\hat{f}(\cdot)$ in Equation (4.2) is a probability distribution. Therefore, the optimization of the response surface $\hat{f}$ would result in a multi-objective optimization problem. Hence, an **acquisition function** $\alpha(\cdot)$ is used for the virtual optimization of the probabilistic model. The purpose of the acquisition function is two-fold: First, it scalarizes the response surface (which is a probability distribution) onto a single function, the acquisition surface $\alpha(\boldsymbol{\theta})$, such that it can be optimized[2]. Thereby, the minimization of the objective function from Equation (5.1) can be rephrased as the minimization of the acquisition surface

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \alpha(\boldsymbol{\theta}) \,. \tag{4.3}$$

Second, the GP expresses model uncertainty, which is used to trade off exploration and exploitation in the optimization. This trade off between exploration and exploitation, and therefore the model uncertainty, is extremely important for the optimization process when we have only few function evaluations. For an example of the optimization process of Bayesian optimization see Figure 4.3.

Algorithm 1 summarized the main steps of Bayesian optimization: A GP model for the (unknown) objective function $f : \boldsymbol{\theta} \mapsto f(\boldsymbol{\theta})$ is learned from the data set $\mathscr{D} = \{\boldsymbol{\theta}, f(\boldsymbol{\theta})\}$ composed by the parameters $\boldsymbol{\theta}$ and the corresponding measurements $f(\boldsymbol{\theta})$ of the true objective function (Line 4 of Algorithm 1). This model is used to predict the response surface $\hat{f}$ and the corresponding acquisition surface $\alpha(\boldsymbol{\theta})$. Using a global optimizer, the minimum $\boldsymbol{\theta}^*$ of the acquisition surface $\alpha(\boldsymbol{\theta})$ is computed (Line 5 of Algorithm 1) without any evaluation of the true objective function, e.g., no robot interaction is required, see Equation (4.3). The parameters $\boldsymbol{\theta}^*$ are evaluated on the robot (Line 6 of Algorithm 1) and, together with the resulting measurement $f(\boldsymbol{\theta}^*)$, added to the data set $\mathscr{D}$ (Line 7 of Algorithm 1). Note that the optimizer can be initialized by past evaluations for the data set $\mathscr{D}$ (Line 1 of Algorithm 1), as well as by a prior of the GP model (Line 2 of Algorithm 1).

---

[2] The correct notation would be $\alpha\big(\hat{f}(\boldsymbol{\theta})\big)$, but we use $\alpha(\boldsymbol{\theta})$ for notational convenience.

Algorithm 1: Bayesian optimization

```
$\mathcal{D}$ $\longleftarrow$ if available: $\{\boldsymbol{\theta}, f(\boldsymbol{\theta})\}$
Prior $\longleftarrow$ Prior of the response surface
while optimize do
    Train a response surface from $\mathcal{D}$
    Find $\boldsymbol{\theta}^*$ that minimizes the acquisition surface $\alpha(\boldsymbol{\theta})$
    Evaluate $f(\boldsymbol{\theta}^*)$ on the real system
    Add $\{\boldsymbol{\theta}^*, f(\boldsymbol{\theta}^*)\}$ to $\mathcal{D}$
```

4.3.1 Gaussian Process Model for the Unknown Objective Function

To create the response surface model that maps $\boldsymbol{\theta} \mapsto f(\boldsymbol{\theta})$, we make use of Bayesian non-parametric GP regression Rasmussen and Williams [2006]. A GP is a distribution over functions $f \sim \mathcal{GP}(m_f, k_f)$, fully defined by a prior mean m_f and a covariance function k_f. We assume a model where we observe noisy function values $y = f(\boldsymbol{\theta}) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ is Gaussian noise. Both the prior mean m_f and the covariance function k_f are usually selected based on expert knowledge. Commonly used covariance functions include the squared exponential and Matérn covariance functions. In our experiments, we choose as prior mean $m_f \equiv \mathbf{0}$[3], while the chosen covariance function k_f is the squared exponential with automatic relevance determination and Gaussian noise

$$k_f(\boldsymbol{\theta}_p, \boldsymbol{\theta}_q) = \sigma_f^2 \exp\left(-\tfrac{1}{2}(\boldsymbol{\theta}_p - \boldsymbol{\theta}_q)^T \boldsymbol{\Lambda}^{-1} (\boldsymbol{\theta}_p - \boldsymbol{\theta}_q)\right) + \sigma_\epsilon^2 \delta_{pq} \tag{4.4}$$

with $\boldsymbol{\Lambda} = \mathrm{diag}([l_1^2, ..., l_D^2])$. Here, l_i are the characteristic length-scales, σ_f^2 is the variance of the latent function $f(\cdot)$ and σ_ϵ^2 the noise variance. The explicit consideration of the measurement noise is important in robotic applications, although common optimization algorithms consider the measurements noise free.

Given n training inputs $\boldsymbol{X} = [\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_n]$ and corresponding training targets $\boldsymbol{y} = [y_1, \ldots, y_n]$, we define the training data set $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$. Hence, the GP predictive distribution is

$$p(f(\boldsymbol{\theta}) | \mathcal{D}, \boldsymbol{\theta}) = \mathcal{N}\left(\mu(\boldsymbol{\theta}), \sigma^2(\boldsymbol{\theta})\right), \tag{4.5}$$

where the mean $\mu(\boldsymbol{\theta})$ and the variance $\sigma(\boldsymbol{\theta})$ are

$$\mu(\boldsymbol{\theta}) = \boldsymbol{k}_*^T \boldsymbol{K}^{-1} \boldsymbol{y}, \qquad \sigma^2(\boldsymbol{\theta}) = k_{**} - \boldsymbol{k}_*^T \boldsymbol{K}^{-1} \boldsymbol{k}_*, \tag{4.6}$$

respectively, where $\boldsymbol{K}$ is the matrix with $K_{ij} = k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$, $k_{**} = k(\boldsymbol{\theta}, \boldsymbol{\theta})$ and $\boldsymbol{k}_* = k(\boldsymbol{X}, \boldsymbol{\theta})$.

A practical issue in Bayesian optimization and GP modeling is the choice of the hyperparameters of the GP model. The hyperparameters of a GP model are the parameters of the covariance function, i.e., the characteristic length-scales l_i, the variance of the latent function σ_f^2 and the noise variance σ_ϵ^2. In gait optimization, these hyperparameters are often fixed a priori Lizotte et al. [2007]. In Lizotte et al. [2012], it is suggested that fixing the hyperparameters can considerably speed up the convergence of Bayesian optimization. However, manually choosing the hyperparameters requires extensive expert knowledge about the system that we want to optimize, which is often not available. In this article, we therefore automatically select the hyperparameters by optimizing the marginal likelihood Rasmussen and Williams [2006].

4.3.2 Acquisition Functions

A number of acquisition functions $\alpha(\cdot)$ have been proposed, such as probability of improvement Kushner [1964], expected improvement Mockus et al. [1978], upper confidence bound Cox and John [1997] and the recent entropy-based improvements Hennig and Schuler [2012]. All acquisition functions depend on the mean μ and the variance σ of the GP prediction and result in different trade-offs between exploration and exploitation. Experimental results Hennig and Schuler [2012] suggest that expected improvement on specific families of artificial functions performs better on average than probability of improvement and upper confidence bound. However, these results assumes a good prior knowledge of the objective functions (e.g., the correct covariance function to use). This assumption do not necessarily hold for real-world problems, such as gait optimization. Probability of improvement Lizotte et al. [2007], expected improvement Tesch et al. [2011] and upper confidence bound Calandra et al. [2014] have all been previously employed in gait optimization. However, little experimental comparison has been carried out in gait optimization Calandra et al. [2014c], and it is still unclear whether one of them should be preferred.

---

[3] A more informative prior can be used if expert knowledge is available.

Probability of Improvement (PI).

Introduced by Kushner Kushner [1964], the acquisition function PI is defined as

$$\alpha(\boldsymbol{\theta}) = \Phi\left(\frac{T-\mu(\boldsymbol{\theta})}{\sigma(\boldsymbol{\theta})}\right), \tag{4.7}$$

where $\Phi(\cdot)$ is the normal cumulative distribution function and T the target value. The target value T is often the minimum of all explored data plus, optionally, a positive constant (for a study of its effects, see Lizotte et al. [2012]). PI is a function bounded by the interval $[0, 1]$. Hence, since the normal cumulative distribution function is monotonically increasing, to minimize PI it is sufficient to minimize

$$\alpha(\boldsymbol{\theta}) = \big(T - \mu(\boldsymbol{\theta})\big)/\sigma(\boldsymbol{\theta}). \tag{4.8}$$

Intuitively, PI computes the probability (cumulative distribution) of the response surface in $\boldsymbol{\theta}$ to be better than the target value T.

Expected Improvement (EI).

Mockus Mockus et al. [1978] introduced EI, which can be seen as an extension of probability of improvement. EI is formulated as

$$\alpha(\boldsymbol{\theta}) = \sigma(\boldsymbol{\theta})[u\Phi(u) + \phi(u)]; \qquad u = \big(T - \mu(\boldsymbol{\theta})\big)/\sigma(\boldsymbol{\theta}), \tag{4.9}$$

where $\phi(\cdot)$ is the standard normal probability density function.

Upper Confidence Bound (UCB).

UCB Cox and John [1997] is defined as

$$\alpha(\boldsymbol{\theta}) = \mu(\boldsymbol{\theta}) - \kappa\sigma(\boldsymbol{\theta}). \tag{4.10}$$

The choice of the free parameter κ is crucial as it determines the trade-off rate between exploration and exploitation. A special case of UCB is GP-UCB Srinivas et al. [2010] where κ is automatically computed according to

$$\kappa = \sqrt{2\log\left(\frac{n^{d/2+2}\pi^2}{3\delta}\right)}, \tag{4.11}$$

where n is the number of past evaluations of the objective function f, $\delta \in (0, 1)$ is a parameters and d the dimensionality of the parameters $\boldsymbol{\theta}$. Automatically selecting κ using GP-UCB allows to estimate regret bounds Srinivas et al. [2010].

4.3.3 Optimizing the Acquisition Surface

Once the acquisition surface $\alpha(\boldsymbol{\theta})$ is computed, it is still necessary to find its minimum

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \alpha(\boldsymbol{\theta}). \tag{4.12}$$

This is still a global optimization problem, but considerably easier compared to the original global optimization problem defined in Equation (5.1):

- The measurements in Equation (4.12) are noise free since the objective function in Equation (4.10) is an analytical model. This allows to use also global optimization algorithms which do not consider stochasticity.
- There is no restriction in terms of how many evaluations of α we can perform: Evaluating the acquisition surface only requires interactions with the model, but not with a physical system, such as a robot. Hence, optimization methods that require thousands or millions of evaluations can be employed to find the global minimum.
- We can compute the gradients of any order, either with finite differences or analytically and, therefore, use first or second-order optimization methods.

Therefore, to find the minimum $\boldsymbol{\theta}^*$ of α virtually any global optimizer can be used. Common choices are DIRECT Jones et al. [1993] to find an approximate global minimum followed by L-BFGS Byrd et al. [1995] or CMA-ES Hansen and Ostermeier [2001] to refine it. In our experiments, we use DIRECT and L-BFGS.

Cost incurred by the analytical solution	-5.57 ± 0.01
Cost incurred by Bayesian optimization	-5.54 ± 0.01

Table 4.2.: Performance of Bayesian optimization compared to the exact solution for the stochastic LQR problem.

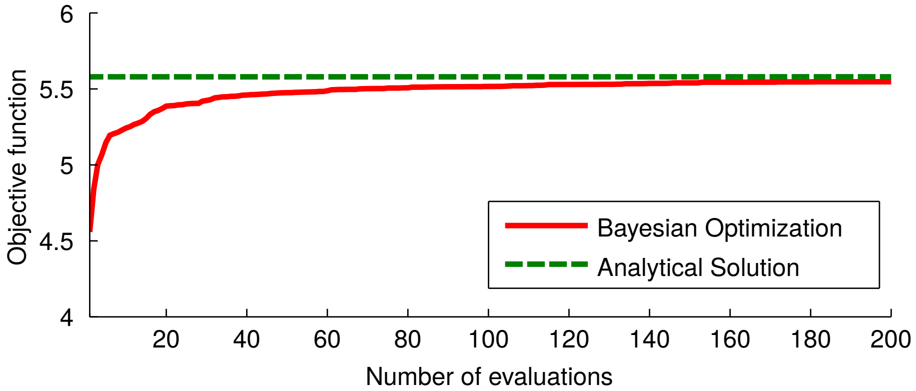


Figure 4.4.: Average over 50 experiments of best parameters found during the minimization process for a stochastic LQR using Bayesian optimization. The average objective value function (red curve) during the optimization process and the average analytical solution (green dashed line) are shown.

4.4 Evaluation and Comparisons

In this section, we experimentally compare Bayesian optimization with different acquisition functions and other baseline optimization methods. At first, we perform a feasibility study on a simulated stochastic linear-quadratic regulator. This system is a classical stochastic optimal control problem for which we can compute an optimal solution against which to compare. Following, we perform an experimental comparison on a real bipedal robot. On this bipedal robot we find gait parameters that maximize the walking speed in real and noisy conditions.

4.4.1 Evaluation on a Stochastic Linear-Quadratic Regulation Task

The linear-quadratic regulator is a classical stochastic optimal control problem. The discrete-time stochastic LQR problem consists of a linear dynamical system

$$\boldsymbol{x}_{t+1} = \boldsymbol{A}_t \boldsymbol{x}_t + \boldsymbol{B}_t \boldsymbol{u}_t + \boldsymbol{w}_t, \qquad t = 0, 1, ..., N-1, \tag{4.13}$$

and a quadratic cost

$$J = \boldsymbol{x}_N^T \boldsymbol{Q}_N \boldsymbol{x}_N + \sum\nolimits_{t=0}^{N-1} \left(\boldsymbol{x}_t^T \boldsymbol{Q}_t \boldsymbol{x}_t + \boldsymbol{u}_t^T \boldsymbol{R}_t \boldsymbol{u}_t \right), \tag{4.14}$$

where $\boldsymbol{w}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma})$ is Gaussian system noise and the matrices $\boldsymbol{R}_t > \boldsymbol{0}$, $\boldsymbol{Q}_t \geq \boldsymbol{0}$, $\boldsymbol{A}_t$, $\boldsymbol{B}_t$ are given and assumed to be time invariant. The objective is to find the control sequence $\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{N-1}$ that minimize Equation (4.14). The control signal $\boldsymbol{u}_t$ is a linear function of the state $\boldsymbol{x}_t$, computed for each time step as

$$\boldsymbol{u}_t = \boldsymbol{L}_t \boldsymbol{x}_t,$$

where $\boldsymbol{L}_t$ is a gain matrix. An analytical solution exists to compute the optimal gain matrix $\boldsymbol{L}_t$ such that the quadratic cost J for the stochastic linear-quadratic regulator is minimized Bertsekas [2007].

To assess the performance of Bayesian optimization, we consider a stochastic LQR system with $\boldsymbol{x} \in \mathbb{R}^2$, $\boldsymbol{u} \in \mathbb{R}^4$. The stationary gain matrix $\boldsymbol{L} \in \mathbb{R}^{4\times 2}$ defines a set of 8 free parameters to be determined by Bayesian optimization. We compare our solution with the corresponding analytical solution for the stationary gain matrix $\boldsymbol{L}$. For Bayesian optimization, we define the objective function as

$$f(\boldsymbol{\theta}) = \log(J/N), \tag{4.15}$$

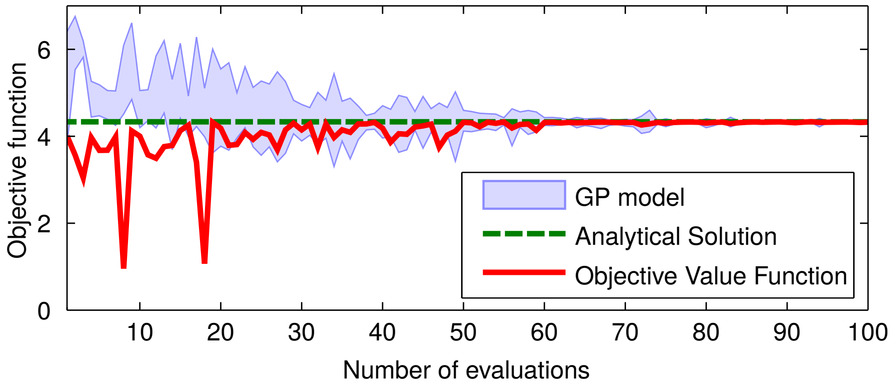


Figure 4.5.: Example of Bayesian optimization for a stochastic LQR. The objective value function (red curve) and the 95% confidence of the model prediction (blue area) are shown during the optimization process, additionally, the analytical solution (green dashed line) is shown as a reference.

where the parameters $\boldsymbol{\theta}$ to optimize are the stationary gain matrix $\boldsymbol{L} \in \mathbb{R}^{4\times 2}$. To initialize Bayesian optimization, 15 uniformly randomly sampled gain matrices $\boldsymbol{L}$ were used. Moreover, the initial state $\boldsymbol{x}_0$ was sampled from a standard normal $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

We performed 50 independent experiments: For each experiment, we selected the best parameters found after 200 steps of Bayesian optimization. These parameters were then evaluated on the stochastic LQR system 100 times. Table 4.2 shows the mean value for the objective function and its standard deviation for both the analytical solutions and the ones obtained through Bayesian optimization. We conclude that Bayesian optimization finds near-optimal solutions for the stochastic LQR problem. Additionally, as shown in Figure 4.4, the average over the 50 experiments of the best parameters found so far in the optimization process suggests that Bayesian optimization reliably quickly finds a near-optimal solution.

Figure 4.5 shows an example of the minimization process of Bayesian optimization for the stochastic LQR problem. The objective function is displayed as a function of the number of evaluations. Each evaluation requires to compute the objective function f in Equation (4.15) for the current parameters $\boldsymbol{\theta} = \boldsymbol{L}$. The analytical minimum is shown by the green dashed line, the shaded area shows the 95% confidence bound of the predicted objective function $p(f(\boldsymbol{\theta}))$ for the parameters selected in the ith evaluation. The red line shows the actual measured function value $f(\boldsymbol{\theta})$. Initially, the model was relatively uncertain. With an increasing number of experiments the model became more certain, and the optimization process converged to the optimal solution.

We conclude that Bayesian optimization can efficiently find gain matrices $\boldsymbol{L}$ that solve the stochastic LQR problem. Additionally, with Bayesian optimization it is possible to find stationary solutions for cases with a short time horizon N where no analytical optimal solution is available: The algebraic Riccati equation is not applicable for finite time horizons N, and the discrete time Riccati equation, which can be applied, does not produce a stationary solution.

4.4.2 Gait Optimization of a Bio-Inspired Biped

To validate our Bayesian gait optimization approach, we used the dynamic bipedal walker *Fox*, shown in Figure 4.1. The walker is mounted on a boom that enforces planar, circular motion. This robot consists of a trunk, two legs made of rigid segments connected by knee joints to telescopic leg springs, and two spheric feet with touch sensors Renjewski [2012]. *Fox* is equipped with low-cost metal-gear DC motors at both hip and knee joints. Together they drive four actuated degrees of freedom. Moreover, there are six sensors on the robot: two on the hip joints, two on the knee joints, and one under each foot. The sensors on the hip and knee joints return voltage measurements corresponding to angular positions of the leg segments, as shown in Figure 4.7. The touch sensors return binary ground contact signals. An additional sensor in the boom measures the angular position of the walker, i.e., the position of the walker on the circle.

The controller of the walker is a **finite state machine (FSM)**, shown in Figure 4.6, with four states: two for the swing phases of each leg Renjewski and Seyfarth [2012]. These states control the actions performed by each of the four actuators, which were extension, flexion or holding of the joint. The transitions between the states are regulated by thresholds based on the angles of the joints.

For the optimization process, we identified eight parameters of the controller that are crucial for the resulting gait. These gait parameters consist of four threshold values of the FSM (two for each leg) and the four control signals applied during extension and flexion (separately for knees and hips). It is important to notice that a set of parameters that proved to be efficient with a set of motors could be ineffective with a different set of motors (e.g., if one or more motors is replaced), due to slightly different mechanical properties. Therefore, automatic and fast gait optimization techniques are essential for this robot.

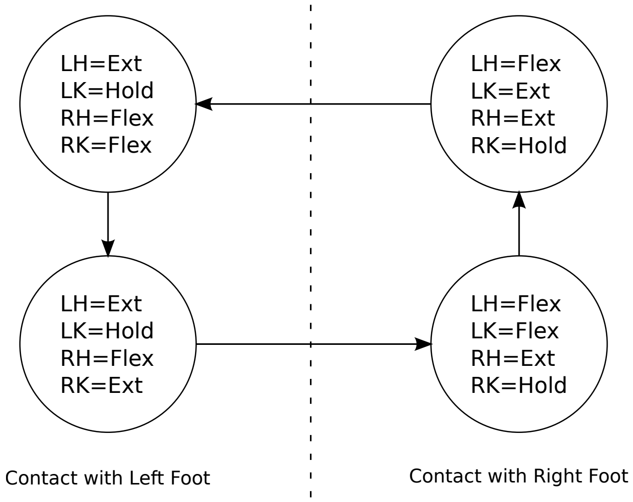


Figure 4.6.: The *Fox* controller is a finite state machine with four states. Each of the four joints, left hip (LH), left knee (LK), right hip (RH) and right knee (RK), can perform one of three actions: flexion (Flex), extension (Ext) or holding (Hold). When a joint reaches the maximum extension or flexion, its state is changed to holding. The transition between the states and the control signals applied during flexion and extension are determined by the controller parameters $\boldsymbol{\theta}$.

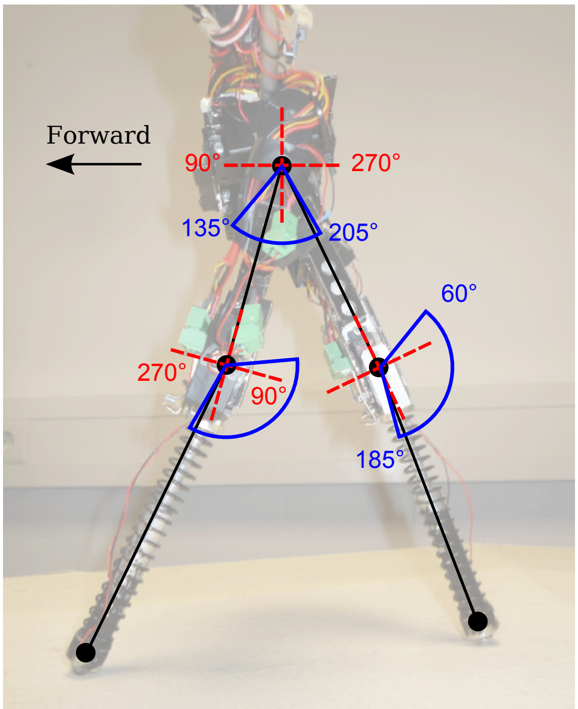


Figure 4.7.: Hip and knee angle reference frames (red dashed) and rotation bounds (blue solid). The hip joint angles' range lies between 135° forward and 205° backward. The knee angles range from 185° when fully extended to 60° when flexed backward.

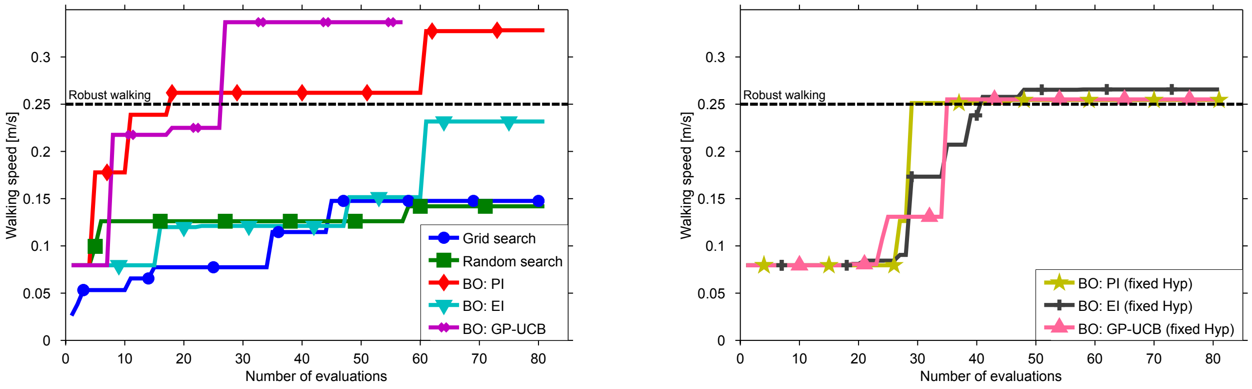
Comparison of Acquisition Functions on a Gait Optimization Task

In the first experiment we compare the performances of Uniform Random Search, Grid Search and Bayesian optimization with the three different acquisition functions introduced in Section 4.3.2. For Bayesian optimization, we also consider the two separate cases when the hyperparameters are automatically optimized, and when they are manually set by an expert. In this experiment, we optimize the four threshold parameters of the controller $\boldsymbol{\theta}$ with respect to the objective function

$$f(\boldsymbol{\theta}) = -\frac{1}{N}\sum\nolimits_{i=1}^{N} V_i(\boldsymbol{\theta}), \tag{4.16}$$

i.e., the negative average walking velocities V_i over $N = 3$ experiments. Minimizing this performance criterion maximizes the walking distance in the given time horizon. Moreover, this criterion does not only guarantee a fast walking gait but also reliability, since the gait must be robust to noise and the initial configurations across multiple experiments. Each experiment was initialized from similar initial configurations and lasted 12 seconds, starting from the moment when the foot of the robot initially touched the ground. To initialize Bayesian optimization, three uniformly randomly sampled parameter sets were used.

The maximum walking speed of *Fox* evaluated during the gait optimization process for the different methods is shown in Figure 4.8a. The optimization process of GP-UCB is limited to 57 evaluations due to a mechanical failure that forcefully interrupted the experiment. Values of the objective function below 0.1 m/s indicate that the robot fell down after a single step. Values between 0.1 m/s and 0.15 m/s indicate that the robot could walk for multiple steps but showed systematic falls thereafter. Between 0.15 m/s and 0.25 m/s only occasional falls occurred. Above 0.25 m/s the achieved gait was robust and did not result in a fall. From the results we see that both grid search and random search performed poorly, finding a maximum, such that the robot could only barely walk, i.e., limp. We noticed that Bayesian optimization performed considerably better with all acquisition functions. Bayesian optimization using PI and GP-UCB achieved robust gaits with similar walking speed, while GP-UCB was slightly faster in finding the best gait parameter settings. On the other hand, Bayesian optimization using EI did not lead to robust gaits. The reason of this result were the inaccuracies of the model of the underlying objective function: The automatically selected hyperparameters had overly long length-scales (see Equation (4.4)), which resulted in an inappropriate model and, thus, evaluating parameters of little interest.



(a) **(b)**

Figure 4.8.: The maximum average walking speed of *Fox* evaluated during the gait optimization process. (a) Bayesian optimization performed better than both grid and random search. BO with the GP-UCB acquisition function performed best, achieving a fast and robust gait in less than 30 experiments. (b) Bayesian optimization of various acquisition functions are shown for fixed hyperparameters. Imprecise fixed hyperparameters led to sub-optimal solutions for all the acquisition functions.

This result is unexpected as EI is considered a versatile acquisition function, and there are experimental results Hennig and Schuler [2012], which suggest that EI on specific families of artificial functions performs better than GP-UCB and PI. We speculate that these results do not necessarily apply to complex real-world objective functions, such as the one we optimized. In particular, during the empirical evaluation, we observed that EI behaved excessively greedily, exploring the parameters space insufficiently. In turn, this insufficient exploration resulted in overly long length-scales and an inappropriate GP model. Therefore, we hypothesize that EI might not perform well when the model is not sufficiently accurate.

As a second comparison, we studied the effects of manually fixing hyperparameters. Thereby, we manually fixed the hyperparameters of the GP models to reasonable values, based on our expert knowledge. Figure 4.8b shows the performance of BO with the different acquisition functions with fixed hyperparameters. It can be observed that all acquisition functions found similar sub-optimal solutions with fixed hyperparameters since one parameter reached only a sub-optimal value. This observation suggests that, at least for that one parameter, the wrong length-scales prevented the creation of an accurate model and, therefore, the optimization process was hindered. As a confirmation of this hypothesis, using all the evaluations performed, we trained a GP model and automatically selected the hyperparameters using the marginal likelihood. The resulting values of the hyperparameters were, in some cases, half of the manually selected values, therefore, suggesting that the chosen length-scales were a rough approximation of the real ones.

Both GP-UCB and PI using fixed hyperparameters performed worse than the respective cases with automatic hyperparameters selection because the longer length-scales limited the exploration of these acquisition functions. In contrast, for EI the use of fixed hyperparameters was beneficial. In fact, the fixed length-scales were smaller than the automatically selected ones and, therefore, more exploration was performed. The hyperparameters of the GP model directly influence the amount of exploration performed by the acquisition functions. Hence, fixing the hyperparameters using expert knowledge can be an attractive choice, since forcing the right amount of exploration can speed up the optimization process. However, the presented experimental results also show that a poor choice of hyperparameters can potentially harm the optimization process by limiting the exploration and leading to sub-optimal solutions.

In Table 4.9, we present the comparison of the optimum found by all the evaluated methods. During the experiments, we observed and empirically estimated that variations of up to 0.04 m/s can depend on the presence of noise in the experiments. Additionally, it should be noticed that the real noise of the objective function is not Gaussian, nor homoscedastic. In fact, configurations of the parameters that produce periodic falls after a single step or stable gaits, typically behave consistently across various experiments and, therefore, result in a smaller noise (0.01 m/s). Instead, for parameters that produce unstable gaits with occasional falls the noise can be larger, typically up to 0.04 m/s. A two-dimensional slice of the response surface computed using the data collected from all the over 1800 evaluations is shown in Figure 4.10. This optimization surface is complex and non-convex, and, therefore, unsuitable for local optimization methods (e.g., gradient-based methods). In Figure 4.11, similar results are shown by taking slices of the parameter space. Additionally, the symmetry between the parameters of the two legs is visible, except for a shift of about 5 degrees due to the circular walking trajectory.

The GP modeling capabilities are often overlooked when evaluating Bayesian optimization's performances, with all the emphasis on the use of different acquisition functions. Based on the results of our experimental evaluation, we speculate

Method	Maximum during optimization
Grid search	0.148
Pure random search	0.142
BO: PI	0.328
BO: EI	0.232
BO: GP-UCB	**0.337**
BO: PI (fixed Hyp)	0.254
BO: EI (fixed Hyp)	0.266
BO: GP-UCB (fixed Hyp)	0.255

Figure 4.9.: Maximum average walking speeds [m/s] found by the different optimization methods. BO using GP-UCB with automatic hyperparameter selection found the best maximum of all methods. The maximum obtained by PI is qualitatively similar.

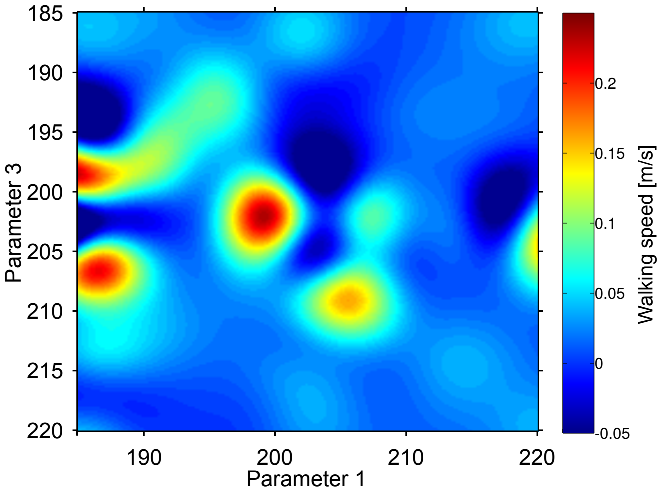


Figure 4.10.: Two-dimensional slice of the mean of the response surface computed using all the experiments performed. Avoiding the local maxima demands global optimization.

that for complex objective functions there exists a strict and yet unexplored connection between the exploration properties of the acquisition function and the capabilities of GP modeling. The performance of an acquisition function depends on the capabilities of properly modeling the function, and vice versa a proper modeling takes place only when the acquisition function evaluates relevant parameters.

Including the Control Signals in the Gait Optimization

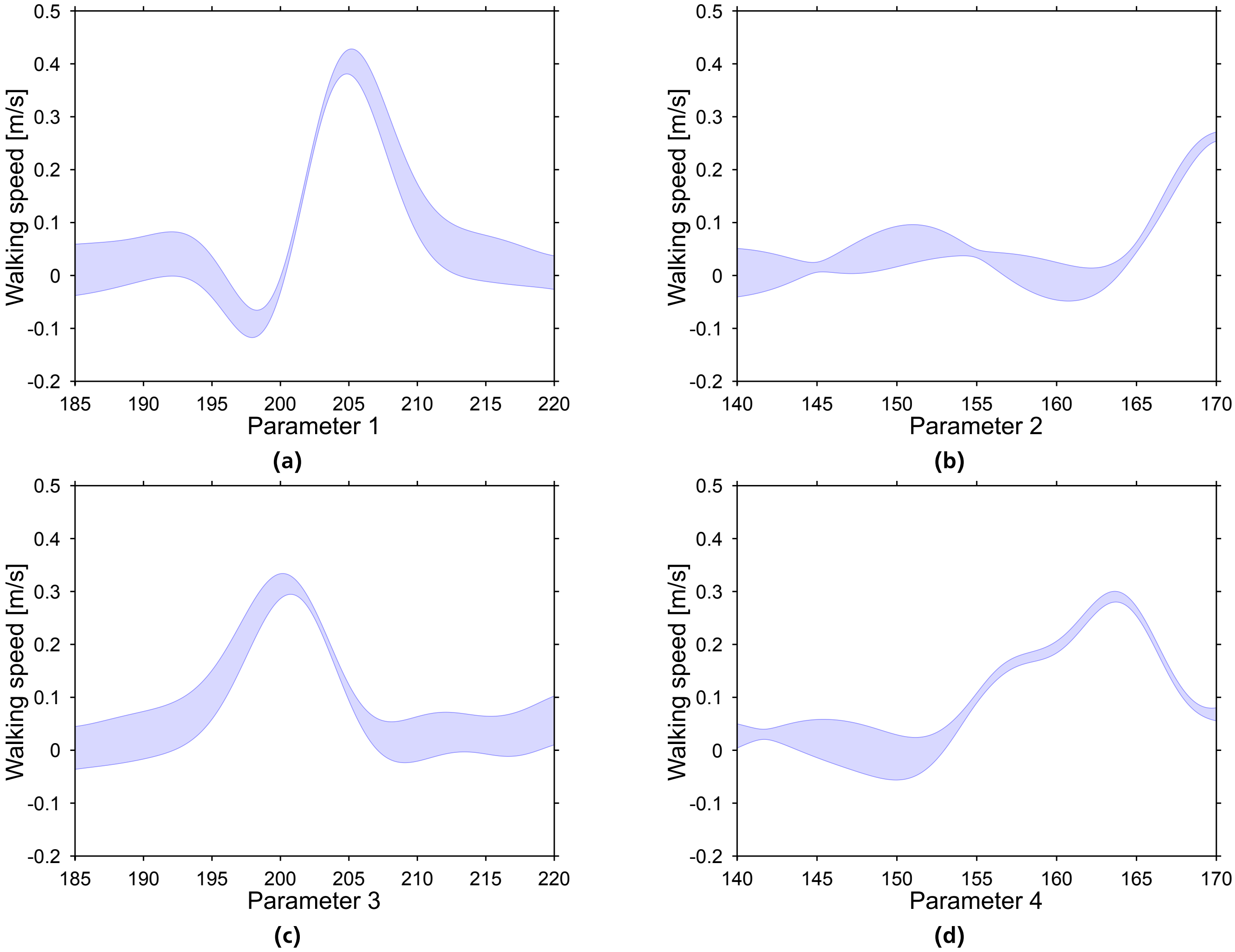
We further evaluated the performances of Bayesian optimization with GP-UCB (the best performing acquisition function in the previous experiment), by increasing the dimensionality of the optimization problem. To the four parameters already selected (the thresholds), we add the other four parameters corresponding to the voltage provided to the motors, for a total of eight parameters. As objective function we again use the velocity defined in Equation (4.16).

In Figure 4.12, the Bayesian optimization process for gait learning is shown. Initially, the learned GP model could not adequately capture the underlying objective function. Average velocities below 0.1 m/s typically indicate a fall of the robot after the first step. Large parts of the first 60 experiments were spent to learn that the control signals applied at the hips had to be sufficiently high in order to swing the leg forward (i.e., against gravity and friction). Once this knowledge was acquired, the produced gaits typically led to walking but were rather unstable, and the robot fell after few steps. After 80 experiments, the model became more accurate (the function evaluations shown in red lied within the 95% confidence bound of the prediction), and Bayesian optimization found a stable walking gait. The resulting gait[4] was evaluated for a longer period of time, and it proved sufficiently robust to walk continuously for 2 minutes without falling, while achieving an average velocity of 0.45 m/s This average velocity was close to the maximum velocity this hardware set-up can achieve Renjewski [2012]. Notably, the parameters obtained through Bayesian optimization that correspond to the values of the thresholds were slightly asymmetrical for the two legs. We explain the superior performance of asymmetrical parameters by the smaller radius of the walking circle for the inner leg.

From our experience with the biped *Fox*, hand-tuning the gait parameters can be a very time-consuming process, taking up to multiple days. Only around 1% of the considered parameter space leads to walking gaits, and the influence and the interaction of the parameters is not trivial. In our experience, expert manual parameter search typically yielded inferior gaits compared to the ones obtained by Bayesian optimization, in both walking velocity and robustness. Automatic gait parameter selection by means of Bayesian optimization sped up the parameter search from days to hours.

---

[4] Videos are available at `http://www.ias.tu-darmstadt.de/Research/Fox`.



(a) (b) (c) (d)

Figure 4.11.: Example of slices of the response surface along each of the parameters. The response surface was trained with the data collected across all the experiments. Only a small zone of each slice leads to a walking gait. The symmetry between parameters 1 and 3 and between parameters 2 and 4 is clear. A shift of about 5 degrees in the optimal parameters is visible due to the constrained circular walk.

4.5 Summary

Automatic gait optimization is a key challenge that needs to be addressed to deploy bipedal walkers in real-world applications. The principal limitation of most common optimization methods is the need to be experimentally-efficient while accounting for various sources of uncertainty in each experiment, including measurement noise, model uncertainty and stochasticity of the robot and environment. In this chapter, we evaluated Bayesian optimization on a bio-inspired bipedal walker. Due to a probabilistic surrogate model, Bayesian optimization is an efficient solution to real-world gait optimization tasks. Performing over 1800 experiments, we have compared different variants of Bayesian optimization and observed that the GP-UCB acquisition function performed best.

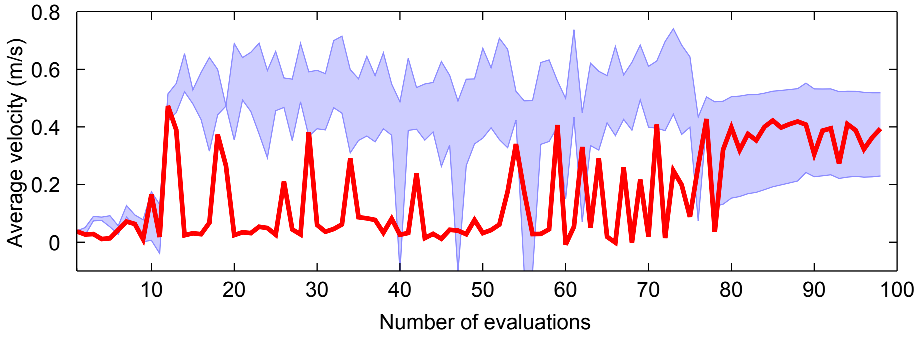


Figure 4.12.: Average walking speed during the gait optimization process of *Fox* using Bayesian optimization. For each evaluation the measurement of the objective function $f(\boldsymbol{\theta}^*)$ (red curve) and the corresponding 95% confidence of the model prediction $\hat{f}(\boldsymbol{\theta}^*)$ (blue area) are shown. Three evaluations are used to initialize Bayesian optimization and are not shown in the plot. After 80 evaluations, Bayesian optimization finds an optimum corresponding to a stable walking gait with an average speed of 0.45 m/s.

5 Robust Multi-Objective Bayesian Optimization

> The mystery of human existence lies not in just staying alive, but in finding something to live for
>
> Fyodor Dostoyevsky

In the previous chapter, we discussed the use of Bayesian optimization for the optimization of control parameters w.r.t. a single objective. However, in many optimization tasks (including in robotics) there exists multiple contrasting criteria. In this chapter, we study the use of Bayesian optimization in multi-objective optimization tasks. Moreover, we demonstrate that by exploiting Bayesian modeling it is possible to introduce robustness indexes that can be used to find parameters resilient to stochasticity and perturbations.

5.1 Introduction

Bayesian optimization (BO) is a global black-box optimization approach [Jones, 2001, Kushner, 1964], which has been successfully applied to expensive optimization, e.g., in automatic tuning of machine learning algorithms [Snoek et al., 2012b, Thornton et al., 2013] and in robotics [Calandra et al., 2015d, Lizotte et al., 2007, Tesch et al., 2011]. The goal of BO is to optimize expensive objective functions by reducing the number of evaluations on the real system. BO belongs to the class of model-based optimization methods meaning that it creates a response surface that models the direct mapping from parameters to objective values. Subsequently, this response surface is used to select the next parameters to evaluate.

Classic BO considers a single-objective optimization (SOO) task, i.e., a single objective function has to be optimized. However, many interesting optimization tasks possess multiple desiderata that need to be simultaneously optimized[1]. Examples can be found in machine learning [Jin, 2006], robotics [Capi et al., 2005, Oliveira et al., 2013, Tesch et al., 2013], finance [Luenberger, 1998, Tapia and Coello, 2007] and engineering [Marler and Arora, 2004, Tsoukalas and Makropoulos, 2015]. These objectives are often in conflict with each other, and increasing the performance in one objective may decrease the performance of another objective. An example is the design and operation of complex robotic systems, where we have to consider multiple criteria, e.g., motion accuracy, speed, robustness to noise or energy-efficiency. This class of optimization problems takes the name of multi-objective optimization (MOO).

The goal of MOO is to find a performance trade-off between the multiple objectives which is considered satisfying, typically by the human designer [Collette and Siarry, 2004, Deb, 2001]. Although, various multi-objective Bayesian optimization (MOBO) methods exist in the literature [Emmerich et al., 2008, Knowles, 2006, Ponweiser et al., 2008], they do not address the real-world problem of robustness to disturbance. Implicitly or explicitly, in most real-world applications one of the desired design objectives is the reliability and robustness of the performance of the system as discussed by Miettinen et al. [2008]. Choosing robust solutions help to obtain consistent performance even in presence of noise and disturbances. The topic of robustness is relevant in many communities, and applications can be found in finance [Best and Grauer, 1991], engineering [Bertsimas et al., 2011], machine learning [Caramanis et al., 2012], optimal control and robotics [Kuindersma et al., 2013]. However, past approaches to measure robustness in MOO problems [Gaspar-Cunha and Covas, 2008] require a large number of evaluations of the objective functions, which is infeasible for problems where it is expensive to evaluate the objective functions, either due to time constraints (e.g., biological experiments [?], simulations [Svenson and Santner, 2016], training of deep learning models [Snoek et al., 2012b]), or hardware constraints (e.g., robotic experiments [Calandra et al., 2015d] or design of FPGAs [Kurek et al., 2016]).

The goal of this chapter is to develop a framework for robustness and reliability in multi-objective optimization of functions that are expensive to evaluate. We develop this comprehensive framework by taking inspiration from multiple communities and disciplines: Bayesian optimization [Jones, 2001], multi-objective optimization [Branke et al., 2008, Deb, 2001] and robust optimization [Ben-Tal et al., 2009, Bertsimas et al., 2011], as illustrated in Figure 5.1. On the one hand, we demonstrate that the use of Bayesian modeling is largely beneficial to the task of estimating robustness and improve the quality of the solutions obtained even from a small number of experiments. On the other hand, we introduce multiple metrics of robustness within the multi-objective Bayesian optimization framework so that they can be used as additional objective during the optimization. In particular, we consider and address two sources of disturbance

---

[1] There is some evidence that humans optimize multiple objectives simultaneously [Wickelgren, 1977].

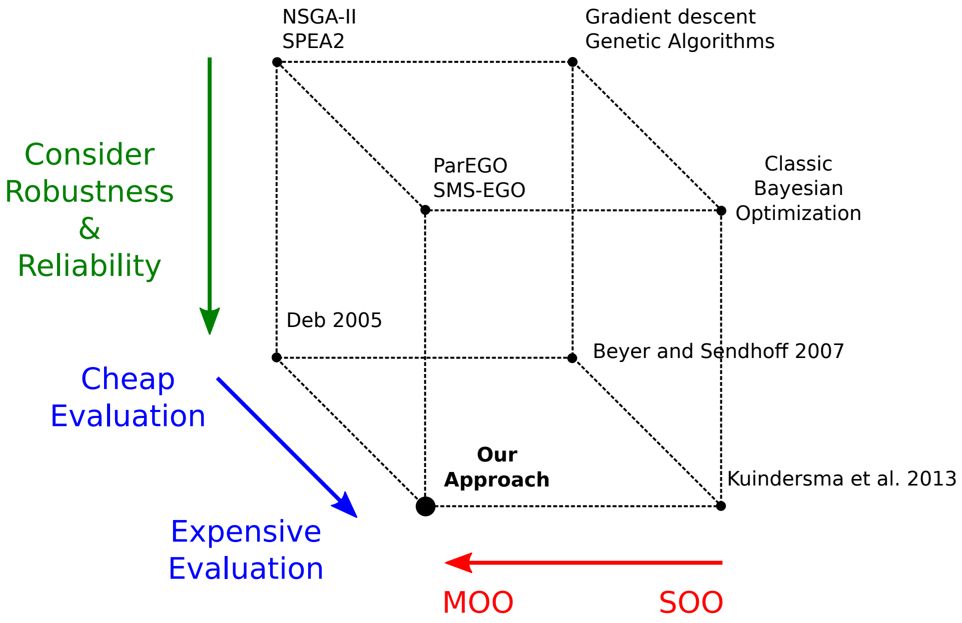


Figure 5.1.: This article fills a literature gap by providing a framework for robustness and reliability in multi-objective optimization of functions that are expensive to evaluate. For this purpose, we relate to multiple related fields. Classic MOO methods, require thousands of evaluations and do not consider robustness and reliability [Deb et al., 2002b, Zitzler et al., 2001]. Extensions to include robustness require even more evaluations [Singh et al., 2003]. Classic BO consider SOO of expensive evaluations [Jones, 2001]. MOBO such as ParEGO [Knowles, 2006] and EIHV [Emmerich et al., 2008] extended BO to the MOO case, but do not consider robustness. Kuindersma et al. [2013] extended BO to include robustness to measurement noise, but does not consider the presence of multiple objectives.

that require the introduction of robustness: 1) input-dependent stochasticity in the objective function 2) uncertainty in the parameters to be employed.

The remainder of this chapter is structured as follow. In Section 5.2, we formalize Bayesian optimization and the multi-objective optimization framework, and extensively discuss related work. In Section 5.3, we formalize multiple robustness measures within our framework and detail our approach to estimate them. In Section 5.4, we experimentally evaluate our proposed approach. In Section 5.5, we discuss the properties and perspectives of our approach. We conclude with final remarks and future work in Section 5.6.

5.2 Background & Related Work

In this paper we build on previous work in BO, MOO and robust optimization. Therefore, in this section, we present a brief introduction to these concepts and to the related literature. We start by formalizing single objective Bayesian optimization and Gaussian processes. Following, we discuss MOO and its two most common approaches (scalarization and Pareto optimization), the metrics used to evaluate MOO, and the challenges of MOO. To conclude, we thoroughly discuss related work from multiple communities and detail two related MOBO algorithms: ParEGO and SMS-EGO.

5.2.1 Single-objective Bayesian Optimization

Global unconstrained single-objective optimization can be formulated as finding the **parameters** $\boldsymbol{\theta}^* \in \mathbb{R}^d$ that minimize[2] an **objective function** $f : \mathbb{R}^d \rightarrow \mathbb{R}$, i.e.,

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^d} f(\boldsymbol{\theta}) \,. \tag{5.1}$$

We consider the set-up where the optimization problem is non-convex and the objective function is unknown. Typically, the objective function is also assumed to be of zero-order, i.e., no gradient $\mathrm{d}f(\boldsymbol{\theta})/\mathrm{d}\boldsymbol{\theta}$ is measured.

Bayesian optimization (BO) is an iterative optimization method [Jones, 2001, Kushner, 1964, Mockus et al., 1978, Osborne et al., 2009], also known as Efficient Global Optimization (EGO) [Jones et al., 1998] or Kriging [Delhomme,

---

[2] To maximize a function it is sufficient to invert the sign of the objective function as $f'(\cdot) = -f(\cdot)$

Algorithm 2: Single-objective Bayesian optimization

```
$\mathscr{D}$ ⟵ if available: $\{\boldsymbol{\theta}, f(\boldsymbol{\theta})\}$
Prior ⟵ if available: Prior of the response surface
repeat
    Train response surface $\hat{f}$ from $\mathscr{D}$
    Find $\boldsymbol{\theta}^*$ maximizing the acquisition surface $\alpha(\boldsymbol{\theta})$
    Evaluate $\boldsymbol{\theta}^*$ on the real system
    Add $\{\boldsymbol{\theta}^*, f(\boldsymbol{\theta}^*)\}$ to $\mathscr{D}$
until stopCriteria
```

1978] and Sequential Parameter Optimization (SPO) in the evolutionary computation community [Bartz-Beielstein et al., 2005]. Similar approaches can also be found in active learning [Cohn et al., 1996] and experimental design [Atkinson, 1996].

At each iteration, BO creates a response surface using the past parameters evaluated and the corresponding objective values measured $\mathscr{D} = \{\boldsymbol{\theta}, f(\boldsymbol{\theta})\}$. In order to choose the next point to evaluate, BO maps the response surface trough an acquisition function returning a metric of the goodness of each set of parameters, therefore producing the acquisition surface. Subsequently, BO optimizes this acquisition surface

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}\in\mathbb{R}^d} \alpha\left(\boldsymbol{\theta} \,\middle|\, \hat{f}\right). \tag{5.2}$$

Finally, we can evaluate the resulting set of parameters $\boldsymbol{\theta}^*$ on the real system, measure the corresponding objective function and store them. The process is repeated until convergence or the desired stop criteria are verified. An overview of the BO algorithm is shown in Algorithm 2.

Gaussian processes (discussed in Section 8) are commonly used for modeling the response surface. Alternative response surface models include random trees [Hutter et al., 2011], polynomials [Jones, 2001] and neural networks [Anjum et al., 1997, Snoek et al., 2015]. For a more complete overview of Bayesian optimization, we refer to Jones [2001], Brochu et al. [2010] and Shahriari et al. [2016].

Acquisition Function

The purpose of the acquisition function is to select, given the response surface, the next sets of parameters to be evaluated. Typically, these parameters are selected by the acquisition function in order to balance exploitation of previously evaluated parameters and exploration of unknown parameters. As such, if the exploration is sufficiently high, the optimization is able to avoid local minima.

A number of acquisition functions $\alpha(\cdot)$ have been proposed in the literature. The most well-known are probability of improvement [Kushner, 1964], expected improvement [Mockus et al., 1978], upper confidence bound [Cox and John, 1997, Srinivas et al., 2010], entropy-based improvements [Hennig and Schuler, 2012] and predictive entropy search [Hernández-Lobato et al., 2014]. Recently, Wang et al. [2016] proposed a unifying framework for some of these acquisition functions. Meta-acquisition functions that make use of multiple acquisition functions have also been proposed [Hoffman et al., 2011]. The use of different acquisition functions results in different trade-offs between exploration and exploitation, with some acquisition functions aggressively exploring the parameter space and others conservatively evaluating in the vicinity of past evaluations. What is the best trade-off between exploration and exploitation, and how to achieve it is an open research question.

Gaussian Processes

Gaussian Processes (GP) are a state-of-the-art regression method [Rasmussen and Williams, 2006] often used in Bayesian optimization. Standard GP aim at modeling a function having the form

$$y = \hat{f}(x) + \epsilon_f, \quad \epsilon_f \sim \mathcal{N}(0, \Sigma). \tag{5.3}$$

In the context of this paper, a GP is a distribution over response surfaces

$$\hat{f} \sim \mathcal{GP}\left(m_f, k_f\right), \tag{5.4}$$

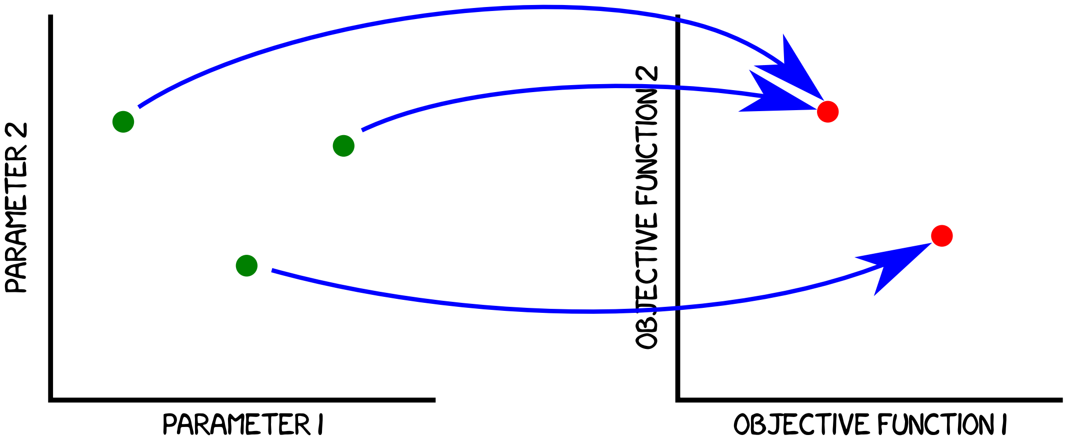


Figure 5.2.: Sketch of a multi-objective optimization task. The objective functions $f_i(\cdot)$ define (in absence of noise) a non-injective, non-surjective mapping, represented by the blue arrows, from the parameter space *(left)* to the objective function space *(right)*.

fully defined by a prior mean function m_f and a covariance function k_f. In this article, we consider a prior mean $m_f \equiv 0$ and a covariance function

$$k(\boldsymbol{x}_p,\boldsymbol{x}_q)=\sigma_f^2\exp\left(-\tfrac{1}{2}(\boldsymbol{x}_p-\boldsymbol{x}_q)^T\boldsymbol{\Lambda}^{-1}(\boldsymbol{x}_p-\boldsymbol{x}_q)\right)+\sigma_w^2\delta_{pq} \tag{5.5}$$

where $\boldsymbol{\Lambda}=\operatorname{diag}\left([l_1^2,\ldots,l_d^2]\right)$ and δ_{pq} is the Kronecker delta (which is one if $p=q$ and zero otherwise). Here, l_i are the characteristic length-scales, σ_f^2 is the variance of the latent function $f(\cdot)$ and σ_w^2 the noise variance. The purpose of $\sigma_w^2\delta_{pq}$ is to identify and explicitly incorporate the presence of input-independent constant (i.e., homoscedastic) Gaussian noise ϵ_f. Given n training inputs $\boldsymbol{\theta}=[\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_n]$ and corresponding training targets $\boldsymbol{y}=[y_1,\ldots,y_n]$, we define the training data set $\mathscr{D}=\{\boldsymbol{\theta},\boldsymbol{y}\}$. Training the GP corresponds to finding good hyperparameters $\boldsymbol{\omega}=[l_1,\ldots,l_d,\sigma_f,\sigma_w]$, which can be done by the standard procedure of maximizing the marginal likelihood [Rasmussen and Williams, 2006]. Alternatively, the hyperparameters can be carefully marginalized out trough numerical techniques, such as MCMC [Rasmussen and Williams, 2006].

The GP yields the predictive distribution over rewards for a new input $\boldsymbol{\theta}^*$

$$p\,(y|\mathscr{D},\boldsymbol{\theta}^*)=\mathcal{N}\left(\mu(\boldsymbol{\theta}^*),\sigma^2(\boldsymbol{\theta}^*)\right)\,, \tag{5.6}$$

where the mean $\mu(\boldsymbol{\theta}^*)$ and the variance $\sigma^2(\boldsymbol{\theta}^*)$ are

$$\mu(\boldsymbol{\theta}^*)=\boldsymbol{k}_*^T\boldsymbol{K}^{-1}\boldsymbol{y}\,, \tag{5.7}$$

$$\sigma^2(\boldsymbol{\theta}^*)=k_{**}-\boldsymbol{k}_*^T\boldsymbol{K}^{-1}\boldsymbol{k}_*\,. \tag{5.8}$$

The entries of the matrix $\boldsymbol{K}$ are $K_{ij}=k\left(\boldsymbol{\theta}_i,\boldsymbol{\theta}_j\right)$, and we define $k_{**}=k(\boldsymbol{\theta}^*,\boldsymbol{\theta}^*)$ and $\boldsymbol{k}_*=k(\boldsymbol{\theta},\boldsymbol{\theta}^*)$.

5.2.2 Multi-objective Optimization

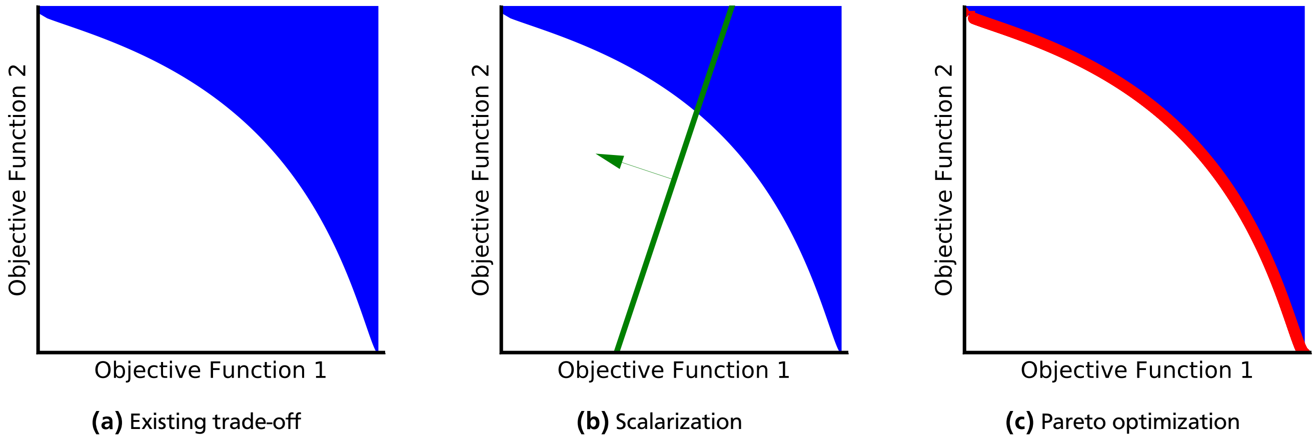
The optimization of different objectives can be formalized as a Multi-Objective Optimization (MOO) problem [Branke et al., 2008]. Multi-objective optimization generalizes the single-objective optimization problem in Equation (5.1) by optimizing a set of objective functions $f_i(\cdot)$, $i=1\ldots n$, simultaneously, such that

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^d}{\arg\min}\,\{f_1(\boldsymbol{\theta})\ldots f_n(\boldsymbol{\theta})\}\,, \tag{5.9}$$

where $\boldsymbol{\theta}$ are the optimization parameters, shared across the multiple objective functions $f_i(\cdot)$. The objective function space is now n-dimensional, and evaluating the objective function returns the vector $\boldsymbol{y}\in\mathbb{R}^n$. A visualization of this mapping is shown in Figure 5.2.

Scalarization vs Pareto optimization

Two main approaches are common in MOO trade-off between different objective functions: scalarization and Pareto optimization. The main difference between the two approaches lay in the way that the human designer specifies the desired trade-off: a priori in scalarization, a posteriori in Pareto optimization.



(a) Existing trade-off **(b)** Scalarization **(c)** Pareto optimization

Figure 5.3.: Scalarization vs Pareto optimization: (a) The trade-off curve is used to visualize the performance of a set of parameters with regards to two objective functions. The (blue area) represents the possible solutions for the benchmark function MOP2 (see Appendix). (b) In scalarization, the problem is optimized by "sliding" the curve defined by the scalarization (green curve) toward its extremes. In this illustration, the PF is not convex, hence, linear scalarization can converge only to one of two possible solutions: The north-west or the south-east extremes. (c) The Pareto front, shown as the (red curve), allows instead a wider range of possible optimal trade-offs.

Scalarization.

In **scalarization**, the human expert explicitly defines an a priori trade-off between different objective functions and combines them into a single performance criterion through a scalarization function $z(\cdot)$. A common scalarization function is the weighted sum [Zadeh, 1963], in which case Equation (5.9) becomes

$$\arg\min_{\boldsymbol{\theta}\in\mathbb{R}^d} z\left(f_1(\boldsymbol{\theta})\dots f_n(\boldsymbol{\theta})\right), \tag{5.10}$$

$$z\left(f_1(\boldsymbol{\theta})\dots f_n(\boldsymbol{\theta})\right) = \sum_{i=1}^{n} \alpha_i f_i(\boldsymbol{\theta}), \tag{5.11}$$

where the weights $\alpha_1, \dots, \alpha_n$ are parameters to be appropriately selected. By designing a suitable scalarization function, the MOO problem is reduced to a straightforward SOO problem.

Scalarization is commonly used in real-world applications due to its ease of use. However, for many real-world applications, the choice of the scalarization itself is an important part of the problem. Small variations in the scalarization or its parameters (e.g., the weights $\boldsymbol{\alpha}$ of a weighted sum) often lead to substantially different solutions (i.e., optimal parameters of the specific scalarization). Moreover, the scalarization and its parameters (and therefore the resulting trade-off) have to be designed a priori, before running the experiments, with often incomplete knowledge about the possible performance of the system. This design is particularly challenging when objectives with different unit measures (e.g., velocities and distances) or different orders of magnitude have to be scalarized together. In practice, it is often unknown even to a human expert which scalarization will lead to the desired trade-off performance [Das and Dennis, 1997].

As a result of these shortcomings, MOO by scalarization is often used in an iterative trial-and-error process where multiple independent optimizations are executed with different scalarizations manually tuned by a human expert [Marler and Arora, 2009]. Due to this independence, no information is shared across multiple optimizations, although they were performed on the same system and can potentially evaluate the same set of parameters multiple times across different optimizations. In a setting where function evaluations are expensive (i.e., we want to minimize the number of evaluations as they are time-consuming and may increase the wear on the hardware) this independence across multiple scalarization is undesirable as it is wasteful of evaluations. Overall, multi-objective optimization by repeated scalarization can be a very inefficient and tedious process. Moreover, it is well known that linear scalarization (e.g., weighted sum) limits the choice of possible trade-off solutions [Das and Dennis, 1997, Messac et al., 2000], as we discuss more in detail in the next paragraph.

Pareto optimization.

A central concept in Pareto optimization is "dominance". Considering the minimization problem in Equation (5.9), a set of parameters $\boldsymbol{\theta}_1$ **dominates** $\boldsymbol{\theta}_2$ when

$$\begin{cases} \forall i \in \{1, \dots, n\}: & f_i(\boldsymbol{\theta}_1) \leq f_i(\boldsymbol{\theta}_2) \\ \exists j \in \{1, \dots, n\}: & f_j(\boldsymbol{\theta}_1) < f_j(\boldsymbol{\theta}_2) \end{cases} \tag{5.12}$$

and we[3] write $\boldsymbol{\theta}_1 \succ \boldsymbol{\theta}_2$. Intuitively, if $\boldsymbol{\theta}_1 \succ \boldsymbol{\theta}_2$, then $\boldsymbol{\theta}_1$ is preferable to $\boldsymbol{\theta}_2$ as it never performs worse, but at least in one objective function it performs strictly better. However, different dominant parameters are equivalent in terms of optimality as they represent different trade-offs.

The goal of **Pareto optimization** is to find the largest *set* $\mathscr{P}$ of dominant solutions with respect to all objective functions called the *Pareto front* [Pareto, 1906]. After the Pareto front (PF) has been estimated during the optimization, one of its possible solutions has to be chosen. In contrast to scalarization, the selection of a particular (Pareto optimal) solution in Pareto optimization is made *a posteriori*, i.e., we can decide on a trade-off among all objectives after experimentation. Choosing a parameter setting *a posteriori* does not require additional experimentation if we want to consider a different trade-off among objectives. We can simply choose a different set of parameters from the Pareto front since the performance of any trade-off is already known.

Pareto optimization is a more general framework than scalarization, as selecting a parameter set can be thought of as choosing the optimal solution for a particular scalarization a posteriori. This intuition brings us to a limitation of scalarization that we already have briefly discussed: Linear scalarizations, such as weighted sum, can only find trade-off solutions lying on convex regions of the Pareto front [Messac et al., 2000]. Intuitively, the weights of a weighted sum define a straight line in the objective function space, which will subsequently be optimized (by translating the line in the objective function space) until converged to a single point in the PF, as shown in Figure 5.3. As a result, only a subset of the PF can ever be discovered by linear scalarization. Moreover, without knowing it beforehand, our desired trade-off might be impossible to obtain. A second limitation of scalarization that Pareto optimization overcomes is the possible waste of evaluations due to the independence assumption of multiple optimizations with different scalarizations (e.g., it would be possible for different scalarizations to evaluate exactly the same parameters multiple time). Instead, within the Pareto optimization framework, there is a single optimization process where all evaluations are considered, and their information content is available at any time to the optimizer (especially in BO). This property is particularly valuable when multiple trade-offs need to be selected by the decision maker for multiple different designs (e.g., depending on a context). One example is robot locomotion, where the context can correspond to different gaits (i.e., walking or running), with different trade-offs of speed, energy efficiency and stability. Overall, these theoretical properties show the limits of linear scalarization, and the benefits of the preferable Pareto optimization.

In order to aid the human decision maker towards an informed decision, it is desirable to have have the most *diverse*, *accurate* and *dense* Pareto front [Zitzler et al., 2003]. The diversity of the performance is important in order to cover the whole trade-off curve and not only subsets of it. High accuracy is required to estimate the performance of the system precisely and, thus, obtain the best possible trade-offs. High density is helpful at providing the decision-maker with a large choice of possible trade-offs such that the desired performance can be finely selected. Providing a Pareto front that does not fulfill these criteria will typically hinder the final decision process and reduce the quality of the selected solution. In the next paragraph, we discuss multiple metrics to evaluate the quality of the Pareto front.

Evaluation of the Pareto Front

Multiple metrics exist to estimate the quality of a Pareto front [Okabe et al., 2003, Zitzler et al., 2003]. A commonly used metric is the hypervolume $\mathscr{H}$, also called $\mathscr{S}$-metric or simply Lebesgue measure [Auger et al., 2012, Naujoks et al., 2005, Zitzler and Thiele, 1999, Zitzler et al., 2008]. Given an estimated Pareto front $\mathscr{P}^*$, the hypervolume

$$\mathscr{H}(\mathscr{P}^*) = \mathrm{Vol}\left(\bigcup_{x \in \mathscr{P}^*} \{ y \in \mathbb{R}^n | x \succ y \succ \mathscr{R} \} \right), \tag{5.13}$$

is computed as the volume of the objective function space dominated by $\mathscr{P}^*$ with respect to a reference point $\mathscr{R}$, with $\mathrm{Vol}(\cdot)$ indicating the Lebesgue measure of the given set.

An illustration of the hypervolume is given in Figure 5.5. In a minimization task, the reference point $\mathscr{R}$ is usually selected as the upper bound of the objectives. Such an upper bound might not be known a priori. Therefore, the

---

[3] Note that in the literature two contrasting notations have been used to indicate a dominating b: Deb et al. [2002a] used $a \prec b$, while Zitzler and Thiele [1999] used $a \succ b$.

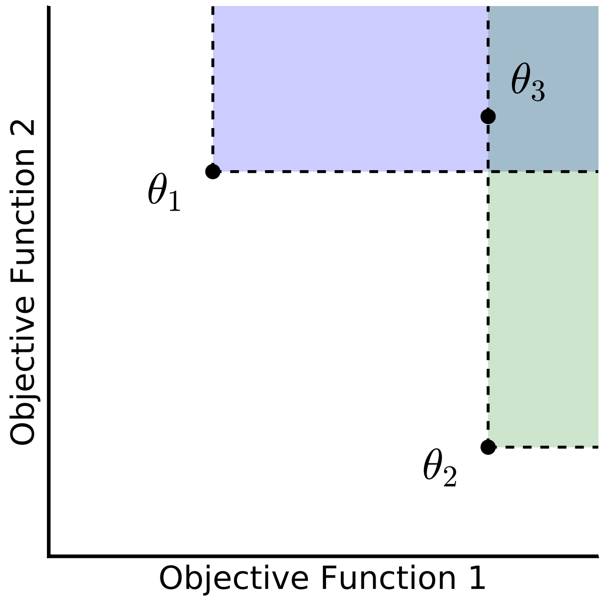


Figure 5.4.: Example of dominance. Both $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ dominate $\boldsymbol{\theta}_3$. Therefore, $\boldsymbol{\theta}_3$ is sub-optimal. However, neither $\boldsymbol{\theta}_1$ nor $\boldsymbol{\theta}_2$ are dominated by any parameter. Therefore, both $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$ belong to the Pareto front.

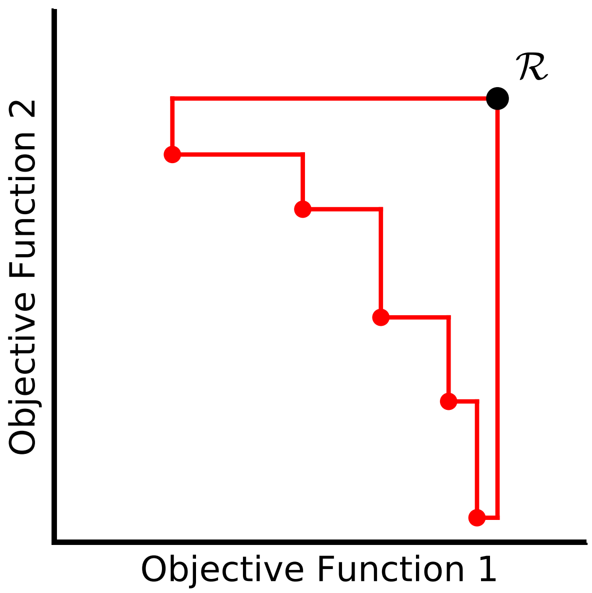


Figure 5.5.: The hypervolume is computed as the volume between the PF and the reference point $\mathscr{R}$.

selection of the reference point might be challenging and require expert knowledge about the objectives. However, since the hypervolume is sensitive to the scale of the objective functions, the choice of an appropriate reference point is important to obtain a meaningful metric[4] as discussed by Zitzler et al. [2008]. As a result, the use of the hypervolume index is not always meaningful or practical. Hence, often multiple indexes are simultaneously used to measure the goodness of the proposed PF, based on different desirable properties, such as diversity, accuracy and density.

In our experimental evaluation, when the ground-truth Pareto front $\mathscr{P}$ is available, we use the hypervolume ratio (HVR)

$$\mathrm{HVR}(\mathscr{P}^*) = \frac{\mathscr{H}(\mathscr{P}^*)}{\mathscr{H}(\mathscr{P})} . \tag{5.14}$$

Optimal values of the HVR are as close as possible to 1, with values HVR > 1 indicating an over-optimistic estimate of the PF (e.g., due to noise as discussed in Section 5.3.1), and values HVR < 1 indicating a poor approximation of the PF.

Optimization with Many Objectives

The term multi-objective optimization is occasionally used exclusively for $n = 2, 3$ objective functions while *many-objective* optimization can be used in case of $n > 3$ objective functions [Farina and Amato, 2002, Fleming et al., 2005]. The use of Pareto optimization when $n > 3$ can be problematic for practical applications for two reasons: First, presenting the Pareto front to the human designer (or even visualizing it) becomes challenging [Agrawal et al., 2004, Blasco et al., 2008]. Second, with a growing number of objective functions the fraction of points belonging to the Pareto front will increase until virtually all points will belong to it [Farina and Amato, 2002, Kukkonen and Lampinen, 2007]. Intuitively, this property is similar to the famous case of a high-dimensional hypersphere where the volume accumulates in its “shell” [Bishop, 2006]. However, this property is undesirable in optimization since the definition of PF lose its purpose of providing support toward helping the human designer in their choice. As a result, extensions to the classic definition of Pareto optimality and dominance have been introduced to address this issue [Farina and Amato, 2002, Kukkonen and Lampinen, 2007].

Readers specifically interested in the challenges of many-objective optimizations are referred to Farina and Amato [2002], Fleming et al. [2005] and Ishibuchi et al. [2008]. In the following, we acknowledge that the two cases, multi- and many-objective optimization, present different challenges. However, we will use the term multi-objective to refer to the most general case $n > 2$.

---

[4] Even when normalizing the objectives in the interval [0,1]

5.2.3 Related work

We now thoroughly discuss related works in MOO (Section 5.2.3), MOBO (Section 5.2.3), robustness for BO (Section 5.2.3), robustness for MOO (Section 5.2.3), and Global sensitivity analysis and reliability-based robust optimization (Section 5.2.3). Subsequently, we detail two relevant approaches to MOBO: ParEGO (Section 5.2.3) and EIHV (Section 11).

Multi-objective optimization

MOO is often performed by means of evolutionary algorithms, such as NSGA-II [Deb et al., 2002b] and SPEA2 [Zitzler et al., 2001]. However, evolutionary algorithms can require thousands or tens of thousands of function evaluations to obtain a good PF. This large number of experiments is often impractical, e.g., for experiments with real systems. Wilson et al. [2001] proposed to use extensive Latin Hypercube sampling followed by polynomial models to assist in the identification of the Pareto front. Similarly, Hartikainen et al. [2011] proposed in the context of MOO a method to approximate the true PF from past observations. However, this approach relies on triangulation and as such provides only a coarse approximation of the PF when few evaluations are available.

Multi-objective Bayesian optimization

Multiple extensions of Bayesian optimization to multi-objective optimization exist [Emmerich et al., 2008, Knowles, 2006, Ponweiser et al., 2008]. Knowles [2006] introduced ParEGO, which iteratively converts a MOO problem into a SOO problem by optimizing an augmented Tchebycheff function. We discuss this approach in more detail in Section 5.2.3. Keane [2006] proposed an extension of the expected improvement acquisition function to the multi-objective case. Other MOO algorithms, such as EIHV [Emmerich et al., 2008] and SMS-EGO [Ponweiser et al., 2008] optimize the hypervolume contribution. Computing the hypervolume contribution can, however, be computationally expensive [Beume et al., 2009] despite the multiple approaches proposed to improve its computational efficiency [Couckuyt et al., 2014, Emmerich et al., 2011, 2008, Hupkens et al., 2015, While et al., 2012]. An in-depth review and analysis of expected-improvement criteria for multi-objective Bayesian optimization methods is presented by Wagner et al. [2010]. Zuluaga et al. [2013] proposed a different approach for model-based MOO based on the reduction of the uncertainty of the performance for potential Pareto optimal parameters. This approach can however be used only for noise-free objective functions since the formulation of the uncertainty does not account for measurement noise. Tesch et al. [2013] applied multi-objective Bayesian optimization using the EIHV acquisition function [Emmerich et al., 2008] to gait optimization of a robotic snake. Binois et al. [2014] introduced a method to compute a probability distribution of the Pareto front using the Vorob'ev deviation. Ariizumi et al. [2014] extended the work of Tesch et al. [2013] by using a heteroscedastic Gaussian process within the multi-objective Bayesian optimization framework. However, the heteroscedastic models are only used to obtain more accurate prediction about the performance distribution and are not used to measure the robustness. Hernández-Lobato et al. [2015] extended the predicted entropy search acquisition function [Hernández-Lobato et al., 2014] to the multi-objective case. For a taxonomy of model-based multi-objective optimization refer to Horn et al. [2015].

Robustness in single-objective Bayesian optimization

In single-objective Bayesian optimization, only few works consider the concept of robustness. These works usually focus on a single aspect of robustness, and to not provide comprehensive frameworks. Kuindersma et al. [2013] introduced a novel acquisition function, expected risk improvement, which extends expected improvement and used it in conjunction with heteroscedastic Gaussian processes. Using this approach, it is possible to quantify the robustness of a set of parameters w.r.t.stochastic noise by estimating the corresponding variance in the model. However, the expected risk improvement acquisition function does not allow to tune the desired trade-off between performance and robustness, which our formulation as a MOO problem allows. Assael et al. [2014] proposed the use of treed GP models to model the heteroscedasticity of the objective function. However, once these models are learned standard Bayesian optimization is performed using the expected improvement acquisition function. Sui et al. [2015] and Berkenkamp et al. [2016] extended Bayesian optimization by proposing a framework for safe optimization trough the use of constraints in the parameter space. The goal of these approaches is to enforce a minimum performance for each evaluation and is therefore only loosely related to our definition of robustness. Nogueira et al. [2016] considered the issue of noise in the parameters by proposing a novel acquisition function that use a probability distribution of the parameters rather then a point-wise estimate. However, this acquisition function does not allow to tune the desired trade-off between performance and robustness, which our formulation as a MOO problem allows. Englert and Toussaint [2016] proposed to learn a binary mask to identify the successful and failing/dangerous areas in the parameter space. Independently, Kurek et al. [2016] developed a similar approach that learn a classifier of feasibility areas in the parameter space.

Robustness in multi-objective optimization

Hughes [2001] focused on uncertainty due to stochastic noise and proposed a ranking schema accounting for the uncertainty of the objective function value. Teich [2001] studied the effects of stochastic noise, and extended the definition of dominance to probabilistic dominance which subsequently used to extend the algorithm SPEA. Similarly, Buche et al. [2002] extended the multi-objective evolutionary algorithm SPEA2 to be tolerant to noisy evaluations. Jin and Sendhoff [2003] introduced a robustness metrics to noise to the parameters in the context of evolutionary algorithms. However, this metric required additional evaluations and its goodness directly depended from the number of additional evaluations. Noisy measurements of objective functions are problematic for most MOO methods, typically leading to sub-optimal solutions. Singh et al. [2003] introduced an extension of NSGA-II, which reduces the negative effects of measurement noise by averaging multiple measurements of the objective function at the same set of parameters. As a result, this extension is experimentally inefficient and impracticable for hardware experiments. Fieldsend and Everson [2005] considered measurement noise, and derived a probabilistic dominance index and a probabilistic formulation of Pareto front used in conjunction with multi-objective evolutionary algorithms. Similarly, Trautmann et al. [2009] introduced another formulation of Pareto dominance in presence of noise. Deb and Gupta [2006] considered parameter uncertainty by computing the mean effective trough the average of multiple neighbor parameters. Hamann et al. [2007] proposed a metric to compute the robustness to stochastic noise. However, this approach requires an elevated number of evaluations in order to accurately approximate the robustness and has no guarantee of convergence even with an infinite number of evaluations. Eskandari and Geiger [2008] introduced a multi-objective evolutionary algorithm capable of dealing with noisy objective functions by incorporating a stochastic ranking. Closely related to our work is the approach presented by Gaspar-Cunha and Covas [2008]. Gaspar-Cunha and Covas [2008] proposed a novel metric to estimate robustness to parameters uncertainty and used this metric as additional objective in multi-objective evolutionary optimization. Bader and Zitzler [2010] studied how to incorporate robustness into the hypervolume indicator and proposed a framework that conceptually unify past robustness approach for evolutionary algorithms. Particularly relevant is the framework proposed by Witting et al. [2013] for computing the robustness to parameters uncertainty in multi-objective tasks. However, in their work, computing the robustness requires to know the analytical form of the objective functions, which is often unrealistic for complex problems. We instead consider a black-box approach where no prior information about the objective functions is necessary. Mlakar et al. [2014] derived a framework for comparing the dominance of MOO solutions in case of stochastic noise.

Global sensitivity analysis & reliability-based robust optimization

Global sensitivity analysis is motivated by the desire to identify the sensitive covariates in order to improve the robustness of the output by decreasing the variability of the sensitive covariates [Iman and Helton, 1988, Iman et al., 1981, McKay, 1992]. However, the identification of the sensitive covariates only provides global information about the sensitivity of the process and does not provide insight into the local robustness (i.e., around a single set of parameters). We are instead interested in studying the so-called local sensitivity analysis. O'Hagan et al. [1999] proposed a sensitivity analysis to uncertainty in the parameters based on the propagation of a distribution through a probabilistic model to estimate the output distribution. This approach is conceptually similar to ours since it creates a model of the objective function, which can subsequently be evaluated without the need to perform evaluations on the real system. However, their procedure relies on expert knowledge to define the input distributions and to analyze and compare the resulting outputs. As described in Section 5.3, we instead take inspiration from control theory to derive bounds of stability that define a metric and can therefore been compared, rather than computing the output distributions. For a survey on (single-objective) robust optimization please refer to the excellent Beyer and Sendhoff [2007]. In the context of reliability-based robust optimization, Lee et al. [2011] proposed to sample from a kriging model in order to estimate the probability of failure using a limited number of expensive evaluations. Motta and Afonso [2016] considered a single-objective design problem as inherently multi-objective and present the designer with the PF between the mean and the standard deviation (considered as metric of robustness) of the objective.

Other

Related to multi-objective optimization are multi-objective sequential decision-making [Roijers et al., 2013], and multi-objective reinforcement learning [Parisi et al., 2014]. In the context of reinforcement learning, the topic of robustness to noise and uncertainties is addressed in risk-sensitive reinforcement learning [Tamar et al., 2012]. For more information about robustness in evolutionary single-objective optimization refer to Jin and Branke [2005].

Algorithm 3: ParEGO

```
$\mathscr{D}$ ⟵ if available: $\{\boldsymbol{\theta}, f(\boldsymbol{\theta})\}$
Prior ⟵ Prior of the response surface
repeat
    Sample $\boldsymbol{\lambda}$
    Compute the scalarized dataset $\tilde{\mathscr{D}} = \{\boldsymbol{\theta}, z(\boldsymbol{\theta})\}$ using Equation (5.15)
    for $i$=1 to $s$ do
        Train a response surface from $\tilde{\mathscr{D}}$
        Find $\boldsymbol{\theta}^*$ that maximizes the acquisition surface $\alpha(\boldsymbol{\theta})$
        Evaluate $\boldsymbol{\theta}^*$ on the real system
        Add $\{\boldsymbol{\theta}^*, f(\boldsymbol{\theta}^*)\}$ to $\mathscr{D}$, and $\{\boldsymbol{\theta}^*, z(\boldsymbol{\theta}^*)\}$ to $\tilde{\mathscr{D}}$
until stopCriteria
```

Algorithm 4: EIHV

```
$\mathscr{D}$ ⟵ if available: $\{\boldsymbol{\theta}, f(\boldsymbol{\theta})\}$
Prior ⟵ Prior of the response surface
repeat
    Train response surfaces $\hat{f}_i$, $i = 1 \dots n$ from $\mathscr{D}$
    Find $\boldsymbol{\theta}^*$ maximizing the acquisition surface EIHV$(\boldsymbol{\theta})$
    Evaluate $\boldsymbol{\theta}^*$ on the real system
    Add $\{\boldsymbol{\theta}^*, f(\boldsymbol{\theta}^*)\}$ to $\mathscr{D}$
until stopCriteria
```

ParEGO

The first and possibly the most widely used multi-objective Bayesian optimization algorithm is ParEGO [Knowles, 2006]. At each iteration of the optimization process, ParEGO randomly scalarizes the objective functions. This scalarization is generated by randomly sampling the weights $\lambda_i \geq 0$, such that $\sum_{i=1}^{n} \lambda_i = 1$ and subsequently computing the Tchebycheff scalarization function

$$z(\boldsymbol{\theta}) = \max_{i=1}^{n} (\lambda_i f_i(\boldsymbol{\theta})) + \rho \sum_{i=1}^{n} \lambda_i f_i(\boldsymbol{\theta}), \tag{5.15}$$

where ρ is a parameter, typically set to the value of 0.05 [Knowles, 2006]. This scalarization is non-linear, which allows to sidestep the limitations of a linear scalarization discussed in Section 5.2.2. ParEGO computes the corresponding scalarized objective $z(\boldsymbol{\theta})$ for each parameter $\boldsymbol{\theta}$ in the dataset $\mathscr{D}$. Subsequently, the dataset $\tilde{\mathscr{D}} = \{\boldsymbol{\theta}, z(\boldsymbol{\theta})\}$ is created, and used to compute the response surface. The optimization continues as in single-objective Bayesian optimization by finding the global optimum of the acquisition function, evaluating it on the real system and adding the outcome to $\mathscr{D}$. After a variable number of evaluations s (possibly a single one), new weights are sampled, a new scalarization computed and the process is repeated until convergence, as shown in Algorithm 3.

ParEGO is non-deterministic since the weights of the scalarization are randomly sampled. At the end of the optimization the solution is considered to be the Pareto front generated from the evaluations performed so far. ParEGO has the limitation that only a limited set of scalarization can be generated since the random weights of the scalarization are bounded. Therefore, in practice, the chosen bounds and scaling of the objective functions still needs to be chosen carefully, as they influence the outcome of the optimization.

EIHV

A more recent approach to multi-objective Bayesian optimization relies on the expected improvement of the hypervolume contribution (EIHV) [Emmerich et al., 2008]. The main idea behind EIHV is to use a comprehensive metric, in this case the hypervolume $\mathscr{H}$, which scalarize the multiple objectives while remaining in a Pareto optimization framework.

Given the hypervolume contribution

$$I(\mathbf{y}, \mathscr{P}^*) = \mathscr{H}(\mathscr{P}^* \cup \{\mathbf{y}\}) - \mathscr{H}(\mathscr{P}^*), \tag{5.16}$$

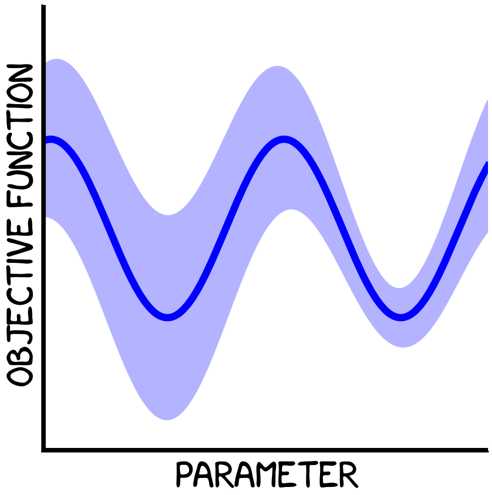


Figure 5.6.: Example of input-dependent stochasticity of the objective function. To different parameters corresponds different levels of noise.

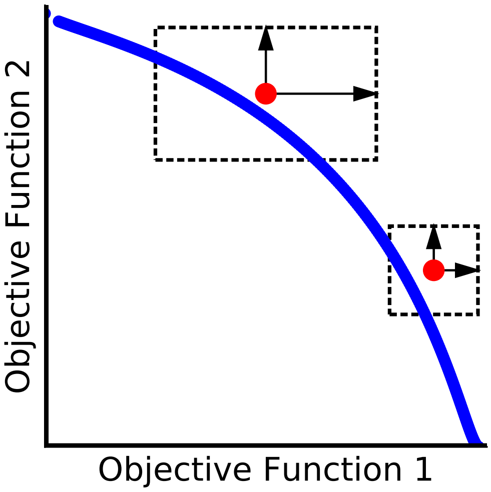


Figure 5.7.: Effects of stochasticity of the objective functions in a MOO task. In the objective function space, the presence of stochastic noise leads to the creation of an hyperrectangle dependent from the covariance matrix of the noise Σ.

where $\mathscr{P}^*$ is the Pareto front computed from the past evaluations, and $\boldsymbol{y}$ is the objective value (or predicted value) of the candidate parameters $\boldsymbol{\theta}$. The expected improvement of the hypervolume contribution can hence be computed as

$$\mathrm{EIHV}(\boldsymbol{\theta}) = \int_{\boldsymbol{y}\in\mathbb{R}^n} I(\boldsymbol{y}, \mathscr{P}^*)\,\mathrm{PDF}(\boldsymbol{y})\,d\boldsymbol{y}\,. \tag{5.17}$$

Subsequently, the MOO task can be rephrased as performing standard single-objective BO using as acquisition function $\alpha(\boldsymbol{\theta}) = \mathrm{EIHV}(\boldsymbol{\theta})$, as shown in Algorithm 4.

5.3 Robust Multi-Objective Optimization

In this section, we first introduce a taxonomy of robustness that considers three different definitions of robustness. Subsequently, based on this taxonomy, we introduce three metrics of robustness. To conclude, we detail our approaches to incorporate these metrics within the Bayesian optimization framework.

5.3.1 A Taxonomy of Robustness

As a base of our taxonomy we consider the unknown objective function to be optimized f from Equation (5.1). Within the Bayesian optimization framework, this unknown function is approximated by the response surface $\hat{f}$ defined by its parameters (or hyperparameters) γ, the input parameters $\boldsymbol{\theta}$, and the corresponding predictions of the objective value $\boldsymbol{y}$, as

$$\boldsymbol{y} = \hat{f}(\gamma, \boldsymbol{\theta})\,. \tag{5.18}$$

Given this general problem formulation, in the following, we identify three distinct notions of robustness with respect to different sources of perturbations.[5]

Stochasticity of the Objective Function.
Starting from Equation (5.18), we consider the stochasticity of the objective function by adding a noise term ϵ_f, such that

$$\boldsymbol{y} = \hat{f}(\gamma, \boldsymbol{\theta}) + \epsilon_f\,, \quad \epsilon_f \sim \mathcal{N}(0, \Sigma)\,. \tag{5.19}$$

---

[5] In this paper, we make use of Gaussian processes as main model for Bayesian optimization. However, the proposed taxonomy of robustness can, without loss of generality, be applied to other probabilistic regression models.

The presence of noise in the objective function implies that two evaluations of the same set of parameters $\boldsymbol{\theta}$ will lead to different objective function value $\boldsymbol{y}$. In MOO tasks, the effects of stochasticity results in the creation of distribution in the objective function space [Mlakar et al., 2014][6], as shown in Figure 5.7. Consequently, the presence of noise in MOO tasks often results in wrong estimations of the Pareto front and, hence, in sub-optimal parameters that have been estimated over-optimistically due to noise. In Section 5.4.1, we demonstrate that the use of a Bayesian model-based approach greatly helps to alleviate this issue.

The stochastic noise ϵ_f in the objective function can in theory be quantified by identifying the noise covariance $\boldsymbol{\Sigma}$. For standard Gaussian Processes, this is often done by defining an additional term in the covariance function that accounts for such noise, as described in Section 8. In this additional term, the noise covariance $\boldsymbol{\Sigma}$ is assumed diagonal, constant and independent from the parameters (i.e., homoscedastic). However, these assumptions often do not hold true for real-world applications as they are too limiting. A more general formulation relax this assumption by considering input-dependent noise (i.e., heteroscedastic)

$$\boldsymbol{y} = \hat{f}(\gamma, \boldsymbol{\theta}) + \epsilon_f(\boldsymbol{\theta}), \quad \epsilon_f(\boldsymbol{\theta}) \sim \mathcal{N}(0, g(\boldsymbol{\theta})). \tag{5.20}$$

Heteroscedastic noise encodes the possibility of the objective function to have different levels of noise for different parameter values as illustrated in Figure 5.6.

Heteroscedastic noise in real-world application for BO was discussed in Calandra et al. [2015d] when optimizing the gait parameters of a bipedal robot. In that example, some parameter settings made the robot consistently fall after a single step (i.e., small mean reward values and small reward variance). Similarly, some parameter settings consistently led to robust and continuously-walking gaits (i.e., large mean reward values and small reward variance). However, a third region of the parameter space led to inconsistent performance where the robot would eventually fall down, but after taking an arbitrary number of steps (i.e. medium mean reward values and large reward variance).

Using the homoscedastic assumption when the underlying function is intrinsically heteroscedastic leads to learning models that will not capture the complexity of the utility function we are interested in modeling. The symptoms of this model mismatch typically manifest during the hyperparameter optimization, where the magnitude of the noise hyperparameter will try to average between the different noise levels present in the underlying heteroscedatic function. In the frequency domain of the noise, this is equivalent to computing a point-wise estimate of a distribution. Additionally, averaging over the noise often has consequences in the estimation of the other hyperparameters of the GP model (especially when employing MAP estimate).

While the heteroscedastic formulation of Equation (5.20) is more flexible then the one of Equation (5.19), the standard Gaussian processes presented in Section 8 can not model heteroscedastic noise. In Section 5.3.2, we will introduce a metric to estimate the robustness w.r.t. the stochasticity of the objective function. Additionally, in Section 5.3.2 we discuss how to train heteroscedastic Gaussian processes models.

Uncertainty in the Parameters.

A second type of uncertainty that can be applied to Equation (5.18) is the uncertainty in the parameters $\boldsymbol{\theta}$ to be evaluated, such that

$$\boldsymbol{y} = \hat{f}\left(\gamma, \boldsymbol{\theta} + \epsilon_p\right), \quad \epsilon_p \sim \mathcal{N}(0, \boldsymbol{\Sigma}). \tag{5.21}$$

In the real-world, this uncertainty derives from the impossibility of evaluating the desired parameters with infinite precision. Possible causes can be insufficient manufacturing accuracy or drifts in the environmental conditions (e.g., changes of temperature). In Section 5.3.3, we will introduce two indexes to measure the robustness w.r.t. the uncertainty in the parameters.

Uncertainty of the Model.

A third type of uncertainty is the uncertainty of the model. The uncertainty of the model considers that the parameters of the learned model $\hat{f}$ are uncertain, such that

$$\boldsymbol{y} = \hat{f}(\gamma + \epsilon_m, \boldsymbol{\theta}), \quad \epsilon_m \sim \mathcal{N}(0, \boldsymbol{\Sigma}). \tag{5.22}$$

A GP considers this kind of model uncertainty explicitly. However, GP can still have uncertainty in the hyperparameters $\boldsymbol{\omega}$, which we can consider a Type-II model uncertainty. In this paper, we do not introduce robustness metrics for model uncertainty or Type-II model uncertainty.

However, when using a fully Bayesian treatment of the model the uncertainty in the model parameters is implicitly considered during the optimization of the model parameters (e.g., the hyperparameters of the covariance function). For more information about hyperparameter optimization see Section 8. Additionally, it should be noted that in the literature the use of robustness w.r.t. uncertainty in the parameters has also been motivated by the need to compensate for the uncertainty in the model, which is usually harder to quantify.

---

[6] Typically $\boldsymbol{\Sigma}$ is assumed to be diagonal and therefore the resulting distributions in the objective function space are hyperrectangle bounding boxes.

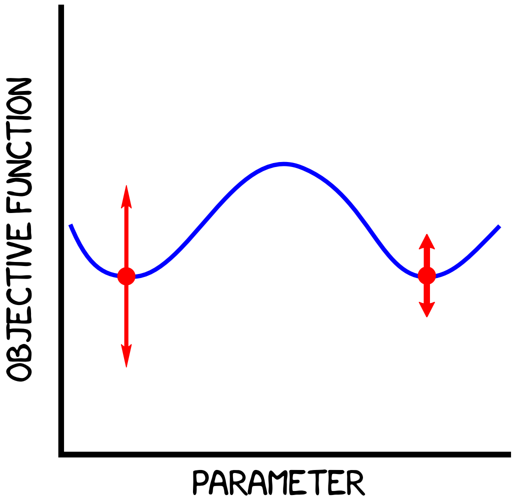


Figure 5.8.: Robustness w.r.t. Stochasticity of the Objective Function: Between two sets of parameters with the same mean, the one with smallest variance is typically preferable. Depending from the circumstances, even sub-optimal parameters might be preferable to the optimal parameters, if their variance is smaller.

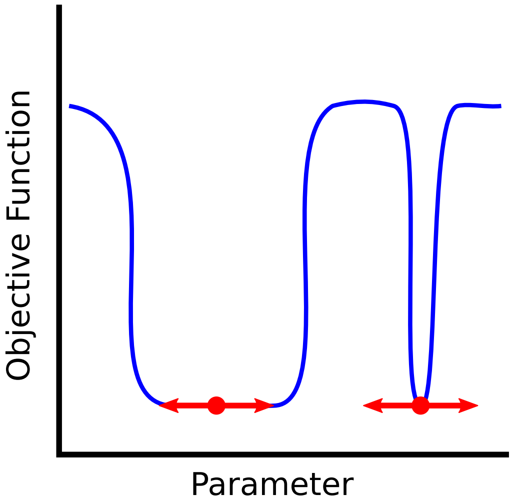


Figure 5.9.: Robustness w.r.t. Parameter Uncertainty: in presence of perturbations of the parameters, between two sets of parameters with the same mean, the one with largest basin with similar performance is typically preferable. Depending from the circumstances, even sub-optimal parameters might be preferable to the optimal parameters if their basin is larger.

5.3.2 Robustness w.r.t. Stochasticity of the Objective Function

The most intuitive metric for measuring the robustness w.r.t. stochasticity of the objective function of the parameters $\boldsymbol{\theta}^*$ is the simple variance

$$r_f(\boldsymbol{\theta}^*) = \sigma^2(\boldsymbol{\theta}^*)\,. \tag{5.23}$$

We now discuss existing approaches to incorporate heteroscedastic noise into GP models.

Heteroscedastic Gaussian Processes

Multiple approaches extend classic GP to model heteroscedastic noise, e.g., [Goldberg et al., 1997, Kersting et al., 2007, Lázaro-Gredilla and Titsias, 2011, Snelson and Ghahramani, 2006, Snelson et al., 2004, Tolvanen et al., 2014, Wang and Neal, 2012]. In this paper, we chose to use the VHGP model introduced by Lázaro-Gredilla and Titsias [2011]. The intuition behind VHGP is to use one covariance function k_f to model the mean of the model (similarly to standard GP) and a separate covariance function k_g to model the noise. The point-wise optimization of the hyperparameters is then performed in a two-stage process that first optimizes the hyperparameters of the mean covariance function k_f, and subsequently optimizes jointly both covariance functions k_f and k_g. The resulting predictive distribution of this model for a new input $\boldsymbol{\theta}^*$ is not Gaussian and cannot be computed analytically. However, it is possible to analytically compute its mean $\mu(\boldsymbol{\theta}^*)$ and variance $\sigma^2(\boldsymbol{\theta}^*)$ to approximate the unknown distribution with a Gaussian (moment matching)

$$\mu(\boldsymbol{\theta}^*) = a_*\,, \tag{5.24}$$

$$\sigma^2(\boldsymbol{\theta}^*) = c_*^2 + e^{\mu_* + \sigma_*^2/2}\,, \tag{5.25}$$

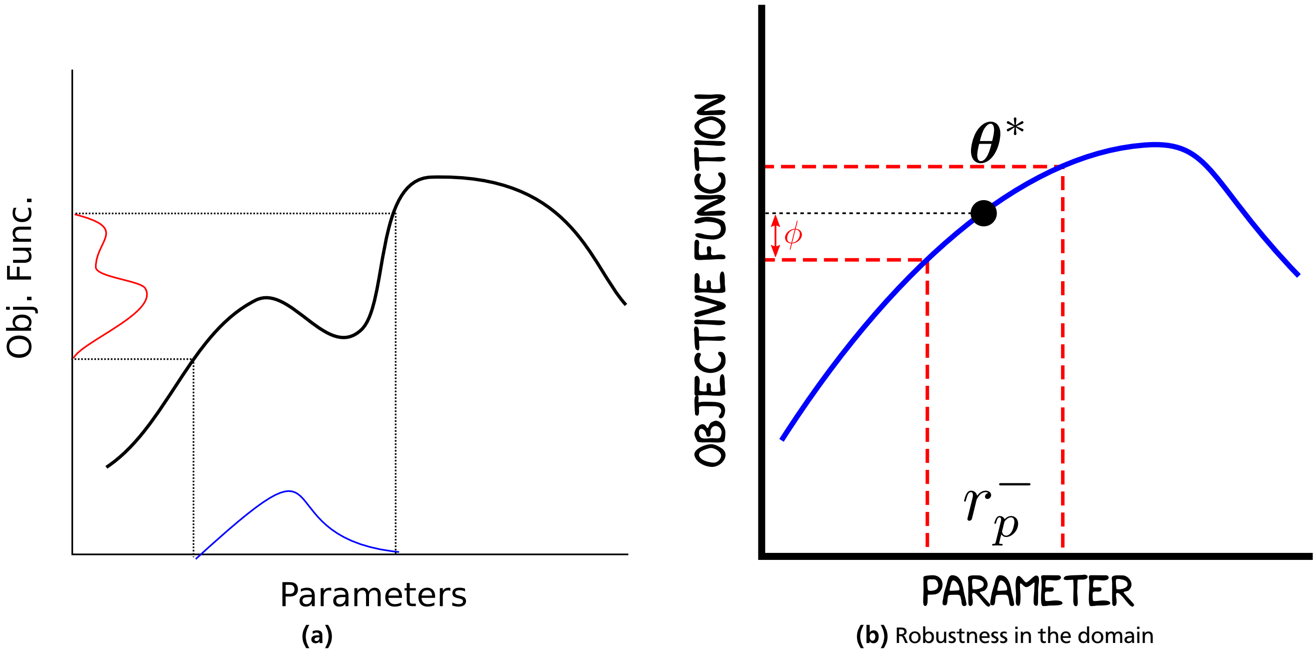
where

$$a_* = \boldsymbol{k}_{f*}^T\left(\boldsymbol{K}_f + \boldsymbol{R}\right)^{-1}\boldsymbol{Y}\,, \tag{5.26}$$

$$c_* = k_{f**} - \boldsymbol{k}_{f*}^T\left(\boldsymbol{K}_f + \boldsymbol{R}\right)^{-1}\boldsymbol{k}_{f*}\,, \tag{5.27}$$

$$\mu_* = \boldsymbol{k}_{g*}^T\left(\boldsymbol{\Lambda} - \tfrac{1}{2}\boldsymbol{I}\right)\mathbf{1} + \mu_0\,, \tag{5.28}$$

$$\sigma_* = k_{g**} - \boldsymbol{k}_{g*}^T\left(\boldsymbol{K}_g + \boldsymbol{\Lambda}^{-1}\right)^{-1}\boldsymbol{k}_{g*}\,, \tag{5.29}$$



(a)

(b) Robustness in the domain

Figure 5.10.: Robustness w.r.t. Parameter Uncertainty:

where $\boldsymbol{R}$ is the diagonal matrix having as elements $\boldsymbol{R}_{ii} = e^{\mu_{*i} - std^2_{*i}/2}$. For more information about variational heteroscedastic GPs we refer to Lázaro-Gredilla and Titsias [2011].

In Equation (5.25), we notice that the predicted variance $\sigma^2(\boldsymbol{\theta}^*)$ is composed of two terms: c_*^2 which depends on the covariance of the mean (and intuitively represents the uncertainty due to the lack of training points that are close to our prediction), and $e^{\mu_* + \sigma_*^2/2}$, which depends on the covariance of the noise, and represents the learned noise model. Note that the presence of these two components that constitute the variance is generally true, even in a standard GP and covariance functions including noise, e.g., in Equation (5.5)). However, in the standard formulation the two components are typically not decoupled since the noise is constant.

When using a heteroscedastic model, we can consider the learned noise distribution as a robustness metric. For simplicity, we assume that the distribution of the noise is Gaussian. Therefore, we can use the simple variance as index. Given a VHGP model we measure the robustness w.r.t. stochasticity of the objective function as

$$r_f(\boldsymbol{\theta}^*) = e^{\mu_* + \sigma_*^2/2} \,. \tag{5.30}$$

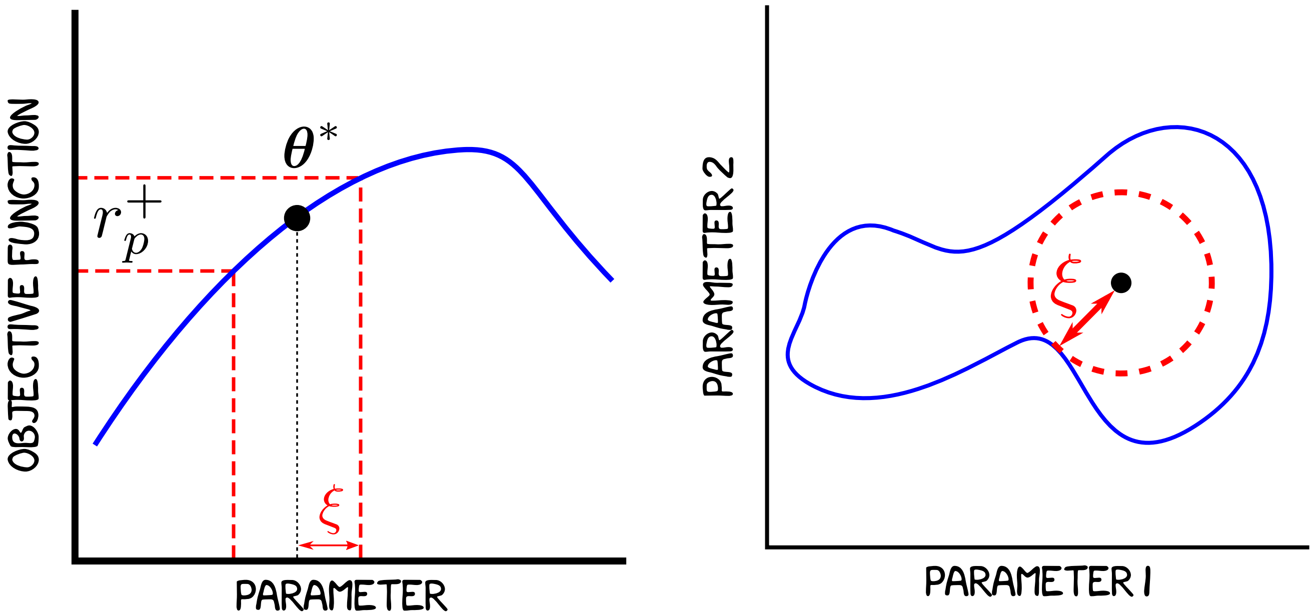
Generally, large variance in the objective value indicate that the considered parameters $\boldsymbol{\theta}^*$ are volatile, while small variance hint at parameters that are robust. As a result, high values of r_f indicate volatility and low values robustness. Note that different objectives could have different noise variances. Although a separate robustness metric could be employed for each separate objective, this is not usually desirable (e.g., due to the limitations of many-objective optimization discussed in Section 5.2.2). Therefore, for MOO tasks it is preferable to use a norm measure such that

$$r_f(\boldsymbol{\theta}^*) = \| e^{\mu_* + \sigma_*^2/2} \| \,. \tag{5.31}$$

5.3.3 Robustness w.r.t. Parameter Uncertainty

The performances of different parameters are not equally influenced by parameter uncertainty. In some cases, perturbations of the parameters might not be very influential. In other cases, even small perturbations translate in substantially different performances. During the selection of the final parameters by the human decision-maker, information regarding the sensitivity of the objective function with respect to the parameter choice can be invaluable. In particular cases, even solutions outside the Pareto front, but with high robustness to parameter uncertainty might be preferable. The robustness w.r.t. uncertainty in the parameters quantifies the variability of the output due to variations in the inputs of the model. As a result of the uncertainty in the parameters the desired choice would be to find parameters $\boldsymbol{\theta}^*$ that lead to a good performance even when they are perturbed.

A first intuitive method to measure robustness would be to estimate the gradients at the given parameter. However, this approach has a number of drawbacks. First, it requires a high number of evaluations to numerically estimate the



(a) Robustness in the co-domain **(b)** Robustness in the co-domain

Figure 5.11.: Robustness w.r.t. Parameter Uncertainty:

gradient. Second, the gradient is a point-wise robustness estimate and, thus, does not provide any information about the neighborhood region. As a result, the derivatives are a metric that is independent from the specific perturbation that is provided to the parameters. A similar approach to derivatives to measure the robustness is the one-at-a-time (OAT) where a single covariate at the time is perturbed [Borgonovo and Plischke, 2016]. This approach is related to directly computing the derivatives of the parameters from the model. However, both OAT and derivatives ignore the correlation between covariates.

A different approach to estimate robustness that make full use of the information about the perturbation rely on computing the distribution of the objective values with parameters subject to perturbations. Given the parameters $\boldsymbol{\theta}^*$ and the parameters perturbation ϵ_p, we can compute the corresponding objective value distribution as $y = f\left(\boldsymbol{\theta}^* + \epsilon_p\right) p(\epsilon_p)$. An illustration is shown in Figure 5.10a. Given the objective value distribution, an important design choice is what metrics to use to measure robustness. This choice can depend from the specific task at hand and from design requisite. For example, for safe optimization the worst case bound of the distribution could be considered.

A common metric of robustness [Deb and Gupta, 2006] is the expected utility

$$r_{\text{eu}}(\boldsymbol{\theta}^*) = \|\mathbb{E}[f\left(\boldsymbol{\theta}^* + \epsilon_p\right)] - f(\boldsymbol{\theta}^*)\|\,, \tag{5.32}$$

where $\|\cdot\|$ can be any suitable norm measure and with the expected objective

$$\mathbb{E}[f\left(\boldsymbol{\theta}^* + \epsilon_p\right)] = \int_{\mathbb{R}^d} f\left(\boldsymbol{\theta}^* + \epsilon_p\right) p(\epsilon_p)\, d\epsilon_p\,. \tag{5.33}$$

The integral of Equation (5.33) is often analytically intractable (both when evaluating the real objective function or when evaluating a response surface, such as non-parametric GP models), in which case, the approximation

$$\mathbb{E}[f\left(\boldsymbol{\theta}^* + \epsilon_p\right)] \approx \tfrac{1}{n_s} \sum_{i=1}^{n_s} f\left(\boldsymbol{\theta}^* + \epsilon_p\right) p(\epsilon_p)\,, \tag{5.34}$$

is frequently used. However, the expected utility is not always a useful metric since it uses very little information about the distribution of the objective value given the perturbation. One simple example which demonstrate this limitation would be a simple slope function; Both a constant function and a slope function would have the same expected utility when evaluating the parameter at the intersection between the two (i.e., where the parameters have the same value). Nonetheless, perturbing the parameters of the constant function would result in no change in the performance, while perturbing the parameters in the case of the slope would change the performance.

To overcome the limitation of these metrics, we now present two novel indexes, $r_p^+(\boldsymbol{\theta})$ and $r_p^-(\boldsymbol{\theta})$ to measure the robustness to parameter uncertainty. While any distribution $p(\epsilon_p)$ could be used as parameters perturbation, in the remaining of this chapter we will consider a uniform distribution $\mathcal{U}(-\xi, +\xi)$ inspired by control theory's bounded-input, bounded-output (BIBO) stability [Sastry, 2013].

Algorithm 5: Robustness w.r.t. Uncertainty in the Parameters 1

```
Input: Model $\hat{f}$, Parameters $\boldsymbol{\theta}^*$, Radius neighborhood $\xi$, Number of samples $N$
$r_p^+ \leftarrow 0$
$\mu_0, \sigma_0 \leftarrow \hat{f}(\boldsymbol{\theta}^*)$
for $i$=1:N do
    $\boldsymbol{\theta}^i \leftarrow \boldsymbol{\theta}^* + \mathcal{U}(-\xi, +\xi)$
    $\mu_\xi, \sigma_\xi \leftarrow \hat{f}\left(\boldsymbol{\theta}^i\right)$
    $r_p^+ \leftarrow r_p^+ + \mathrm{NLL}\left(\mu_0, \sigma_0, \mu_\xi\right)$
return $r_p^+/N$
```

Algorithm 6: Robustness w.r.t. Uncertainty in the Parameters 2

```
Input: Model $f$, delta performance $\phi$
$r_p^- \leftarrow 0$
$\xi_u = \arg\min_\xi g$
Sample $\xi_*$ within $\xi_u$
$r_p^- \leftarrow largest \xi_*$ that satisfy Equation (5.39)
return $r_p^-$
```

Robustness in the domain

Our proposed robustness metric is motivated by the possibility of exploiting additional information provided by the use of a probabilistic model, compared to the use of simple evaluations from the objective function. In particular, the presence of variance is crucial to identify both stochastic noise and model uncertainty. The proposed metric is computed as

$$r_p^+(\boldsymbol{\theta}^*) = r_s(\boldsymbol{\theta}^*)/r_u(\boldsymbol{\theta}^*), \tag{5.35}$$

$$r_s(\boldsymbol{\theta}^*) = \tfrac{1}{n_s}\sum\nolimits_s -\log(N(\hat{f}(\boldsymbol{\theta}_s)\,|\boldsymbol{\mu}_f(\boldsymbol{\theta}^*), \boldsymbol{\sigma}_f(\boldsymbol{\theta}^*))), \tag{5.36}$$

$$r_u(\boldsymbol{\theta}^*) = -\log(\mathcal{N}(\boldsymbol{\mu}_f(\boldsymbol{\theta}) + 3\boldsymbol{\sigma}_f(\boldsymbol{\theta})|\boldsymbol{\mu}_f(\boldsymbol{\theta}), \boldsymbol{\sigma}_f(\boldsymbol{\theta}))). \tag{5.37}$$

The first term $r_s(\boldsymbol{\theta}^*)$ is the sum of the negative log likelihoods of the perturbed parameters, w.r.t. the distribution of the parameter $\boldsymbol{\theta}^*$. The second term $r_u(\boldsymbol{\theta}^*)$ serves instead the purpose of rescaling the first term such that $r_p^+(\boldsymbol{\theta}^*) = [0, 1]$. It is important to notice that the samples are evaluated on the response surface and therefore do not require additional experiments on the real system. As such the number of samples evaluated can be in the order of the thousands with only minor time consumption. (This measure is related to the Gradient and the Hessian, but both of these measures are point-wise, while our proposed robustness is region-wise) This robustness metric is particularly valuable if an estimate of the parameters uncertainty is available, such that ξ can be easily defined. An illustration of this metric is shown in Figure 5.11.

Note that this metric requires the presence of a probabilistic model. Importantly, the use of a response surface to estimate the robustness can drastically reduce the number of evaluations performed on the objective function. The reason is that with typical robustness metrics the number of evaluations is dependent from the number of points where the robustness has to be estimated. Instead, when using a response surface the number of points where to estimate the robustness is independent from the number of evaluations used to train the response surface itself. In Section 5.4, we present experimental results which evaluate this property.

Robustness in the co-domain

A second measure of robustness can be defined in terms of maximum desired deviation from the reference value, and in that case a robustness index can be defined as the largest neighborhood ξ such that

$$r_p^-(\boldsymbol{\theta}^*) = \max \xi \in [0, \infty] \tag{5.38}$$

$$s.t. \|\hat{f}(\boldsymbol{\theta}^*) - \hat{f}(\boldsymbol{\theta}^* \pm \xi)\| < \phi \tag{5.39}$$

where ϕ is the open parameter that defines the maximum acceptable loss of performance. Finding the maximum ξ that satisfies the constrain is equivalent to search for the smallest ξ along the isocurve $\hat{f}(\boldsymbol{\theta}^* \pm \xi) = \hat{f}(\boldsymbol{\theta}^*) - \phi$. This computation is, up to the best of our knowledge, analytically intractable when using Gaussian processes models. Therefore, we

Algorithm 7: Robust Multi-objective Bayesian Optimization

```
Compute a response surface for each objective
Compute robustness to parameters from the response surfaces
Compute robustness to noise from the response surfaces
Add the two robustness as additional objectives
Use the MOO acquisition function of choice
```

have to resort to numerical approximations. Even numerically estimating ξ is not trivial and can raise two critical issues: 1) performing the estimation within a reasonable numerical complexity 2) providing any guarantee of the accuracy of the estimation. A basic approach to estimate ξ is to sample many parameters in a neighborhood of the parameters of interest $\boldsymbol{\theta}^*$. However, this approach is computationally expensive since we will need to sample from an ever growing volume until the constraint of Equation (5.39) is no longer satisfied.

To reduce the computational cost, we propose to use a sample-efficient approach to compute a probabilistic estimator of the ξ. The first step is to find an upper limit ξ_u of the ξ by performing gradient descent until the constraint are violated[7]. Once this upper limit is known, we can use a sampling schema that bound the distance between the parameters evaluated within this upper limit. Hence, we can compute the probability that the response function in the space between the points violate the constraint. By doing so, we can regulate the number of parameters evaluations necessary to obtain the desired probability.

The main difference between the two indexes $r_p^-(\boldsymbol{\theta}^*)$ and $r_p^+(\boldsymbol{\theta}^*)$ lies in the open parameter: Whereas in r_p^+ the open parameter is defined in the domain of the objective function (i.e., the parameter space), in r_p^- the open parameter is defined in the co-domain of the objective function (i.e., the value of the objective function itself). As a result, the most appropriate of the robustness measures can be chosen based on the specific knowledge of the system that we want to optimize and the design constraint.

5.3.4 Robust Multi-Objective Bayesian Optimization

Once the robustness metrics introduced in Section 5.3.2 and Section 5.3.3 are computed, they can be added to the MOO task as additional objectives [Gaspar-Cunha and Covas, 2008]. While this is not the only possibility [Bader and Zitzler, 2010], we believe it is a sensible choice, since being additional objectives, the designer can now decide their desired trade-off *a posteriori*.

5.4 Experimental Evaluation

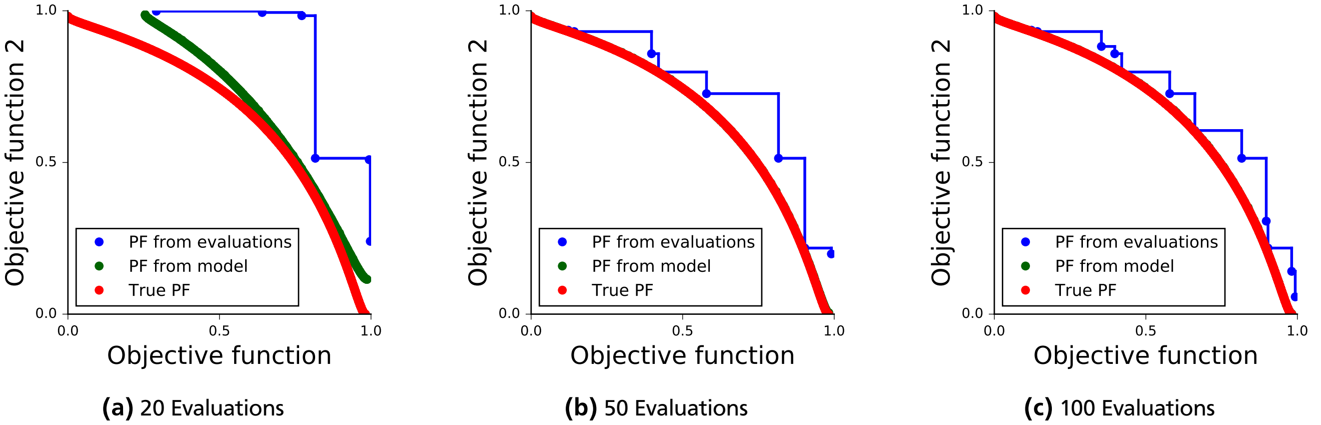
In this section, we experimentally the different aspects of our approach to robust multi-objective Bayesian optimization. In particular, we evaluate the following: We demonstrate that using a model-based approach allows to obtain better approximation of the Pareto front, even in an off-line setting (Section 5.4.1) and in presence of measurement noise (Section 5.4.2). In Section 5.4.3, we evaluate our robustness metric to input-dependent noise. Robustness to parameter uncertainty is evaluated in Section 5.4.4. Additionally, the use of robustness to parameter uncertainty to compensate for errors in the model is evaluated in Section 5.4.5.

5.4.1 Pareto Front Approximation using the Model of the Response Surface

We now demonstrate how model-based approaches can exploit past evaluations to provide an accurate, dense and smooth Pareto front approximation from already performed evaluations. As benchmark we employ the MOP2 and the ZDT3 functions (see Benchmark Test Functions). Instead of directly using the evaluations to estimate the PF, we used these evaluations to train a response surface. Subsequently, we evaluated the response surface to obtain an estimate of the PF. This approach can be advantageous if only a small number of evaluations are available as shown in Figure 5.12: With only a small number of evaluations the resulting approximation of the real Pareto front (blue curve) is poor, discontinuous and sparse. However, when using the response surface model we can compute a dense Pareto front approximation (green curve), which approximates the ground-truth Pareto front (red curve) better. The same conclusion is reached by analyzing the HVR in Figure 5.13, where using the response surface approximates the ground truth better. Note that the goodness of the Pareto front obtained from the response surface depends on the past evaluations. Hence, the approximation can be inaccurate when only very few evaluations are available. An additional advantage of using a model-based approach is

---

[7] This optimization can be performed by minimizing the auxiliary objective function $g = \left|\hat{f}(\boldsymbol{\theta}) - \phi\right|$.



(a) 20 Evaluations **(b)** 50 Evaluations **(c)** 100 Evaluations

Figure 5.12.: Pareto Front Approximation using the Model of the Response Surface: Each plot shows for the MOP2 function the PF from the evaluated randomly chosen parameters (blue curve), the PF from the response surface learned using the evaluated parameters (green curve) and the real PF (red curve). (a) Using only a few evaluations leads to a poor approximation of the real PF. Using the response surface learned from the same evaluations already improves the quality of the approximated PF. (b,c) With more evaluations, the response surface accurately approximates the real PF. In contrast, the PF from the evaluations is still a poor and sparse approximation.

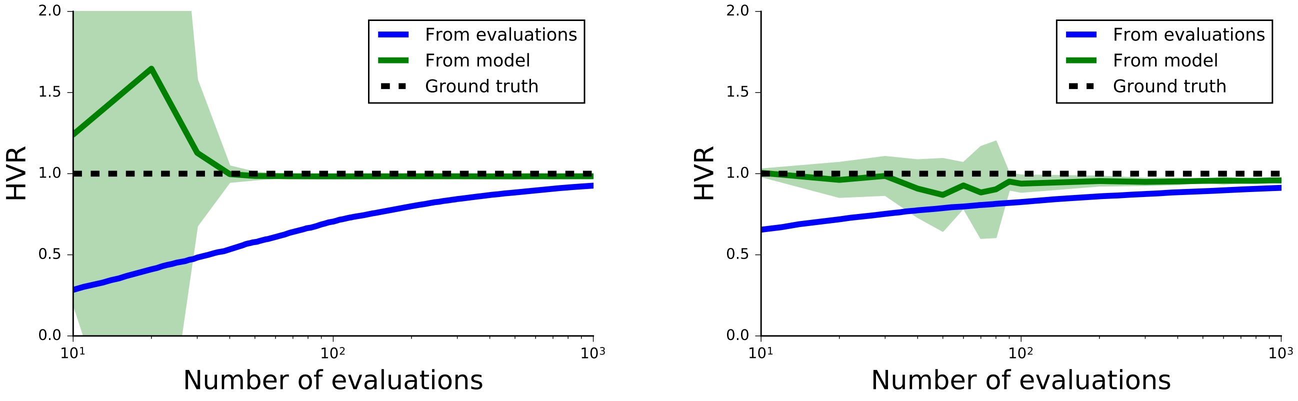
Function	Number Evaluations	HVR (mean ± 2std)	
		From Evaluations	From Model
MOP2	50	0.673 ± 0.236	1.032 ± 0.317
	100	0.810 ± 0.176	0.973 ± 0.086
	1000	1.167 ± 0.093	0.971 ± 0.051
ZDT3	50	0.949 ± 0.199	0.839 ± 0.388
	100	1.037 ± 0.182	0.926 ± 0.255
	1000	1.271 ± 0.110	0.953 ± 0.034

Table 5.1.: Pareto Front Approximation in Presence of Stochastic Noise: Mean ±2std of the HVR for the MOP2 and ZDT3 functions varying the number of evaluations. Using the noisy measurements over-estimate the true PF, and as a result the HVR. The use of a model allows to reduce the effects of noise and to estimate the PF, and as a consequence the HVR, more accurately. HVR values closer to 1 are generally better.

that we can compute a probability for each point to belong to the Pareto front, rather then a simple binary classification. Such probability can be helpful to the decision-maker. For example, between a set of parameters which has not been evaluated, but are considered from the model to have a high likelihood of being Pareto optimal, and a set of parameters evaluated, but which the model consider unlikely to be Pareto optimal, we might prefer the first.

5.4.2 Pareto Front Approximation in Presence of Stochastic Noise

We will now extend the analysis performed in the previous section to the case of stochastic measurement noise in the objective functions (as defined in Equation (5.19)). The presence of measurement noise in MOO is problematic since it can lead to wrongly estimate the PF. In the next experiments, we show that the use of a model-based approach greatly alleviate the estimation errors introduced by measurement noise. Similarly to Section 5.4.1, we tested the MOP2 and the ZDT3 functions. For the MOP2 function, we considered the additive noise $\epsilon_f \sim \mathcal{N}(0, \Sigma)$ with $\Sigma = \text{diag}([0.0025, 0.0025])$. After uniformly randomly sampling 50,000 parameters, we evaluated them on the noisy MOP2. As shown in Figure 5.14, approximating the PF from the resulting measurements leads to an over-optimistic PF since the evaluations with highest optimistic noise dominate all other noisy evaluations. However, when creating a response surface from noisy evaluations it is possible to approximate the real Pareto front accurately. The noise variances learned by the GP model through hyperparameter optimization is $\Sigma = \text{diag}([0.0024, 0.0026])$, which closely resembles the real noise. Such values of the



(a) MOP2.

(b) ZDT3.

Figure 5.13.: Pareto Front Approximation using the Model of the Response Surface: Mean ±2std of the HVR for the MOP2 and the ZDT3 functions varying the number of evaluations. The PFs computed using the plain evaluations (blue curve) even after 1000 evaluations are still not dense nor accurate. As a result, the estimated HVR do still have a visible difference to the ground truth. The use of models (green curve) allows to estimate the PF, and as a consequence the HVR, more accurately even with a small number of evaluations.

hyperparameters suggests that the GP model truthfully capture the underlying function and its noise, and subsequently accurately approximate the PF. To confirm this hypothesis we analyzed the HVR for the MOP2 and the ZDT3 functions when varying the number of evaluations. As shown in Figure 5.15, increasing the number of evaluations, the value of the hypervolume (and the resulting HVR) computed from the measurements is over-estimated. Nevertheless, the use of a response surface allows to accurately approximate the ground truth of the hypervolume. Table 5.1 reports numerical values of the HVR. Note that for both the MOP2 and the ZDT3 there exists a specific number of evaluations that lead to an HVR close to 1. However, this is not due to the correct estimation of the PF, but because the over-estimation of some PF points will compensate for the poor estimation of other parts of the PF.

5.4.3 Robustness w.r.t. Stochasticity of the Objective Function

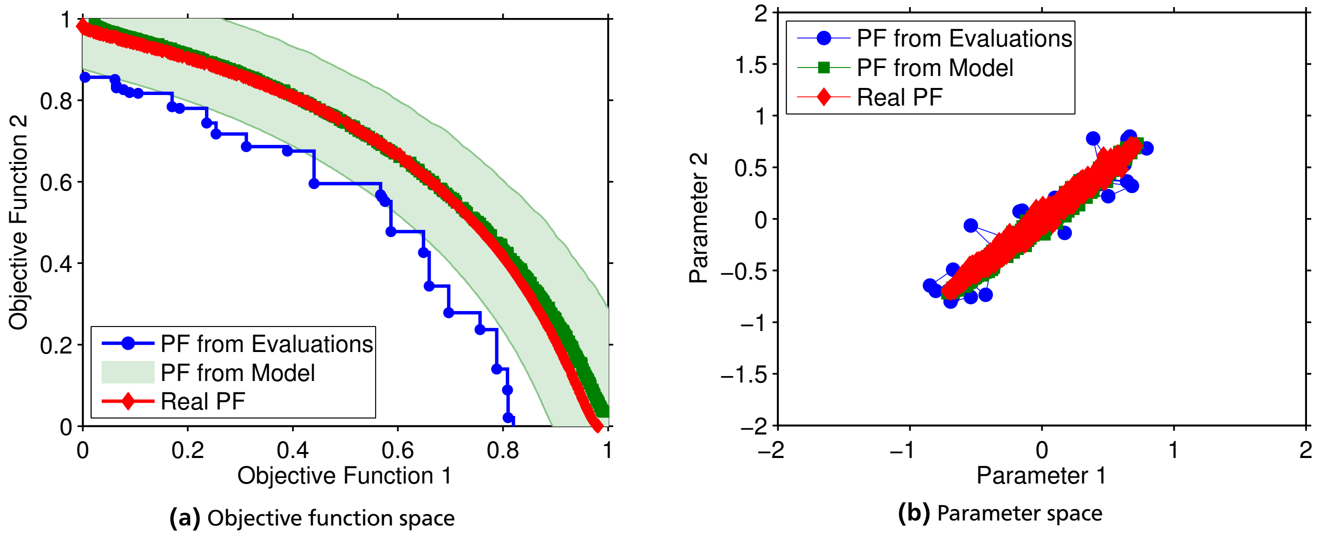
In the following, we demonstrate the use of VHGPs for modeling heteroscedastic objective functions and the metric of robustness to measurement noise introduced in Section 5.3.2. As objective function we consider the Branin function with heteroscedastic noise (see Appendix for more details). We trained an VHGP model using 500 uniform randomly sampled parameters and the corresponding measured objective values. Subsequently, we uniform randomly sampled 1000 parameters $\boldsymbol{\theta}_*$ and predicted the corresponding distribution using the trained model. Using these predictions, we can consider the robust multi-objective optimization task having as first objective the mean of the prediction, and as second objective the robustness index $r_f(\boldsymbol{\theta}_*)$. In Figure 5.16, we can see the prediction in the objective function space, and the corresponding Pareto front. It can be noticed that the estimate from the model, and as a result the Pareto front, is qualitatively similar to the analytical ground truth. Moreover, projecting the Pareto optimal parameters in the parameters space shows that the PF is correctly identified, as shown in Figure 5.17.

5.4.4 Robustness w.r.t. Parameter Uncertainty

In this experiment, we evaluate the robustness w.r.t. parameter uncertainty introduced in Section 5.3.3. For this purpose, we perform a sensitivity analysis of the MOP2 and on the RMTP3 test functions. For the MOP2 function, we collected 625 sampled parameters along a grid and the corresponding values of the objective function and learned a response surface from them. From the resulting response surface, we computed the robustness index of Equation (5.37) for the parameters along a 300 × 300 grid.

Figure 5.18b visualizes the resulting intensity map of the robustness measure in parameter space. Higher values of robustness denote parameters to whose perturbations the objective function is less sensitive. For the MOP2 test function, we notice four areas of the parameter space, which are robust to parameter variations: two respectively at the North-West and South-East corners, and two respectively at the coordinates [-0.7 -0.7] and [0.7 0.7].

Figure 5.18a shows that along the corresponding Pareto front there are only two areas with robust solutions, respectively at the North-West and South-East corners of the objective function space. These two areas of robust solutions correspond to the parameters at the coordinates [-0.7 -0.7] and [0.7 0.7]. Instead, the other two areas of the parameter



(a) Objective function space

(b) Parameter space

Figure 5.14.: Pareto Front Approximation in Presence of Stochastic Noise: (a) Estimating the real Pareto front (red curve) exclusively from noisy evaluations leads to an over-optimistic Pareto front (blue curve). Using a response surface, we obtain a better approximation of the Pareto front (green curve) and estimate the 95% confidence of the noise (green area). (b) The poor approximation of the Pareto front from noisy measurements leads to sub-optimal parameters (blue dots). The parameters from the response surface (green dots) closely resemble the parameters of the real Pareto front (red dots).

space are mapped in the objective functions space to sub-optimal solutions far away from the Pareto front (in the North-East corner of the objective function space). Overall, the results obtained by the robustness metric is consistent with the symmetry of the MOP2 function.

When considering the Pareto optimality all solutions on the Pareto front are equally good. However, if we additionally consider the robustness with respect to the parameter choice this is no longer true. When using model-based MOO, an informed decision by the human decision-maker can and should consider the robustness of the solutions. Moreover, it should be noted that in order to compute the robustness metric without the use of the model, we would have needed hundreds of thousands of evaluations.

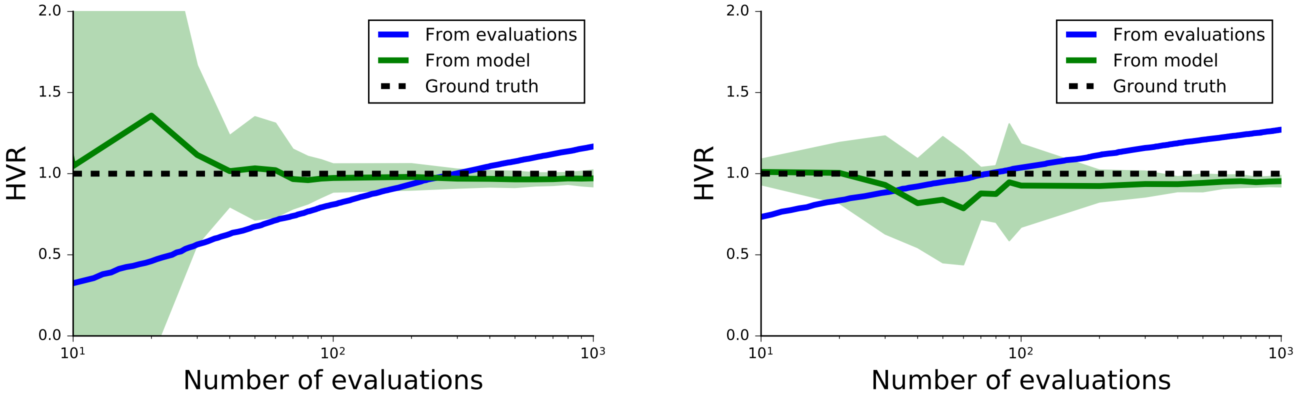
We performed further evaluations on the RMTP3 function, which is explicitly designed to test robustness (see Benchmark Test Functions). After learning a model of the function from 500 randomly sampled parameters, we computed the robustness along a 300 × 300 grid. The global minimum region corresponding to $\boldsymbol{\theta}_2 \approx 0.85$ is correctly identified as having a low robustness, as visible in both the parameters space (see Figure 5.19b), and in the PF (see Figure 5.19a). It can also be noticed that around $\boldsymbol{\theta} = [0.7, 0.4]$ and $\boldsymbol{\theta} = [0.1, 0.3]$ there are two ellipsoids with a relatively high sensitivity. The reasons for these regions is the high uncertainty in the GP model due to the lack of training points in the immediate neighborhood, which resulted in low robustness guarantees.

5.4.5 Parameter Uncertainty as Robustness to Model Uncertainty

We now evaluate the use or robustness to parameters uncertainty as a indirect metric of robustness to model uncertainty, and in particular to hyperparameter uncertainty.

After sampling N samples from the Branin function, we trained a model using MAP estimates to compute the "optimal" hyperparameters. Subsequently, we predicted the objective function and the robustness of the objective along a grid when using a model with perturbed hyperparameters (w.r.t. the "optimal" hyperparameters). Figure 5.20 shows there is a clear correlation between our robustness index to parameter uncertainty and maximum error in the prediction even when the hyperparameters are sub-optimal.

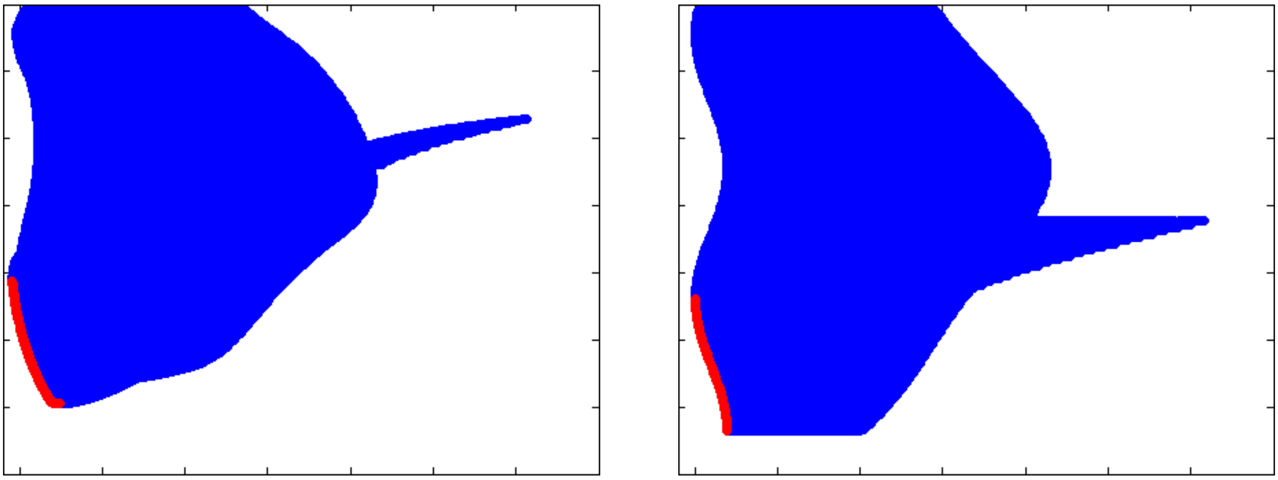
Intuitively, this result can be explained by the observation that often parameters on a "plateau" (i.e., with a high robustness to parameter uncertainty) will not radically change if the model is slightly different. On the other hand, the performance of parameters that are in regions of high variability are often more likely to change as a result of model changes.



(a) MOP2.

(b) ZDT3.

Figure 5.15.: Pareto Front Approximation in Presence of Stochastic Noise: Mean ±2std of the HVR for the MOP2 and ZDT3 functions varying the number of evaluations. The PFs computed using the noisy measurements (blue curve) over-estimate the true PF, and as a result the HVR. The use of a model (green curve) allows to reduce the effects of noise and to estimate the PF, and as a consequence the HVR, more accurately.



(a) Estimated from the HGP model

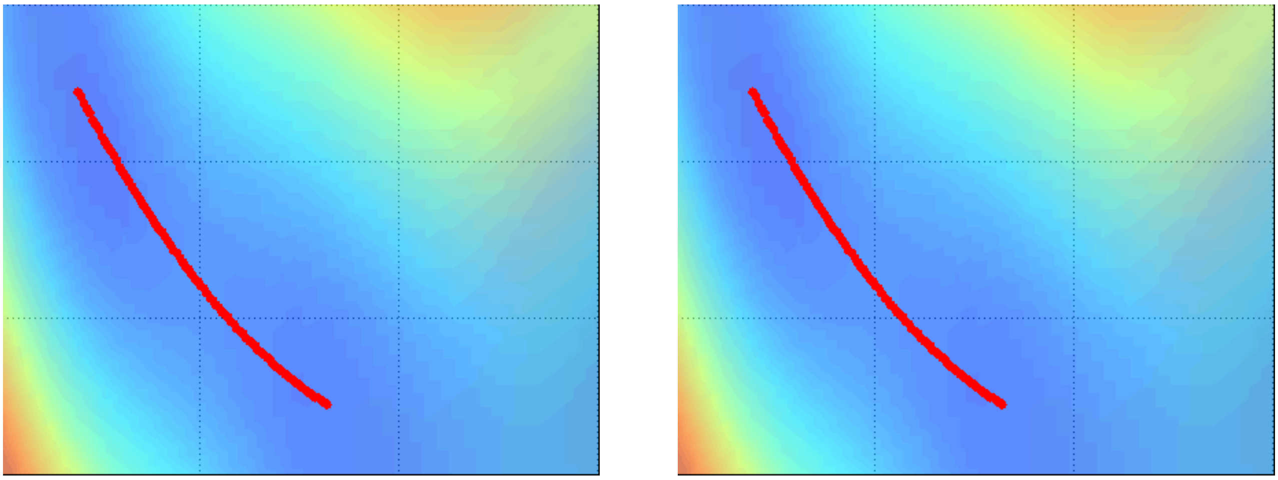
(b) Ground truth

Figure 5.16.: Robustness w.r.t. Stochasticity of the Objective Function: Trade-off curve and Pareto front (red curve) for the noisy Branin function.

5.5 Discussion

Optimization is a crucial task in many fields and especially for Machine Learning. In fact, many problems can be reduced or transformed into an optimization task. It is thus essential to have fast and robust algorithms that can be applied to different problems.

In Machine Learning, the difference between a successful method and an unsuccessful one is often to find the right objective function to optimize. Typically, these objectives strive to find a good balance between different objectives, such as modeling the training data and being capable of generalize to new data (e.g., by mean of regularization), or similarly to supervised learning and unsupervised learning. Hence, MOO tasks are in practice extremely common in ML and many ML optimization tasks could be rephrased as MOO tasks. At the same time, robustness of the parameters might hint to good generalization of the model to unseen test set, while sensitivity of the parameters might indicate overfitting (this is the hypothesis behind regularization approaches such as early-stopping and stochastic gradient descent).



(a) Estimated from the HGP model **(b)** Ground truth

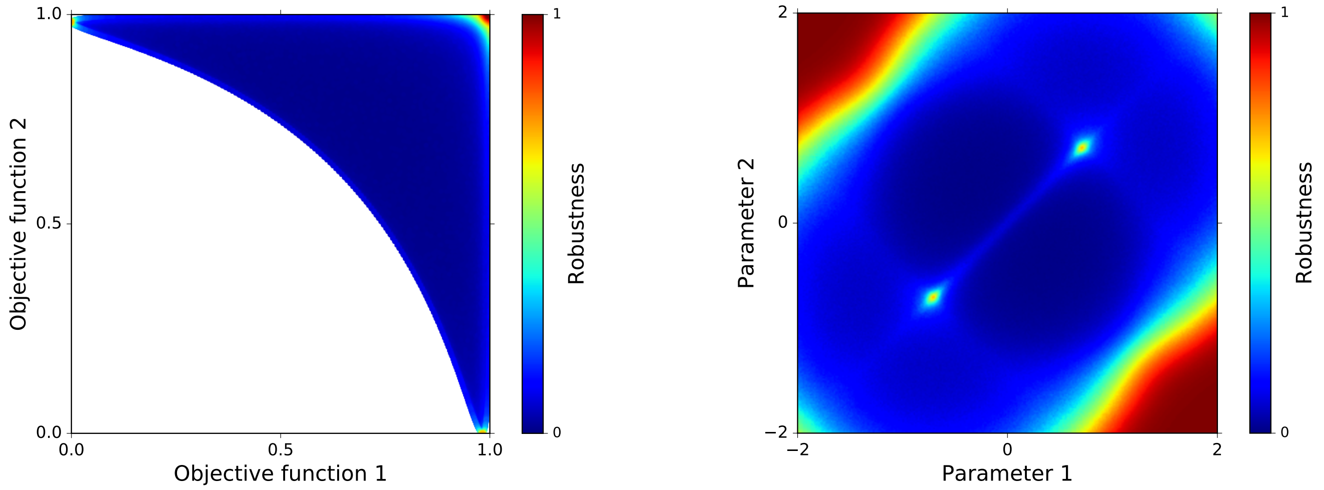
Figure 5.17.: Robustness w.r.t. Stochasticity of the Objective Function: Pareto front for the noisy Branin function in the parameters space.

5.6 Summary

In this chapter, we presented a unifying framework for robust multi-objective optimization using Bayesian optimization. The theoretical contributions of this paper can be applied to a wide range of applications from machine learning to robotics, engineering, biology, experimental design and finance.

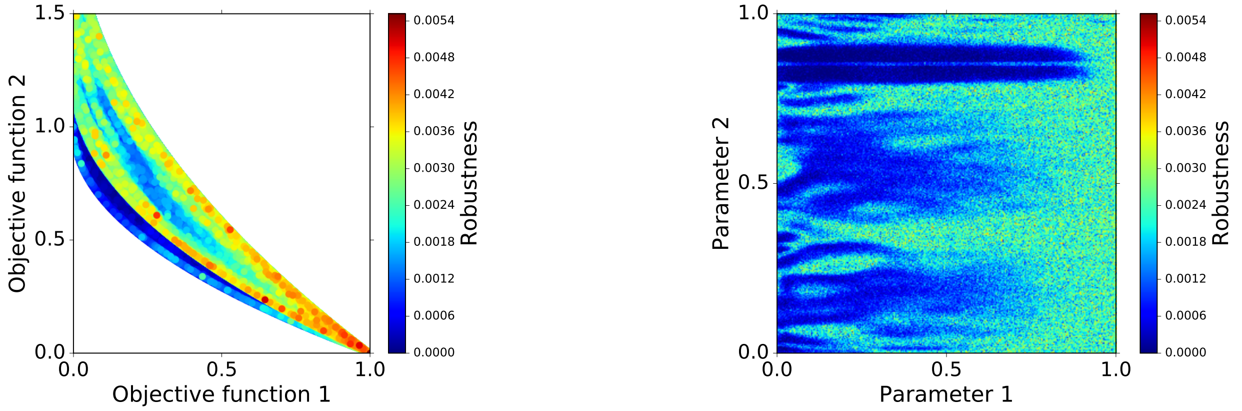
Crucial to the contributions of this chapter is the use of a Bayesian model-based approach. Trough the use of models we demonstrated it is possible to improve the quality of the Pareto front, even in off-line situations. Additionally, it is possible to compute robustness measures that are meaningful and accurate, without the need to perform extensive additional evaluations. These contributions push the state-of-the-art in both multi-objective optimization and robust optimization.

In future work, we will apply our proposed approach to real-world machine learning and robotics tasks.



(a) Objective function space **(b)** Parameter space

Figure 5.18.: Robustness w.r.t. Parameter Uncertainty: MOP2 function (a) Along the Pareto front only the solutions at the two extremities of the front are robust. A further area of stable solutions is visible in the North-East corner, but the values of the objective functions are sub-optimal. (b) The robust solutions along the Pareto front are the two small areas at [-0.7 -0.7] and [0.7 0.7]. The large portions of the parameter space on the corners corresponds to the sub-optimal solutions.



(a) Objective function space **(b)** Parameter space

Figure 5.19.: Robustness w.r.t. Parameter Uncertainty: RMTP3 function (a). (b) .

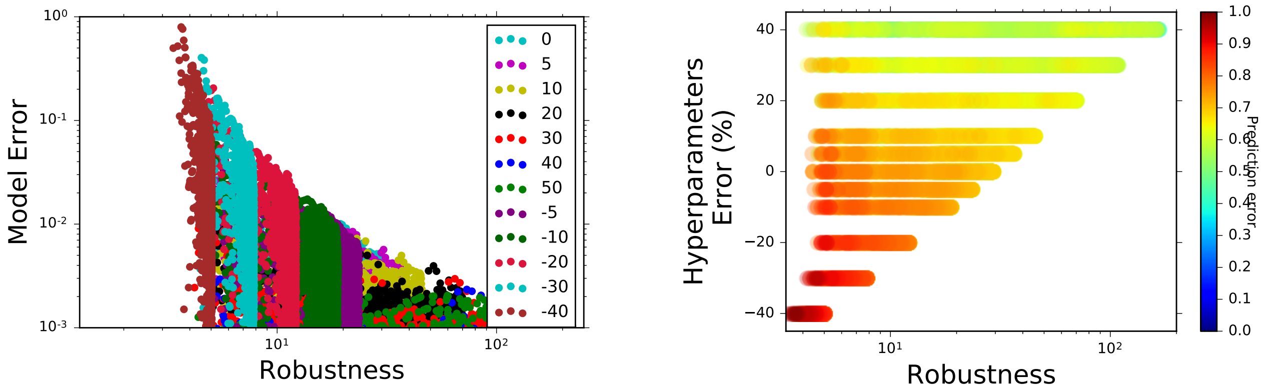


Figure 5.20.: Robustness to parameters uncertainty for the Branin function to compensate for uncertainty in the model parameters.

6 Conclusion

> Uproot your questions from their ground and the dangling roots will be seen. More questions!
>
> Mentat Zensufi

In this thesis, we have discussed the application and tailoring of Bayesian modeling for optimization and control in robotics. In the next section, we provide an overview of the key contributions of this thesis. Following, we discuss open challenges and possible future directions.

6.1 Summary

This thesis contribute to the state-of-the-art of Bayesian modeling, and to its application to the learning of dynamic models used for control and to the automatic tuning of parameters of robotic systems.

In Chapter 2, we have proposed a novel probabilistic model, the mGP, which allows to overcome the limitation of traditional GPs' covariance functions. This model is based on a joint optimization of a latent space transformation (e.g., a non-linear neural network) and a subsequent GPs' covariance function. As a result, we obtain a flexible model capable of learning a family of covariance functions, but which is still a valid GP model. We evaluate the MGP by learning function where traditional GP do not perform well, such as, step functions, multi-length scale functions and forward dynamics model of bipedal walkers in presence of discontinuities (due to contacts with the ground). The experimental results demonstrate the merits of our approach and suggest a novel way to incorporate feature learning techniques into traditional Bayesian models.

In Chapter 3, we have proposed a novel approach to learn inverse dynamics models in presence of contacts by exploiting tactile sensors. Due to the high dimensionality of the raw input from the tactile sensors, we propose the use of a mixture-of-contacts model. A gating network has the purpose of classifying the class of the ongoing contact and activate the corresponding GP model to compensate for it. By decoupling the raw tactile sensors from the dynamics model, it is still possible to learn the inverse dynamics models in presence of contacts without suffering from the curse of dimensionality. The proposed model can exploit the superposition of multiple contacts and does not require spatial calibration of the tactile sensors. We first evaluated this approach on a modeling task using data collected on the humanoid robot iCub, and in presence of ground truth measured from a joint torques sensor. Following, we evaluated our model applied within a controller on a different humanoid robot iCub. The experimental results demonstrate that our approach can accurately estimate the torques exerted by external objects, and therefore can achieve more accurate control. We believe that this approach is a first step towards being able to take a handful of miniaturized tactile sensors, spread them on the structure of the robot, and quickly start learning from them, without having to know where each one of the millions of individual sensors is located.

In Chapter 4, we have proposed the use of Bayesian optimization for efficiently performing black-box optimization of complex robotic systems. Thanks to the use of Bayesian modeling we can fit from past evaluations a probabilistic response surface that predicts the performance for a given set of parameters. We can therefore use an iterative procedure where an appropriate set of parameters to be evaluated on the real robot is selected (by an acquisition function) and subsequently the response surface is updated with the new evaluation. The use of BO in robotics is appealing since it can deal with stochastic noise and does not require the measurement (or numerical approximation) of gradients. Moreover, crucially for real-world robotics applications, BO can obtain good parameters in a small number of evaluations. We validated our approach on the bipedal dynamic walker *Fox* and analyzed the performance of different acquisition functions against standard tuning methods (such as random search and grid search). The results obtained suggests that the GP-UCB acquisition function employ a conservative sampling policy that render it robust to real-world modeling inaccuracies. In our experiments, we successful optimized the gait of the bipedal walker *Fox* in less then 100 evaluations, therefore reducing the tuning time from weeks to less then a single day.

In Chapter 5, we have proposed a novel framework to exploit Bayesian optimization in a multi-objective optimization scenario and to include robustness as one of the objectives. Robustness to perturbations is a critical issue when performing real-world optimizations where both stochastic noise and parameters uncertainty exists. Trough the use of Bayesian modeling, we demonstrate that more accurate and meaningful estimations of the Pareto front can be performed. Additionally, we by introducing novel metrics of robustness, we can estimate the sensitivity to measurement noise and to parameters uncertainty without the need of extensive additional evaluations. We evaluated our framework on a wide range of benchmark test functions, and on a dataset generated from the bipedal walker *Fox*.

6.2 Open Challenges

In this thesis, we have contributed to the state-of-the-art of Bayesian modeling applied to optimization and control of robotic systems. We now detail some of the remaining open challenges that have been identified throughout the PhD, and briefly discuss possible extensions of the presented work and future lines of research that go beyond the current state-of-the-art.

Learning & Exploiting Dynamics Models with Input-dependent Noise

Improving the Quality of the Dynamical Models

The analytical dynamics models employed for robotics are usually deterministic and assume no presence of noise. Although learned dynamics model, such as Gaussian Processes improve by assuming Gaussian noise, this noise is considered homoscedastic (i.e., constant w.r.t. the input space). This assumption does not always hold for real-world application. For example, when performing slow movements the position of the robot can be determined with low uncertainty. However, when performing high speed dynamic movement, the uncertainty regarding the position of the robot will naturally be higher. The use of heteroscedastic models that consider a input-dependent noise (such as the Heteroscedastic Gaussian Processes described in Section 5.3.2) could improve the quality of the learned dynamics model by better matching the real distribution of the noise. We hypothesize that the application of heteroscedastic models in control schema could improve the resulting movement accuracy.

Learning Uncertainty Minimization Policies

Heteroscedastic treatment and estimation of the uncertainty could be use during planning in order select trajectories with low uncertainty, and therefore low “risk”. The same process could be extended not only to select low-uncertainty trajectories, but also to *learn* uncertainty minimization policies. Uncertainty minimization policies are widely used in real-world by humans, for example, by cello players who occasionally reach with their fingers the top of the instrument to “recalibrate” the position of the fingers (w.r.t. a fix reference point), or when navigating in partially unknown environment when people search for known landmarks in order to triangulate their position. In robotics, the “homing” calibration of motors can similarly be considered an uncertainty minimization policy. Learning such minimization policies could reduce the need for a manual design, which typically require extensive expert knowledge, and provide new mechanisms to improve the estimation of the current state of a system

Learning Stiffness Control with Opponent Actuators

One issue that need to be solved before introducing robots in roles interacting with humans is safety. Robot can easily exert forces that might result dangerous or even fatal to humans. For this and other reasons, compliance has in the last years become an important topic in control. The work presented in Chapter 3 is a first step towards the identification of external forces and their compensation trough the use of tactile sensors. However, no direct stiffness/compliance control is considered in the proposed model. Stiffness/compliance of a joint can intuitively be measured as the variance of its state when subject to external perturbations. Whereas stiff joints will not be influenced by the external perturbation, the more compliant a joints will be, the more it will move subject to the perturbation. Trough the use of opponent actuators and heteroscedastic Bayesian models we foresee the possibility of learning a forward dynamic model $\boldsymbol{s}_t, \boldsymbol{a}_t \rightarrow \boldsymbol{s}_{t+1}$ where the variance of the prediction would be directly used as a measure of compliance. The use of such models within a classic control schema (such as, model predictive control) would allow the seamless integration and control of compliance.

High-dimensional Bayesian optimization

In Chapter 4, we discussed the use of Bayesian optimization for automatic optimization of robot parameters. One limitation of Bayesian optimization is its poor scaling capability to increases in the number of parameters. When using traditional Gaussian processes model for the response surface, BO typically becomes impractical with more then 20 parameters. Recent approaches Djolonga et al. [2013], Garnett et al. [2013], Wang et al. [2013] extend BO to higher dimensionality of parameters by presuppose the existence of lower-dimensional embedding. These approaches however are limited either in the class of embeddings (e.g., linear), or require an unfeasible number of evaluations to be applied in real-world robotics problems. Future work should aim to improve the data efficiency of the current non-linear embeddings and study novel models capable of better scaling with the number of parameters.

Multi-objective optimization for Machine Learning

In Chapter 5 we discussed robust MOBO. Outside robotics, one application where the use of MOO would be beneficial is automatic tuning of machine learning algorithms. While there is a wide use of single-objective automatic tuning in ML Hutter et al. [2011], Snoek et al. [2012b], the use of multiple objective is rare. Nonetheless, it is clear that even in ML there exists natural trade-off, computational time vs precision, training accuracy vs generalization, supervised objectives vs unsupervised (e.g., regularization). The use of an appropriate MOO framework could naturally extend the SOO and provide more flexibility during the training of such models.

Bayesian modeling for High-dimensional Reinforcement Learning

A common approach to learn to perform new tasks with robots is Reinforcement Learning (RL) [Deisenroth et al., 2013]. While in this thesis we do not consider the RL framework, the Bayesian methodologies can be applied to RL, as demonstrated by Deisenroth et al. [2015]. However, the current Bayesian modeling methods do not scale well with the increase in the number of input covariates (due to the curse of dimensionality Donoho [2000]). Therefore, to apply Bayesian modeling in high-dimensional RL it would be necessary to improve the current methods. Moreover, the model-based RL typically requires the capability of making highly accurate long-term prediction about the effects of the actions performed. The capability of current methods is often not yet sufficiently accurate and, hence, would be an interesting future topic of research.

Learning Multi-step and Partially Observable Forward Dynamics

In many control schema Camacho and Alba [2013], Deisenroth et al. [2015], it is required the existence of accurate dynamical models. This dynamical models are often assumed to strictly belong to a Markov Decision Process (MDP) framework, where the current state and current action fully determine (a probability distribution of) the future state in a single-step look-ahead 1st order MDP $\boldsymbol{s}_t, \boldsymbol{a}_t \rightarrow \boldsymbol{s}_{t+1}$. While useful to greatly simply the model used, this assumption does not always hold true. Future work should focus on extending learning dynamical models to non-MDP settings. A first case is the study of the propagation of the dynamics of single-step ahead to compose n-step ahead predictions (i.e., where we try to predict what will happen after multiple units of time) or alternatively the creation of models capable of directly estimating the state given a sequence of control signals $\boldsymbol{s}_t, \boldsymbol{a}_t, \cdots, \boldsymbol{a}_{t+n-1} \rightarrow \boldsymbol{s}_{t+n}$. A second case is the creation of dynamical models in presence of Partially Observable MDP (POMDPs) where the state is not fully observable, and therefore it has to be estimated. POMDPs are often approximated by using higher-orders MDP where the last n states/actions are used in order to make a prediction $\boldsymbol{s}_{t-n}, \boldsymbol{a}_{t-n}, \cdots, \boldsymbol{s}_t, \boldsymbol{a}_t \rightarrow \boldsymbol{s}_{t+1}$. However, the use of auto-regressive models or of memory-based models seems an appealing alternative that would avoid the increase in dimensionality resulting from expanding the input space using higher-order MDP.

6.3 Publications

The work presented in this thesis contributed to the following publications:

Journal Papers

- Roberto Calandra, Felix Unverzagt, Jan Peters, and Marc. P. Deisenroth. Robust multi-objective Bayesian optimization. submitted to the Journal of Machine Learning Research (JMLR)
- Roberto Calandra, Serena Ivaldi, Marc. P. Deisenroth, and Jan Peters. Learning from artificial skin: Torque control in presence of contacts using high-dimensional tactile sensors. submitted to the IEEE Transactions on Robotics
- Roberto Calandra, André Seyfarth, Jan Peters, and Marc P. Deisenroth. Bayesian optimization for learning gaits under uncertainty. *Annals of Mathematics and Artificial Intelligence (AMAI)*, 76(1):5–23, 2015d. ISSN 1573-7470. doi:10.1007/s10472-015-9463-9

Conference Papers

- Roberto Calandra, Jan Peters, Carl E. Rasmussen, and Marc P. Deisenroth. Manifold Gaussian Processes for Regression. In *International Joint Conference on Neural Networks (IJCNN)*, pages 3338–3345, 2016
- P. Weber, E. Rueckert, Roberto Calandra, Jan Peters, and P. Beckerle. A low-cost sensor glove with vibrotactile feedback and multiple finger joint and hand motion sensing for human-robot interaction. In *IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2016
- Zhengkun Yi, Roberto Calandra, Filipe Fernandes Veiga, Herke van Hoof, Tucker Hermans, Yilei Zhang, and Jan Peters. Active tactile object exploration with gaussian processes. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016
- Roberto Calandra, Serena Ivaldi, Marc P. Deisenroth, and Jan Peters. Learning torque control in presence of contacts using tactile sensing from robot skin. In *IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS)*, pages 690–695, Nov 2015a. doi:10.1109/HUMANOIDS.2015.7363429
- Roberto Calandra, Serena Ivaldi, Marc. P. Deisenroth, E. Rueckert, and Jan Peters. Learning inverse dynamics models with contacts. In *International Conference on Robotics and Automation (ICRA)*, pages 3186–3191. IEEE, May 2015b. doi:10.1109/ICRA.2015.7139638
- Lars Fritsche, Felix Unverzagt, Jan Peters, and Roberto Calandra. First-person tele-operation of a humanoid robot. In *IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS)*, pages 997–1002, November 2015. doi: 10.1109/HUMANOIDS.2015.7363475
- Roberto Calandra, Nakul Gopalan, André Seyfarth, Jan Peters, and Marc Peter Deisenroth. Bayesian gait optimization for bipedal locomotion. In *Learning and Intelligent OptimizatioN Conference (LION)*, pages 274–290, 2014a. ISBN 978-3-319-09584-4. doi:10.1007/978-3-319-09584-4_25
- Roberto Calandra, André Seyfarth, Jan Peters, and Marc Peter Deisenroth. An experimental comparison of Bayesian optimization for bipedal locomotion. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1951–1958, May 2014c. doi:10.1109/ICRA.2014.6907117
- Marc Peter Deisenroth, Roberto Calandra, André Seyfarth, and Jan Peters. Toward fast policy search for learning legged locomotion. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 1787–1792, Oct 2012. doi:10.1109/IROS.2012.6385955
- Roberto Calandra, Tapani Raiko, Marc Peter Deisenroth, and Federico Montesino Pouzols. Learning deep belief networks from non-stationary streams. In *International Conference on Artificial Neural Networks (ICANN)*, pages 379–386, 2012. doi:10.1007/978-3-642-33266-1_47

Workshop Papers

- Roberto Calandra, Serena Ivaldi, Marc. P. Deisenroth, Elmar Rueckert, and Jan Peters. Learning inverse dynamics models with contacts using tactile sensors. ICRA2015 Workshop on Tactile & force sensing for autonomous, compliant, intelligent robots, 2015c
- Elmar Rueckert, R. Lioutikov, Roberto Calandra, M. Schmidt, P. Beckerle, and Jan Peters. Low-cost sensor glove with force feedback for learning from demonstrations using probabilistic trajectory representations. ICRA2015 Workshop on Tactile & force sensing for autonomous, compliant, intelligent robots, 2015
- Roberto Calandra, Jan Peters, and Marc P. Deisenroth. Pareto front modeling for sensitivity analysis in multi-objective bayesian optimization. NIPS Workshop on Bayesian Optimization (BayesOpt), 2014b. URL `http://www.ias.tu-darmstadt.de/uploads/Publications/Calandra-NIPS2015-bayesopt.pdf`
- Katayon Radkhah, Roberto Calandra, and Marc P. Deisenroth. Learning musculoskeletal dynamics with non-parametric models. ICRA2013 Workshop on Novel Methods for Learning and Optimization of Control Policies and Trajectories for Robotics, 2013. URL `http://www.ias.tu-darmstadt.de/uploads/Research/ICRA2013/Radkah.pdf`
- Roberto Calandra, Jan Peters, Andre Seyfarth, and Marc Deisenroth. An experimental evaluation of bayesian optimization on bipedal locomotion. NIPS Workshop on Bayesian Optimization (BayesOpt), 2013

Bibliography

Gautam Agrawal, Kemper Lewis, K Chugh, CH Huang, S Parashar, and CL Bloebaum. Intuitive visualization of pareto frontier for multi-objective optimization in n-dimensional performance space. In *10th AIAA/ISSMO multidisciplinary analysis and optimization conference*, pages 1523–1533, 2004.

M. Farooq Anjum, Imran Tasadduq, and Khaled Al-Sultan. Response surface methodology: A neural network approach. *European Journal of Operational Research*, 101(1):65–73, 1997. ISSN 0377-2217. doi:10.1016/S0377-2217(96)00232-9.

Ryo Ariizumi, Matthew Tesch, Howie Choset, and Fumitoshi Matsuno. Expensive multiobjective optimization for robotics with consideration of heteroscedastic noise. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2230–2235, 2014. doi:10.1109/IROS.2014.6942863.

John-Alexander M. Assael, Ziyu Wang, Bobak Shahriari, and Nando de Freitas. Heteroscedastic treed bayesian optimisation. *NIPS Workshop on Bayesian Optimization (BayesOpt)*, 2014.

A. C. Atkinson. The usefulness of optimum experimental designs. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):59–76, 1996. ISSN 00359246. doi:10.2307/2346165.

Anne Auger, Johannes Bader, Dimo Brockhoff, and Eckart Zitzler. Hypervolume-based multiobjective optimization: Theoretical foundations and practical implications. *Theoretical Computer Science*, 425:75–103, March 2012. ISSN 0304-3975. doi:10.1016/j.tcs.2011.03.012.

Morteza Azad and Michael N. Mistry. Balance control strategy for legged robots with compliant contacts. In *International Conference on Robotics and Automation (ICRA)*, pages 4391–4396. IEEE, 2015.

Johannes Bader and Eckart Zitzler. Robustness in hypervolume-based multiobjective search. Technical Report TIK Report 317, Computer Engineering and Networks Laboratory, ETH Zurich, 2010.

Thomas Bartz-Beielstein, Christian WG Lasarczyk, and Mike Preuß. Sequential parameter optimization. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 1, pages 773–780. IEEE, 2005.

Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust optimization*. Princeton University Press, 2009.

Florian Bergner, Philipp Mittendorfer, Emmanuel Dean-Leon, and Gordon Cheng. Event-based signaling for reducing required data rates and processing power in a large-scale artificial robotic skin. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 2124 – 2129, 2015.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research (JMLR)*, 13:281–305, 2012.

Felix Berkenkamp, Andreas Krause, and Angela P. Schoellig. Bayesian optimization with safety constraints: Safe and automatic parameter tuning in robotics. *ArXiv e-prints*, February 2016. URL `http://arxiv.org/abs/1602.04450`.

Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 3rd edition, 2007. ISBN 1886529302, 9781886529304.

Dimitris Bertsimas, David B Brown, and Constantine Caramanis. Theory and applications of robust optimization. *SIAM review*, 53(3):464–501, 2011.

Michael J. Best and Robert R. Grauer. On the sensitivity of mean-variance-efficient portfolios to changes in asset means: Some analytical and computational results. *Review of Financial Studies*, 4(2):315–342, 1991. doi:10.1093/rfs/4.2.315.

N. Beume, C. M. Fonseca, M. Lopez-Ibanez, L. Paquete, and J. Vahrenhold. On the complexity of computing the hypervolume indicator. *IEEE Transactions on Evolutionary Computation*, 13(5):1075–1082, Oct 2009. ISSN 1089-778X. doi:10.1109/TEVC.2009.2015575.

Hans-Georg Beyer and Bernhard Sendhoff. Robust optimization–a comprehensive survey. *Computer methods in applied mechanics and engineering*, 196(33):3190–3218, 2007.

Mickael Binois, David Ginsbourger, and Olivier Roustant. Quantifying uncertainty on Pareto fronts with Gaussian process conditional simulations. In *Learning and Intelligent OptimizatioN Conference (LION8)*, 2014.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006.

X Blasco, JM Herrero, Javier Sanchis, and Manuel Martínez. A new graphical visualization of n-dimensional pareto front for decision-making in multiobjective optimization. *Information Sciences*, 178(20):3908–3924, 2008.

Emanuele Borgonovo and Elmar Plischke. Sensitivity analysis: A review of recent advances. *European Journal of Operational Research*, 248(3):869–887, 2016. ISSN 0377-2217. doi:http://dx.doi.org/10.1016/j.ejor.2015.06.032. URL `http://www.sciencedirect.com/science/article/pii/S0377221715005469`.

Jürgen Branke, Kalyanmoy Deb, Kaisa Miettinen, and Roman Slowinski, editors. *Multiobjective Optimization - Interactive and Evolutionary Approaches*, volume 5252 of *Lecture Notes in Computer Science*, 2008. Springer. ISBN 978-3-540-88907-6. doi:10.1007/978-3-540-88908-3.

Eric Brochu, Vlad M. Cora, and Nando De Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

Samuel H. Brooks. A discussion of random methods for seeking maxima. *Operations Research*, 6(2):244–251, 1958.

D. Buche, P. Stoll, R. Dornberger, and P. Koumoutsakos. Multiobjective evolutionary algorithm for the optimization of noisy combustion processes. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 32 (4):460–473, Nov 2002. ISSN 1094-6977. doi:10.1109/TSMCB.2002.804372.

Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

R. Calandra, J. Peters, C. E. Rasmussen, and M. P. Deisenroth. Manifold Gaussian processes for regression. *ArXiv e-prints*, February 2014.

Roberto Calandra, Tapani Raiko, Marc Peter Deisenroth, and Federico Montesino Pouzols. Learning deep belief networks from non-stationary streams. In *International Conference on Artificial Neural Networks (ICANN)*, pages 379–386, 2012. doi:10.1007/978-3-642-33266-1_47.

Roberto Calandra, Jan Peters, Andre Seyfarth, and Marc Deisenroth. An experimental evaluation of bayesian optimization on bipedal locomotion. NIPS Workshop on Bayesian Optimization (BayesOpt), 2013.

Roberto Calandra, Nakul Gopalan, André Seyfarth, Jan Peters, and Marc Peter Deisenroth. Bayesian gait optimization for bipedal locomotion. In *Learning and Intelligent OptimizatioN Conference (LION)*, pages 274–290, 2014a. ISBN 978-3-319-09584-4. doi:10.1007/978-3-319-09584-4_25.

Roberto Calandra, Jan Peters, and Marc P. Deisenroth. Pareto front modeling for sensitivity analysis in multi-objective bayesian optimization. NIPS Workshop on Bayesian Optimization (BayesOpt), 2014b. URL `http://www.ias.tu-darmstadt.de/uploads/Publications/Calandra-NIPS2015-bayesopt.pdf`.

Roberto Calandra, André Seyfarth, Jan Peters, and Marc Peter Deisenroth. An experimental comparison of Bayesian optimization for bipedal locomotion. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1951–1958, May 2014c. doi:10.1109/ICRA.2014.6907117.

Roberto Calandra, Serena Ivaldi, Marc P. Deisenroth, and Jan Peters. Learning torque control in presence of contacts using tactile sensing from robot skin. In *IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS)*, pages 690–695, Nov 2015a. doi:10.1109/HUMANOIDS.2015.7363429.

Roberto Calandra, Serena Ivaldi, Marc. P. Deisenroth, E. Rueckert, and Jan Peters. Learning inverse dynamics models with contacts. In *International Conference on Robotics and Automation (ICRA)*, pages 3186–3191. IEEE, May 2015b. doi:10.1109/ICRA.2015.7139638.

Roberto Calandra, Serena Ivaldi, Marc. P. Deisenroth, Elmar Rueckert, and Jan Peters. Learning inverse dynamics models with contacts using tactile sensors. ICRA2015 Workshop on Tactile & force sensing for autonomous, compliant, intelligent robots, 2015c.

Roberto Calandra, André Seyfarth, Jan Peters, and Marc P. Deisenroth. Bayesian optimization for learning gaits under uncertainty. *Annals of Mathematics and Artificial Intelligence (AMAI)*, 76(1):5–23, 2015d. ISSN 1573-7470. doi: 10.1007/s10472-015-9463-9.

Roberto Calandra, Jan Peters, Carl E. Rasmussen, and Marc P. Deisenroth. Manifold Gaussian Processes for Regression. In *International Joint Conference on Neural Networks (IJCNN)*, pages 3338–3345, 2016.

Roberto Calandra, Serena Ivaldi, Marc. P. Deisenroth, and Jan Peters. Learning from artificial skin: Torque control in presence of contacts using high-dimensional tactile sensors. submitted to the IEEE Transactions on Robotics.

Roberto Calandra, Felix Unverzagt, Jan Peters, and Marc. P. Deisenroth. Robust multi-objective Bayesian optimization. submitted to the Journal of Machine Learning Research (JMLR).

Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer Science & Business Media, 2013.

Giorgio Cannata, M. Maggiali, G. Metta, and G. Sandini. An embedded artificial skin for humanoid robots. In *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI).*, pages 434–438, Aug 2008. doi:10.1109/MFI.2008.4648033.

Giorgio Cannata, S. Denei, and F. Mastrogiovanni. Towards automated self-calibration of robot skin. In *International Conference on Robotics and Automation (ICRA)*, pages 4849–4854, 2010.

G. Capi, M. Yokota, and K. Mitobe. A new humanoid robot gait generation based on multiobjective optimization. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 450–454, 2005. doi:10.1109/AIM.2005.1501032.

Constantine Caramanis, Shie Mannor, and Huan Xu. *Optimization for machine learning*, chapter Robust Optimization in Machine Learning, pages 369–402. Mit Press, 2012.

Sonia Chernova and Manuela Veloso. An evolutionary approach to gait learning for four-legged robots. In *International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 2562–2567. IEEE, 2004.

David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 1996.

Yann Collette and Patrick Siarry. *Multiobjective optimization: principles and case studies*. Springer Science & Business Media, 2004. doi:10.1007/978-3-662-08883-8.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995. ISSN 1573-0565. doi:10.1007/BF00994018. URL `http://dx.doi.org/10.1007/BF00994018`.

Ivo Couckuyt, Dirk Deschrijver, and Tom Dhaene. Fast calculation of multiobjective probability of improvement and expected improvement criteria for pareto optimization. *Journal of Global Optimization*, 60(3):575–594, 2014.

Dennis D. Cox and Susan John. SDO: A statistical method for global optimization. *Multidisciplinary design optimization: state of the art*, pages 315–329, 1997.

R.S. Dahiya, G. Metta, M. Valle, and G. Sandini. Tactile sensing – from humans to humanoids. *IEEE Transactions on Robotics*, 26(1):1–20, Feb 2010. ISSN 1552-3098. doi:10.1109/TRO.2009.2033627.

R.S. Dahiya, P. Mittendorfer, M. Valle, G. Cheng, and V.J. Lumelsky. Directions toward effective utilization of tactile skin: A review. *IEEE Sensors Journal*, 13(11):4121–4138, Nov 2013. ISSN 1530-437X. doi:10.1109/JSEN.2013.2279056.

Andreas C. Damianou and Neil D. Lawrence. Deep Gaussian Processes. *ArXiv e-prints*, November 2013.

I. Das and J.E. Dennis. A closer look at drawbacks of minimizing weighted sums of objectives for pareto set generation in multicriteria optimization problems. *Structural optimization*, 14(1):63–69, 1997. ISSN 0934-4373. doi:10.1007/BF01197559.

Kalyanmoy Deb. *Multi-objective optimization using evolutionary algorithms*, volume 16. John Wiley & Sons, 2001.

Kalyanmoy Deb and Himanshu Gupta. Introducing robustness in multi-objective optimization. *Evolutionary Computation*, 14(4):463–494, December 2006. ISSN 1063-6560. doi:10.1162/evco.2006.14.4.463. URL `http://dx.doi.org/10.1162/evco.2006.14.4.463`.

Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and Tamt Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002a.

Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. Scalable multi-objective optimization test problems. In *Congress on Evolutionary Computation (CEC)*, pages 825–830, 2002b.

Marc Peter Deisenroth, Roberto Calandra, André Seyfarth, and Jan Peters. Toward fast policy search for learning legged locomotion. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 1787–1792, Oct 2012. doi: 10.1109/IROS.2012.6385955.

Marc Peter Deisenroth, Gerhard Neumann, and Jan Peters. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2(1-2):1–142, 2013. doi:10.1561/2300000021.

M.P. Deisenroth, D. Fox, and C. Rasmussen. Gaussian processes for data-efficient learning in robotics and control. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 37(2):408–423, 2015. ISSN 0162-8828. doi:10.1109/TPAMI.2013.218.

Andrea Del Prete, Simone Denei, Lorenzo Natale, Fulvio Mastrogiovanni, Francesco Nori, Giorgio Cannata, and Giorgio Metta. Skin spatial calibration using force/torque measurements. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 3694–3700. IEEE, 2011.

Andrea Del Prete, Francesco Nori, Giorgio Metta, and Lorenzo Natale. Control of contact forces: The role of tactile feedback for contact localization. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 4048–4053. IEEE, 2012.

JP Delhomme. Kriging in the hydrosciences. *Advances in water resources*, 1(5):251–266, 1978.

Josip Djolonga, Andreas Krause, and Volkan Cevher. High-dimensional gaussian process bandits. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1025–1033. 2013. URL `http://media.nips.cc/nipsbooks/nipspapers/paper_files/nips26/548.pdf`.

David L. Donoho. High-Dimensional Data Analysis: The Curses and Blessings of Dimensionality, 2000.

David Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Structure Discovery in Nonparametric Regression through Compositional Kernel Search. In *Proceedings of the 30th International Conference on Machine Learning*, June 2013.

M. T. M. Emmerich, A. H. Deutz, and J. W. Klinkenberg. Hypervolume-based expected improvement: Monotonicity properties and exact computation. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 2147–2154, June 2011. doi:10.1109/CEC.2011.5949880.

Michael Emmerich, André Deutz, and Jan-Willem Klinkenberg. The computation of the expected improvement in dominated hypervolume of Pareto front approximations. Technical Report LIACS-TR 9-2008, Leiden Institute for Advanced Computer Science, September 2008. URL `http://www.liacs.nl/~emmerich/TR-ExI.pdf`.

Peter Englert and Marc Toussaint. Combined optimization and reinforcement learning for manipulation skills. In *Robotics: Science and Systems (RSS)*, 2016.

Tom Erez and Emanuel Todorov. Trajectory optimization for domains with contacts using inverse dynamics. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 4914–4919. IEEE, 2012.

Hamidreza Eskandari and Christopher D. Geiger. Evolutionary multiobjective optimization in noisy problem environments. *Journal of Heuristics*, 15(6):559–595, 2008. ISSN 1572-9397. doi:10.1007/s10732-008-9077-z.

Marco Farina and Paolo Amato. On the optimal solution definition for many-criteria optimization problems. In *Proceedings of the NAFIPS-FLINT international conference*, pages 233–238, 2002.

Roy Featherstone and D.E. Orin. Dynamics. In *Springer Handbook of Robotics*, pages 35–65. 2008. doi:10.1007/978-3-540-30301-5_3.

J. E. Fieldsend and R. M. Everson. Multi-objective optimisation in the presence of uncertainty. In *IEEE Congress on Evolutionary Computation (CEC)*, volume 1, pages 243–250, Sept 2005. doi:10.1109/CEC.2005.1554691.

Peter J Fleming, Robin C Purshouse, and Robert J Lygoe. Many-objective optimization: An engineering design perspective. In *Evolutionary multi-criterion optimization*, pages 14–32. Springer, 2005.

Lars Fritsche, Felix Unverzagt, Jan Peters, and Roberto Calandra. First-person tele-operation of a humanoid robot. In *IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS)*, pages 997–1002, November 2015. doi: 10.1109/HUMANOIDS.2015.7363475.

M. Fumagalli, A. Gijsberts, S. Ivaldi, L. Jamone, G. Metta, L. Natale, F. Nori, and G. Sandini. Learning how to exploit proximal force sensing: a comparison approach. In *From Motor Learning to Interaction Learning in Robots*, volume 264 of *Studies in Computational Intelligence*, pages 159–177. Springer-Verlag, 2010.

M. Fumagalli, S. Ivaldi, M. Randazzo, L. Natale, G. Metta, G. Sandini, and F. Nori. Force feedback exploiting tactile and proximal force/torque sensing - Theory and implementation on the humanoid robot iCub. *Autonomous Robots*, 33(4): 381–398, 2012.

R. Garnett, M. A. Osborne, and S. J. Roberts. Bayesian optimization for sensor set selection. In *ACM/IEEE International Conference on Information Processing in Sensor Networks*, IPSN '10, pages 209–219. ACM, 2010. ISBN 978-1-60558-988-6. doi:10.1145/1791212.1791238.

Roman Garnett, Michael A. Osborne, and Philipp Hennig. Active learning of linear embeddings for Gaussian processes. *arXiv preprint arXiv:1310.6740*, 2013.

António Gaspar-Cunha and José A Covas. Robustness in multi-objective optimization using evolutionary algorithms. *Computational Optimization and Applications*, 39(1):75–96, 2008.

Tao Geng, Bernd Porr, and Florentin Wörgötter. Fast biped walking with a sensor-driven neuronal controller and real-time online learning. *The International Journal of Robotics Research*, 25(3):243–259, 2006.

Peter Gibbons, Martin Mason, Alexandre Vicente, Guido Bugmann, and Phil Culverhouse. Optimisation of dynamic gait for small bipedal robots. In *Proc. 4th Workshop on Humanoid Soccer Robots. A workshop of the 2009 IEEE-RAS Intl. Conf. On Humanoid Robots (Humanoids 2009), Paris (France)*, pages 9–14, 2009.

Paul W Goldberg, Christopher KI Williams, and Christopher M Bishop. Regression with input-dependent noise: A Gaussian process treatment. *Advances in neural information processing systems (NIPS)*, 10:493–499, 1997.

Nooshin HajiGhassemi and Marc P. Deisenroth. Approximate Inference for Long-Term Forecasting with Periodic Gaussian Processes. In *AISTATS*, 2014.

A. Hamann, R. Racu, and R. Ernst. Multi-dimensional robustness optimization in heterogeneous distributed embedded systems. In *13th IEEE Real Time and Embedded Technology and Applications Symposium (RTAS'07)*, pages 269–280, April 2007. doi:10.1109/RTAS.2007.19.

Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, 9(2):159–195, 2001.

Markus Hartikainen, Kaisa Miettinen, and Margaret M. Wiecek. Paint: Pareto front interpolation for nonlinear multi-objective optimization. *Computational Optimization and Applications*, 52(3):845–867, 2011. ISSN 1573-2894. doi: 10.1007/s10589-011-9441-z. URL `http://dx.doi.org/10.1007/s10589-011-9441-z`.

Thomas Hemker, Maximilian Stelzer, Oskar von Stryk, and Hajime Sakamoto. Efficient walking speed optimization of a humanoid robot. *International Journal of Robotics Research (IJRR)*, 28(2):303–314, 2009. doi:10.1177/0278364908095171.

Philipp Hennig and Christian J. Schuler. Entropy search for information-efficient global optimization. *Journal of Machine Learning Research (JMLR)*, 13:1809–1837, 2012.

Daniel Hernández-Lobato, José Miguel Hernández-Lobato, Amar Shah, and Ryan P Adams. Predictive entropy search for multi-objective bayesian optimization. In *NIPS workshop on Bayesian Optimization (BayesOpt)*, 2015.

José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *Advances in Neural Information Processing Systems (NIPS)*, pages 918–926, 2014.

Matthew D Hoffman, Eric Brochu, and Nando de Freitas. Portfolio allocation for bayesian optimization. In *UAI*, pages 327–336, 2011.

Daniel Horn, Tobias Wagner, Dirk Biermann, Claus Weihs, and Bernd Bischl. *Evolutionary Multi-Criterion Optimization: 8th International Conference, EMO 2015, Guimarães, Portugal, March 29 –April 1, 2015. Proceedings, Part I*, chapter Model-Based Multi-objective Optimization: Taxonomy, Multi-Point Proposal, Toolbox and Benchmark, pages 64–78. Springer International Publishing, Cham, 2015. ISBN 978-3-319-15934-8. doi:10.1007/978-3-319-15934-8_5. URL `http://dx.doi.org/10.1007/978-3-319-15934-8_5`.

Deng Huang, Theodore T Allen, William I Notz, and N Zeng. Global optimization of stochastic black-box systems via sequential kriging meta-models. *Journal of Global Optimization*, 34(3):441–466, 2006.

Evan J. Hughes. Evolutionary multi-objective ranking with uncertainty and noise. In *Evolutionary multi-criterion optimization*, pages 329–343. Springer, 2001.

Iris Hupkens, André Deutz, Kaifeng Yang, and Michael Emmerich. *Evolutionary Multi-Criterion Optimization: 8th International Conference, EMO 2015, Guimarães, Portugal, March 29 –April 1, 2015. Proceedings, Part II*, chapter Faster Exact Algorithms for Computing Expected Hypervolume Improvement, pages 65–79. Springer International Publishing, Cham, 2015. ISBN 978-3-319-15892-1. doi:10.1007/978-3-319-15892-1_5. URL `http://dx.doi.org/10.1007/978-3-319-15892-1_5`.

Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization (LION)*, pages 507–523. Springer, 2011.

Aapo Hyvärinen and Erkki Oja. Independent Component Analysis: Algorithms and Applications. *Neural networks*, 13(4): 411–430, 2000.

Ronald L. Iman and Jon C. Helton. An investigation of uncertainty and sensitivity analysis techniques for computer models. *Risk analysis*, 8(1):71–90, 1988.

Ronald L. Iman, Jon C. Helson, and James E. Campbell. An approach to sensitivity analysis of computer models: Part ii-ranking of input variables, response surface validation, distribution effect and technique synopsis. *Journal of Quality Technology*, 13(4), 1981.

Hisao Ishibuchi, Noritaka Tsukamoto, and Yusuke Nojima. Evolutionary many-objective optimization: A short review. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 2419–2426, 2008.

S. Ivaldi, M. Fumagalli, M. Randazzo, F. Nori, G. Metta, and G. Sandini. Computing robot internal/external wrenches by means of inertial, tactile and F/T sensors: theory and implementation on the iCub. In *International Conference on Humanoid Robots (HUMANOIDS)*, pages 521–528, 2011.

S. Ivaldi, J. Peters, V. Padois, and F. Nori. Tools for simulating humanoid robot dynamics: a survey based on user feedback. In *International Conference on Humanoid Robots (HUMANOIDS)*, 2014.

Serena Ivaldi, Jan Babič, Michael Mistry, and Robin Murphy. Special issue on whole-body control of contacts and dynamics for humanoid robots. *Autonomous Robots*, pages 1–4, 2016. ISSN 1573-7527. doi:10.1007/s10514-016-9545-5. URL `http://dx.doi.org/10.1007/s10514-016-9545-5`.

A. Jain, M.D. Killpack, A. Edsinger, and C.C. Kemp. Reaching in clutter with whole-arm tactile sensing. *International Journal of Robotics Research (IJRR)*, 32(4):458–482, 2013.

Lorenzo Jamone, Bruno Damas, and José Santos-Victor. Incremental learning of context-dependent dynamic internal models for robot control. In *IEEE International Symposium on Intelligent Control (ISIC)*, pages 1336–1341. IEEE, 2014.

Yaochu Jin, editor. *Multi-Objective Machine Learning*, volume 16 of *Studies in Computational Intelligence*. Springer, 2006. ISBN 978-3-540-30676-4.

Yaochu Jin and Jürgen Branke. Evolutionary optimization in uncertain environments-a survey. *IEEE Transactions on Evolutionary Computation*, 9(3):303–317, 2005.

Yaochu Jin and Bernhard Sendhoff. Trade-off between performance and robustness: an evolutionary multiobjective approach. In *Evolutionary Multi-Criterion Optimization*, pages 237–251. Springer, 2003.

Donald R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21(4):345–383, 2001.

Donald R. Jones, Cary D. Perttunen, and Bruce E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, 1993.

Donald R. Jones, Matthias Schonlau, and William J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, 1998.

Andy J. Keane. Statistical improvement criteria for use in multiobjective design optimization. *AIAA journal*, 44(4): 879–891, 2006.

Kristian Kersting, Christian Plagemann, Patrick Pfaff, and Wolfram Burgard. Most likely heteroscedastic Gaussian process regression. In *International Conference on Machine Learning (ICML)*, pages 393–400. ACM, 2007.

Joshua Knowles. ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66, January 2006.

S. Kolev and E. Todorov. Physically consistent state estimation and system identification for contacts. In *International Conference on Humanoid Robots (HUMANOIDS)*, 2015.

Scott R. Kuindersma, Roderic A. Grupen, and Andrew G. Barto. Variable risk control via stochastic optimization. *The International Journal of Robotics Research (IJRR)*, 32(7):806–825, 2013.

Saku Kukkonen and Jouni Lampinen. Ranking-dominance and many-objective optimization. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 3983–3990. IEEE, 2007.

J. Kulk and J.S. Welsh. Evaluation of walk optimisation techniques for the NAO robot. In *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*, pages 306–311, Oct 2011. doi:10.1109/Humanoids.2011.6100827.

Maciej Kurek, Wayne Luk, Timothy Todman, and Marc P. Deisenroth. Knowledge transfer in automatic optimisation of reconfigurable designs. In *Proceedings of the IEEE International Symposium on Field-Programmable Custom Computing Machines*, 2016.

Harold J. Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 86:97, 1964.

Neil D. Lawrence. Probabilistic Non-linear Principal Component Analysis with Gaussian Process Latent Variable Models. *Journal of Machine Learning Research (JMLR)*, 6:1783–1816, November 2005. URL `http://www.jmlr.org/papers/volume6/lawrence05a/lawrence05a.pdf`.

Miguel Lázaro-Gredilla and Michalis Titsias. Variational heteroscedastic Gaussian process regression. In *International Conference on Machine Learning (ICML)*, pages 841–848, New York, NY, USA, June 2011. ACM. ISBN 978-1-4503-0619-5.

Ikjin Lee, KK Choi, and Liang Zhao. Sampling-based rbdo using the stochastic sensitivity analysis and dynamic kriging method. *Structural and Multidisciplinary Optimization*, 44(3):299–317, 2011.

Dong C. Liu and Jorge Nocedal. On the Limited Memory BFGS Method for Large Scale Optimization. *Mathematical Programming*, 45(3):503–528, 1989.

Daniel J. Lizotte, Tao Wang, Michael Bowling, and Dale Schuurmans. Automatic gait optimization with Gaussian process regression. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 944–949, 2007.

Daniel J. Lizotte, Russell Greiner, and Dale Schuurmans. An experimental methodology for response surface optimization methods. *Journal of Global Optimization*, 53(4):699–736, 2012. doi:10.1007/s10898-011-9732-z.

David G. Luenberger. *Investment Science*. Oxford University Press, 1998. ISBN 9780195108095.

Jan Marian Maciejowski. *Predictive control: with constraints*. Pearson education, 2002.

David J. C. MacKay. Introduction to Gaussian Processes. In *Neural Networks and Machine Learning*, volume 168, pages 133–165. 1998.

R. Timothy Marler and Jasbir S. Arora. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395, 2004. ISSN 1615-147X. doi:10.1007/s00158-003-0368-6.

R. Timothy Marler and Jasbir S. Arora. The weighted sum method for multi-objective optimization: new insights. *Structural and Multidisciplinary Optimization*, 41(6):853–862, 2009. ISSN 1615-1488. doi:10.1007/s00158-009-0460-7. URL `http://dx.doi.org/10.1007/s00158-009-0460-7`.

James Martens. Deep learning via hessian-free optimization. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, volume 951, page 2010, 2010.

Andrew McHutchon and Carl E. Rasmussen. Gaussian process training with input noise. In *Advances in Neural Information Processing Systems (NIPS)*. 2011.

Michael D. McKay. Latin hypercube sampling as a tool in uncertainty analysis of computer models. In *Proceedings of the 24th Conference on Winter Simulation*, WSC '92, pages 557–564. ACM, 1992. ISBN 0-7803-0798-4. doi:10.1145/167293.167637.

A. Messac, J. E. Renaud, G. J. Sundararaj, and R. V. Tappeta. Ability of objective functions to generate points on nonconvex Pareto frontiers. *AIAA Journal*, 38:1084–1091, June 2000. doi:10.2514/2.1071.

Kaisa Miettinen, Kalyanmoy Deb, Johannes Jahn, Wlodzimierz Ogryczak, Koji Shimoyama, and Rudolf Vetschera. Future challenges. In Branke et al. [2008], pages 435–461. ISBN 978-3-540-88907-6. doi:10.1007/978-3-540-88908-3.

P. Mittendorfer, E. Yoshida, and G. Cheng. Realizing whole-body tactile interactions with a self-organizing, multi-modal artificial skin on a humanoid robot. *Advanced Robotics*, 29(1):51–67, 2015.

Philipp Mittendorfer and Gordon Cheng. Integrating discrete force cells into multi-modal artificial skin. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 847–852. IEEE, 2012.

Miha Mlakar, Tea Tušar, and Bogdan Filipič. Comparing solutions under uncertainty in multiobjective optimization. *Mathematical Problems in Engineering*, 2014, 2014.

Jonas Mockus, V. Tiesis, and A. Zilinskas. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization*, 2:117–129, 1978.

de Renato S. Motta and Silvana M. B. Afonso. An efficient procedure for structural reliability-based robust design optimization. *Structural and Multidisciplinary Optimization*, pages 1–20, 2016. ISSN 1615-1488. doi:10.1007/s00158-016-1418-1.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted Boltzmann machines. In *International Conference on Machine Learning (ICML)*, pages 807–814, 2010.

Jun Nakanishi, Rick Cory, Michael Mistry, Jan Peters, and Stefan Schaal. Operational space control: A theoretical and empirical comparison. *International Journal of Robotics Research (IJRR)*, 27(6):737–757, 2008. doi:10.1177/0278364908091463. URL `http://dx.doi.org/10.1177/0278364908091463`.

L. Natale, F. Nori, G. Metta, M. Fumagalli, S. Ivaldi, U. Pattacini, M. Randazzo, A. Schmitz, and G. G. Sandini. The iCub platform: a tool for studying intrinsically motivated learning. In *Intrinsically motivated learning in natural and artificial systems*, pages 433–458. Springer-Verlag, 2013.

Boris Naujoks, Nicola Beume, and Michael Emmerich. Multi-objective optimisation using s-metric selection: Application to three-dimensional solution spaces. In *IEEE Congress on Evolutionary Computation (CEC)*, volume 2, pages 1282–1289. IEEE, 2005.

Radford M. Neal. *Bayesian Learning for Neural Networks*. PhD thesis, Department of Computer Science, University of Toronto, 1995.

Duy Nguyen-Tuong and Jan Peters. Model learning for robot control: a survey. *Cognitive Processing*, 12(4):319–340, 2011.

Duy Nguyen-Tuong, Jan Peters, and Matthias Seeger. Local Gaussian process regression for real time online model learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1193–1200, 2008.

Duy Nguyen-Tuong, Matthias Seeger, and Jan Peters. Model learning with local gaussian process regression. *Advanced Robotics*, 23(15):2015–2034, 2009.

Cord Niehaus, Thomas Röfer, and Tim Laue. Gait optimization on a humanoid robot using particle swarm optimization. In *Proceedings of the Second Workshop on Humanoid Soccer Robots in conjunction with the*, 2007.

José Nogueira, Ruben Martinez-Cantin, Alexandre Bernardino, and Lorenzo Jamone. Unscented Bayesian optimization for safe robot grasping. *arXiv preprint arXiv:1603.02038*, 2016.

Francesco Nori, Silvio Traversaro, Jorhabib Eljaik, Francesco Romano, Andrea Del Prete, and Daniele Pucci. iCub whole-body control through force regulation on rigid noncoplanar contacts. *Frontiers in Robotics and AI*, 2(6), 2015. doi: 10.3389/frobt.2015.00006.

Y. Ogawa, G. Venture, and C. Ott. Dynamic parameters identification of a humanoid robot using joint torque sensors and/or contact forces. In *Proceedings of the International Conference on Humanoid Robots (HUMANOIDS)*, 2014.

Anthony O'Hagan, Kennedy. M. C., and J. E. Oakley. Uncertainty analysis and other inference tools for complex computer codes. *Bayesian Statistics*, 6:503–524, 1999.

T. Okabe, Y. Jin, and B. Sendhoff. A critical survey of performance indices for multi-objective optimisation. In *Congress on Evolutionary Computation (CEC)*, volume 2, pages 878–885, Dec 2003. doi:10.1109/CEC.2003.1299759.

Miguel Oliveira, Vitor Matos, Cristina P Santos, and Lino Costa. Multi-objective parameter CPG optimization for gait generation of a biped robot. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3130–3135, 2013.

Michael A. Osborne, Roman Garnett, and Stephen J. Roberts. Gaussian processes for global optimization. In *International Conference on Learning and Intelligent OptimizatioN (LION3)*, pages 1–15, 2009.

Vilfredo Pareto. *Manuale di Economia Politica*, volume 13. Societa Editrice, 1906.

Simone Parisi, Matteo Pirotta, N. Smacchia, L. Bascetta, and Marcello Restelli. Policy gradient approaches for multi-objective sequential decision making: A comparison. In *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 1–8, Dec 2014. doi:10.1109/ADPRL.2014.7010618.

Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

N. Pedrocchi, E. Villagrossi, F. Vicentini, and L.M. Tosatti. Robot-dynamic calibration improvement by local identification. In *International Conference on Robotics and Automation (ICRA)*, 2014.

Wolfgang Ponweiser, Tobias Wagner, Dirk Biermann, and Markus Vincze. Multiobjective optimization on a limited budget of evaluations using model-assisted $\mathcal{S}$-metric selection. In *Parallel Problem Solving from Nature (PPSN) X*, pages 784–794. Springer, 2008.

Ganna Pugach, Alexandre Pitti, and Philippe Gaussier. Neural learning of the topographic tactile sensory information of an artificial skin through a self-organizing map. *Advanced Robotics*, 29(21):1393–1409, 2015. doi:10.1080/01691864.2015.1092395. URL `http://dx.doi.org/10.1080/01691864.2015.1092395`.

Katayon Radkhah, Roberto Calandra, and Marc P. Deisenroth. Learning musculoskeletal dynamics with non-parametric models. ICRA2013 Workshop on Novel Methods for Learning and Optimization of Control Policies and Trajectories for Robotics, 2013. URL `http://www.ias.tu-darmstadt.de/uploads/Research/ICRA2013/Radkah.pdf`.

Carl E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

Daniel Renjewski. *An engineering contribution to human gait biomechanics*. PhD thesis, TU Ilmenau, 2012.

Daniel Renjewski and André Seyfarth. Robots in human biomechanics - a study on ankle push-off in walking. *Bioinspiration & Biomimetics*, 7(3):036005, 2012. URL `http://stacks.iop.org/1748-3190/7/i=3/a=036005`.

Martin Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In *IEEE International Conference on Neural Networks*, pages 586 –591 vol.1, 1993. doi:10.1109/ICNN.1993.298623.

Diederik M. Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. A survey of multi-objective sequential decision-making. *J. Artif. Int. Res.*, 48(1):67–113, October 2013. ISSN 1076-9757. URL `http://dl.acm.org/citation.cfm?id=2591248.2591251`.

A. Roncone, M. Hoffmann, U. Pattacini, and G. Metta. Automatic kinematic chain calibration using artificial skin: Self-touch in the icub humanoid robot. In *International Conference on Robotics and Automation (ICRA)*, pages 2305–2312, 2014.

Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290 (5500):2323–2326, 2000.

Elmar Rueckert, R. Lioutikov, Roberto Calandra, M. Schmidt, P. Beckerle, and Jan Peters. Low-cost sensor glove with force feedback for learning from demonstrations using probabilistic trajectory representations. ICRA2015 Workshop on Tactile & force sensing for autonomous, compliant, intelligent robots, 2015.

Ruslan Salakhutdinov and Geoffrey E. Hinton. Using deep belief nets to learn covariance kernels for gaussian processes. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

Shankar Sastry. *Nonlinear systems: analysis, stability, and control*, volume 10. Springer Science & Business Media, 2013.

S. Schaal, C. G. Atkeson, and S. Vijayakumar. Scalable techniques from nonparameteric statistics for real-time robot learning. In *Applied Intelligence*, number 1, pages 49–60, 2002.

Alexandra M Schmidt and Anthony O'Hagan. Bayesian inference for non-stationary spatial covariance structure via spatial deformations. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(3):743–758, 2003.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics: Modelling, Planning and Control*. Springer, 2009.

Olivier Sigaud, Camille Salaün, and Vincent Padois. On-line regression algorithms for learning mechanical models of robots: a survey. *Robotics and Autonomous Systems*, 59(12):1115–1129, 2011.

Abhishek Singh, Barbara Minsker, and David E. Goldberg. Combining reliability and pareto optimality-an approach using stochastic multi-objective genetic algorithms. In *World Water and Environmental Resources Congress*, 2003.

Edward Snelson and Zoubin Ghahramani. Variable noise and dimensionality reduction for sparse gaussian processes. In *Uncertainty in Artifical Intelligence 22 (UAI)*, 2006.

Edward Snelson, Carl Edward Rasmussen, and Zoubin Ghahramani. Warped Gaussian processes. In *Advances in Neural Information Processing Systems (NIPS)*, volume 16, pages 337–344, 2004.

Jasper Snoek, Ryan Prescott Adams, and Hugo Larochelle. Nonparametric guidance of autoencoder representations using label information. *Journal of Machine Learning Research (JMLR)*, 13:2567–2588, 2012a.

Jasper Snoek, Hugo Larochelle, and Ryan Prescott Adams. Practical Bayesian optimization of machine learning algorithms. *arXiv preprint arXiv:1206.2944*, 2012b.

Jasper Snoek, Kevin Swersky, Richard S. Zemel, and Ryan Prescott Adams. Input warping for bayesian optimization of non-stationary functions. In *International Conference on Machine Learning (ICML)*, 2014.

Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Md Patwary, Mostofa Ali, Ryan P Adams, et al. Scalable Bayesian optimization using deep neural networks. *arXiv preprint arXiv:1502.05700*, 2015.

Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proceedings of International Conference on Machine Learning (ICML)*, pages 1015–1022, Haifa, Israel, June 2010. Omnipress. URL `http://www.icml2010.org/papers/422.pdf`.

Yanan Sui, Alkis Gotovos, Joel Burdick, and Andreas Krause. Safe exploration for optimization with gaussian processes. In *International Conference on Machine Learning (ICML)*, pages 997–1005, 2015.

Joshua Svenson and Thomas Santner. Multiobjective optimization of expensive-to-evaluate deterministic computer simulator models. *Computational Statistics & Data Analysis*, 94:250 – 264, 2016. ISSN 0167-9473. doi:http://dx.doi.org/10.1016/j.csda.2015.08.011. URL `http://www.sciencedirect.com/science/article/pii/S0167947315001991`.

Aviv Tamar, Dotan Di Castro, and Shie Mannor. Policy gradients with variance related risk criteria. *International Conference on Machine Learning (ICML)*, pages 387–396, 2012.

Zhe Tang, Changjiu Zhou, and Zengqi Sun. Humanoid walking gait optimization using ga-based neural network. In Lipo Wang, Ke Chen, and YewSoon Ong, editors, *Advances in Natural Computation*, volume 3611 of *Lecture Notes in Computer Science*, pages 252–261. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-28325-6. doi:10.1007/11539117_37. URL `http://dx.doi.org/10.1007/11539117_37`.

Ma Guadalupe Castillo Tapia and Carlos A Coello Coello. Applications of multi-objective evolutionary algorithms in economics and finance: A survey. In *IEEE congress on evolutionary computation*, volume 7, pages 532–539, 2007.

Russ Tedrake, T.W. Zhang, and H.S. Seung. Stochastic policy gradient reinforcement learning on a simple 3D biped. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 2849–2854, 2004.

Jürgen Teich. *Pareto-Front Exploration with Uncertain Objectives*, pages 314–328. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. ISBN 978-3-540-44719-1. doi:10.1007/3-540-44719-9_22. URL `http://dx.doi.org/10.1007/3-540-44719-9_22`.

Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

Matthew Tesch, Jeff Schneider, and Howie Choset. Using response surfaces and expected improvement to optimize snake robot gait parameters. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 1069–1074. IEEE, 2011.

Matthew Tesch, Jeff Schneider, and Howie Choset. Expensive multiobjective optimization for robotics. In *International Conference on Robotics and Automation (ICRA)*, pages 973 – 980, 2013.

Chris Thornton, Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 847–855. ACM, 2013.

Jo-Anne Ting, Michael Mistry, Jan Peters, Stefan Schaal, and Jun Nakanishi. A bayesian approach to nonlinear parameter identification for rigid body dynamics. In *Robotics: Science and Systems (RSS)*, 2006.

Michalis K. Titsias and Neil D. Lawrence. Bayesian Gaussian process latent variable model. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 844–851, 2010. URL `http://www.cs.man.ac.uk/~{}mtitsias/papers/vargplvmAISTATS10.pdf`.

E. Todorov. Analytically-invertible dynamics with contacts and constraints: theory and implementation in MuJoCo. In *International Conference on Robotics and Automation (ICRA)*, 2014.

Ville Tolvanen, Pasi Jylanki, and Aki Vehtari. Expectation propagation for nonstationary heteroscedastic Gaussian process regression. In *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, 2014.

Heike Trautmann, Jö Mehnen, and Boris Naujoks. Pareto-dominance in noisy environments. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 3119–3126. IEEE, 2009.

Silvio Traversaro, Andrea Del Prete, Riccardo Muradore, Lorenzo Natale, and Francesco Nori. Inertial parameter identification including friction and motor dynamics. In *International Conference on Humanoid Robots (HUMANOIDS)*, 2013.

Silvio Traversaro, Andrea Del Prete, Serena Ivaldi, and Francesco Nori. Inertial parameters identification and joint torques estimation with proximal force/torque sensing. In *International Conference on Robotics and Automation (ICRA)*, 2015.

Ioannis Tsoukalas and Christos Makropoulos. Multiobjective optimisation on a budget: Exploring surrogate modelling for robust multi-reservoir rules generation under hydrological uncertainty. *Environmental Modelling & Software*, 69: 396–413, 2015. ISSN 1364-8152. doi:10.1016/j.envsoft.2014.09.023.

David A. Van Veldhuizen and Gary B. Lamont. Multiobjective evolutionary algorithm test suites. In *ACM symposium on Applied computing*, pages 351–357. ACM, 1999.

Sethu Vijayakumar and Stefan Schaal. Locally weighted projection regression: Incremental real time learning in high dimensional space. In *International Conference on Machine Learning (ICML)*, pages 1079–1086, San Francisco, CA, USA, 2000. ISBN 1-55860-707-2. URL `http://dl.acm.org/citation.cfm?id=645529.657811`.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and Composing Robust Features with Denoising Autoencoders. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1096–1103. ACM, 2008.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research (JMLR)*, 11:3371–3408, December 2010.

Tobias Wagner, Michael Emmerich, André Deutz, and Wolfgang Ponweiser. On expected-improvement criteria for model-based multi-objective optimization. In *Parallel Problem Solving from Nature (PPSN) XI*, pages 718–727. Springer, 2010.

Niklas Wahlström, Thomas B. Schön, and Marc P. Deisenroth. Learning Deep Dynamical Models From Image Pixels. In *SYSID*, 2015.

Chunyi Wang and Radford M Neal. Gaussian process regression with heteroscedastic or non-Gaussian residuals. *arXiv preprint arXiv:1212.6246*, 2012.

Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian Process Dynamical Models for Human Motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):283–298, 2008. doi:10.1109/TPAMI.2007.1167. URL `http://dx.doi.org/10.1109/TPAMI.2007.1167`.

Zi Wang, Bolei Zhou, and Stefanie Jegelka. Optimization as estimation with gaussian processes in bandit settings. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2016.

Ziyu Wang, Masrour Zoghi, Frank Hutter, David Matheson, and Nando de Freitas. Bayesian optimization in high dimensions via random embeddings. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2013. URL `http://www.cs.ubc.ca/~hutter/papers/13-IJCAI-BO-highdim.pdf`.

P. Weber, E. Rueckert, Roberto Calandra, Jan Peters, and P. Beckerle. A low-cost sensor glove with vibrotactile feedback and multiple finger joint and hand motion sensing for human-robot interaction. In *IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2016.

L. While, L. Bradstreet, and L. Barone. A fast way of calculating exact hypervolumes. *IEEE Transactions on Evolutionary Computation*, 16(1):86–95, Feb 2012. ISSN 1089-778X. doi:10.1109/TEVC.2010.2077298.

Wayne A. Wickelgren. Speed-accuracy tradeoff and information processing dynamics. *Acta psychologica*, 41(1):67–85, 1977.

Andrew Gordon Wilson and Ryan Prescott Adams. Gaussian process covariance kernels for pattern discovery and extrapolation. *arXiv preprint arXiv:1302.4245*, 2013.

Benjamin Wilson, David Cappelleri, Timothy W Simpson, and Mary Frecker. Efficient Pareto frontier exploration using surrogate approximations. *Optimization and Engineering*, 2(1):31–50, 2001.

Katrin Witting, Sina Ober-Blöbaum, and Michael Dellnitz. A variational approach to define robustness for parametric multiobjective optimization problems. *Journal of Global Optimization*, 57(2):331–345, 2013. ISSN 0925-5001. doi:10.1007/s10898-012-9972-6. URL `http://dx.doi.org/10.1007/s10898-012-9972-6`.

K. Yamane. Practical kinematic and dynamic calibration methods for force-controlled humanoid robots. In *International Conference on Humanoid Robots (HUMANOIDS)*, 2011.

Katsu Yamane. Geometry and biomechanics for locomotion synthesis and control. In Katja Mombaur and Karsten Berns, editors, *Modeling, Simulation and Optimization of Bipedal Walking*, volume 18 of *Cognitive Systems Monographs*, pages 273–287. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-36367-2. doi:10.1007/978-3-642-36368-9_22. URL `http://dx.doi.org/10.1007/978-3-642-36368-9_22`.

Zhengkun Yi, Roberto Calandra, Filipe Fernandes Veiga, Herke van Hoof, Tucker Hermans, Yilei Zhang, and Jan Peters. Active tactile object exploration with gaussian processes. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

N. Yogeswaran, W. Dang, W.T. Navaraj, D. Shakthivel, S. Khan, E.O. Polat, S. Gupta, H. Heidari, M. Kaboli, L. Lorenzelli, G. Cheng, and R. Dahiya. New materials and advances in making electronic skin for interactive robots. *Advanced Robotics*, 29(21):1359–1373, 2015. doi:10.1080/01691864.2015.1095653. URL `http://dx.doi.org/10.1080/01691864.2015.1095653`.

Lotfi A. Zadeh. Optimality and non-scalar-valued performance criteria. *IEEE Transactions on Automatic Control*, 8(1): 59–60, Jan 1963. ISSN 0018-9286. doi:10.1109/TAC.1963.1105511.

E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, Nov 1999. ISSN 1089-778X. doi:10.1109/4235.797969.

Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8(2):173–195, 2000.

Eckart Zitzler, Marco Laumanns, and Lothar Thiele. SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In K. C. Giannakoglou, D. T. Tsahalis, J. Périaux, K. D. Papailiou, and T. Fogarty, editors, *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, pages 95–100, Athens, Greece, 2001. International Center for Numerical Methods in Engineering.

Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M Fonseca, and Viviane Grunert Da Fonseca. Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2): 117–132, 2003.

Eckart Zitzler, Joshua D. Knowles, and Lothar Thiele. Quality assessment of Pareto set approximations. In *Multiobjective Optimization*, pages 373–404, 2008.

Marcela Zuluaga, Guillaume Sergent, Andreas Krause, and Markus Püschel. Active learning for multi-objective optimization. In *International Conference on Machine Learning (ICML)*, pages 462–470, 2013.

A Benchmark Test Functions

MOP2 Function

The MOP2 function [Van Veldhuizen and Lamont, 1999] is a widely used MOO benchmark function defined as

$$f_1(\boldsymbol{\theta}) = 1 - \exp\left(-\sum_{i=1}^{d}\left(\theta_i - \frac{1}{\sqrt{d}}\right)^2\right), \tag{A.1}$$

$$f_2(\boldsymbol{\theta}) = 1 - \exp\left(-\sum_{i=1}^{d}\left(\theta_i + \frac{1}{\sqrt{d}}\right)^2\right), \tag{A.2}$$

where $-2 < \theta_i < 2$, as shown in Figure A.1. In our experiments we set the number of parameters $d = 2$, such that both the parameters and objective functions space can be easily visualized. The Pareto front of the MOP2 function can be analytically computed and is the curve defined as

$$\mathscr{P} = \left\{\boldsymbol{\theta} : \quad \theta_1 = \theta_2, \quad -\frac{1}{\sqrt{2}} < \theta_1 < \frac{1}{\sqrt{2}}\right\}. \tag{A.3}$$

For computing the hypervolume of the MOP2 function, we use the standard reference point $\mathscr{R} = [1,1]$ and as a results the optimal PF has an hypervolume of 0.3421.

ZDT3 Function

Introduced by Zitzler et al. [2000], the ZDT3 test function is a MOO test function defined as

$$f_1(\boldsymbol{\theta}) = \theta_1, \tag{A.4}$$

$$f_2(\boldsymbol{\theta}) = g(\boldsymbol{\theta})[1 - \sqrt{\theta_1/g(\boldsymbol{\theta})} - \theta_1/g(\boldsymbol{\theta}) sin(10\pi\theta_1)], \tag{A.5}$$

$$g(\boldsymbol{\theta}) = 1 + \frac{9}{d-1}\sum_{i=2}^{d}\theta_i, \tag{A.6}$$

where $0 < \theta_i < 1$. One of the main features of this test function is the presence of a disjointed PF. For our experiments, we set the number of parameters $d = 4$. For computing the hypervolume, we use the reference point $\mathscr{R} = [1,10]$.

RMTP3 function

Introduced by Deb and Gupta [2006], the RMTP3 test function is a MOO test function specifically designed for robust optimization. The RMTP3 is defined as

$$f_1(\boldsymbol{\theta}) = \theta_1, \tag{A.7}$$

$$f_2(\boldsymbol{\theta}) = h(\boldsymbol{\theta})(g(\boldsymbol{\theta}) + S(\boldsymbol{\theta})), \tag{A.8}$$

$$g(\boldsymbol{\theta}) = \sum_{i=3}^{d} 50\,\theta_i^2, \tag{A.9}$$

$$S(\boldsymbol{\theta}) = 1 - \sqrt{\theta_1}, \tag{A.10}$$

$$h(\boldsymbol{\theta}) = 2 - 0.8\exp\left(-\left(\frac{\theta_2 - 0.35}{0.25}\right)^2\right) - \exp\left(-\left(\frac{\theta_2 - 0.85}{0.03}\right)^2\right). \tag{A.11}$$

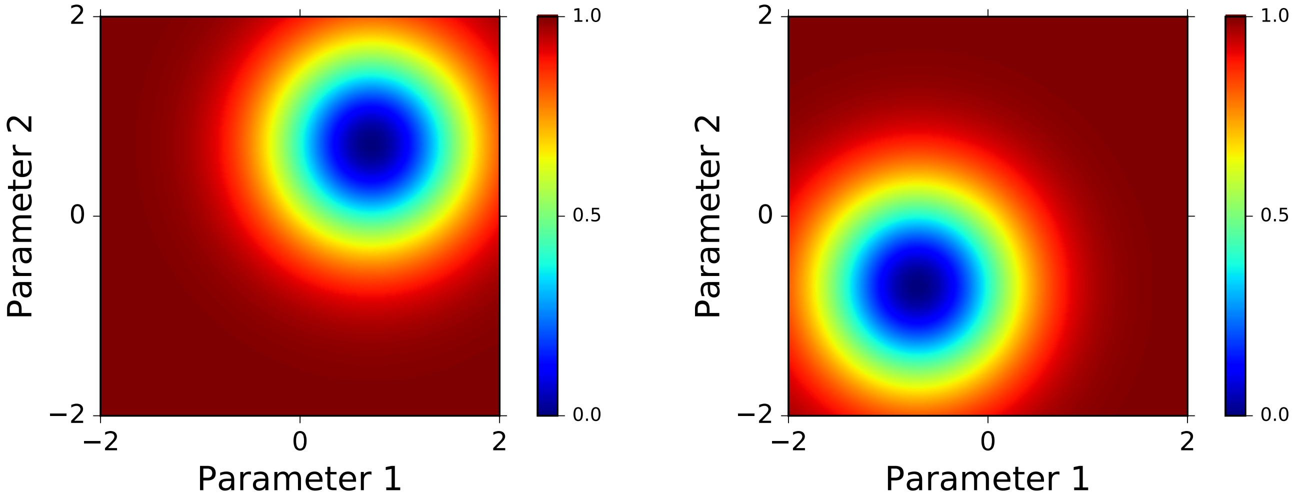
This function possess one global minima in $\boldsymbol{\theta} = [0, 0.85]$ and a local minima in $\boldsymbol{\theta} = [0, 0.35]$. However, the parameters region around $\theta_2 \approx 0.85$ is sensitive to perturbations of the parameter θ_2. On the other hand, the region around $\theta_2 \approx 0.35$ is fairly robust to perturbations, and is therefore preferable, despite being suboptimal (i.e., this set of parameters is not on the PF). In our experiments, we set the number of parameters $d = 2$.

Branin function

The “Branin rcos 1” function is a widely used SOO benchmark function defined as

$$f_1(\boldsymbol{\theta}) = 1 - \exp\left(-\sum_{i=1}^{d}\left(\theta_i - \frac{1}{\sqrt{d}}\right)^2\right), \tag{A.12}$$

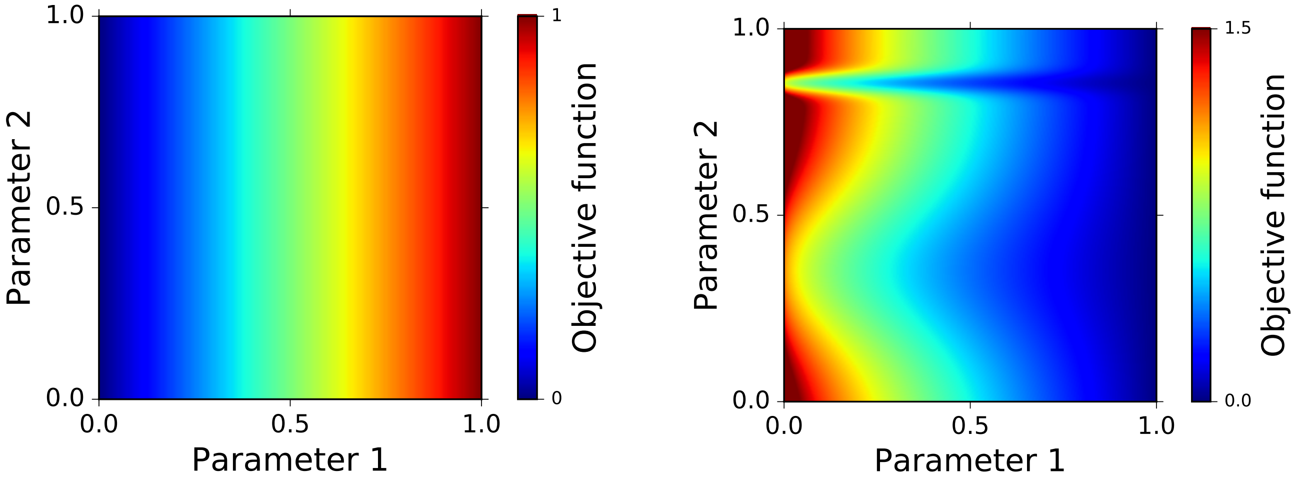
where $-5 < \theta_1 < 10$ and $0 < \theta_2 < 15$.



(a) Objective 1

(b) Objective 2

Figure A.1.: MOP2 function.



(a) Objective 1

(b) Objective 2

Figure A.2.: RMTP3 function.

Branin Function with Heteroscedastic Noise

We modify the standard Branin function (see A) by adding heteroscedastic noise as

$$f_1(\boldsymbol{\theta}) = 1 - \exp\left(-\sum_{i=1}^{d}\left(\theta_i - \frac{1}{\sqrt{d}}\right)^2\right) + \mathcal{N}\left(0, \frac{\pi}{10}|\theta_1|\right), \tag{A.13}$$

$$f_2(\boldsymbol{\theta}) = 1 - \exp\left(-\sum_{i=1}^{d}\left(\theta_i + \frac{1}{\sqrt{d}}\right)^2\right) + \mathcal{N}\left(0, \frac{\pi}{10}|\theta_1|\right), \tag{A.14}$$

where $-5 < \theta_1 < 10$ and $0 < \theta_2 < 15$.

Fox dataset

The Fox dataset has been collected from the bipedal walker “Fox”, shown in Figure A.6. The dataset consists of 1909 evaluations having 4 input parameters and 2 objectives functions: walking speed and energy consumption. For each evaluation, the robot was let walking for 12 seconds. The first objective function (the average walking speed) is computed using the total walked distance, as measured by the boon of the robot. The second objective function (the energy consumption) is instead estimated using the Ohm’s law equation $P = IV$ as the sum of the power $\sum P$ over all the

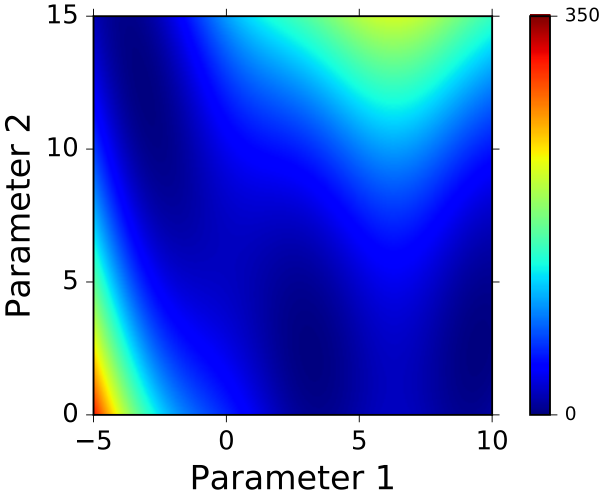


Figure A.3.: Branin rcos 1 function.

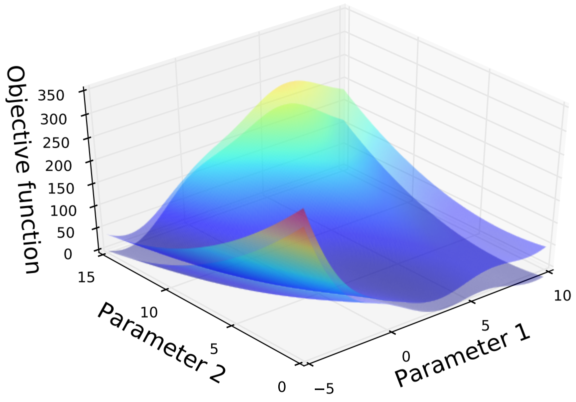


Figure A.4.: The Branin function with heteroscedastic noise defined in Equation (A.14). The two surfaces visualize the mean ± std.

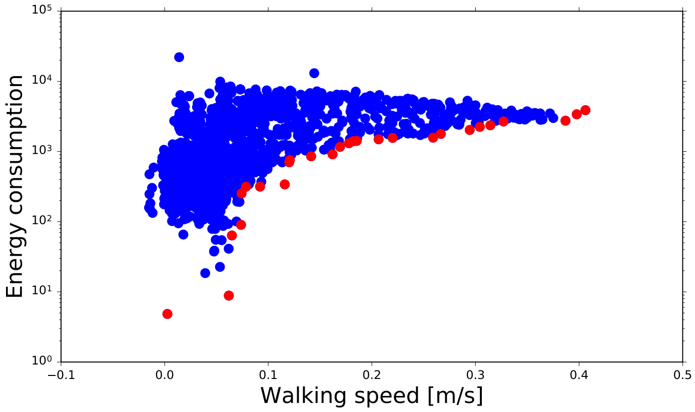


Figure A.5.: Trade-off curve for the Fox dataset from the 1909 evaluations performed on the real bipedal walker. The red datapoints belongs to the PF computed from the evaluations.

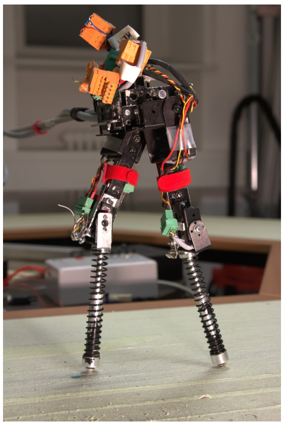


Figure A.6.: The bipedal walker "Fox" used to collect the dataset.

actuated degrees of freedom (i.e., the knees and the hips). For further information about the Fox robot and its controller you are referred to Calandra et al. [2015d].

B Curriculum Vitae

Roberto Calandra

Education

2012 - 2016 **Ph.D. Student in Computer Science, Technische Universität Darmstadt, Germany**
Thesis topic: Bayesian Modeling for Optimization and Control in Robotics
Supervisor: Jan Peters (TU Darmstadt)
Instructor: Marc P. Deisenroth (Imperial College London)
Website: `http://www.ias.tu-darmstadt.de/Team/RobertoCalandra`
Collaborating in the European Project CoDyCo

2009 - 2011 **M.Sc. in Machine Learning and Data Mining, Aalto University, Finland**
Thesis topic: An Exploration of Deep Belief Networks toward Adaptive Learning
Supervisor: Olli Simula (Aalto University)
Instructors: Federico Montesino Pouzols (University of Helsinki), Tapani Raiko (Aalto University)

2004 - 2009 **B.Sc. in Computer Science Engineering, Università degli Studi di Palermo, Italy**
Thesis topic: Design and Building of a Robotics mobile platform
Supervisor: Haris Dindo (Università degli Studi di Palermo)

Additional Education

15 - 24 July 2013 Attendance of the **iCub Summer School 2013**
Sestri Levante, Italy
Topic: Software engineering for humanoid robots

9 - 27 July 2012 Attendance of the **Graduate Summer School 2012**
Institute for Pure and Applied Mathematics (IPAM), UCLA, US
Topic: Deep Learning, Feature Learning

11 - 20 April 2012 Attendance of the **Machine Learning Summer School 2012**
La Palma, Spain
Topic: Machine Learning

18 - 22 September 2006 Attendance of the **6th International UJI Robotic School**
Robotic Intelligence Lab of Jaume-I University, Spain
Topic: Research and development on Humanoid Robots

Work Experience

July - October 2015 **Internship**, Microsoft Research, Cambridge, UK
Worked in the Machine Learning and Perception (MLP) group on the Malmo project.
[BBC coverage of the project]
Advisors: Andrew Blake, Katja Hofmann

May - December 2010 **Research Assistant**, BECS Department, Aalto University, Finland
Worked as data analyst using Gaussian Processes. Developed novel variational inference methods based on EP for LOO.
Contributed to the development of the Gaussian Processes toolbox GPStuff.
Advisor: Aki Vehtari

March - May 2008 **Internship**, BELTEC s.r.l., Italy
Designed software and hardware components for industrial automation on two different projects for *SELEX Galileo*, and on a project for the *INAF - Istituto Nazionale di Astrofisica* (Italian national institute for astrophysics).

Teaching / Supervision

Teaching:

- Machine Learning I Lectures, Spring 2015, Teaching assistant, TU Darmstadt
- Robot Learning Lectures, Fall 2014, Teaching assistant, TU Darmstadt
- Robot Learning Lectures, Fall 2013, Teaching assistant, TU Darmstadt

Supervision:

- Supervisor ADP Project: Leser G. and Hatzenbühler, J. and Schwaab, J. and Eschner, N. - Implementation and Improvement of the bipedal walking robot *Fox*2 (jointly with Philipp Beckerle and Alexandra Voloshina)
- Supervisor Robot Learning Project Spring 2015: Fritsche L. and Unverzagt F. - Human-friendly Telecontrol of the *iCub*
- Supervisor Bachelor Thesis: Unverzagt F. - Modeling Robustness for Multi-Objective Optimization
- Supervisor Fritsche L. - Learning to walk on rough terrain
- Co-supervisor Bachelor Thesis: Andreas Schaefer - Prediction of Finger Flexion from ECoG Data with Deep Neural Networks
- Co-supervisor Bachelor Thesis: Dominik Pfau - Multi-Objective Optimization and Analysis of a Musculoskeletal Robot for Bipedal Locomotion (jointly with Katayon Radkhah)
- Supervisor Bachelor Thesis: Melvin Laux - Online Feature Learning for Reinforcement Learning
- Supervisor Bachelor Thesis: Aaron Hochlaender - Reinforcement Learning of PACMAN (co-supervised by Gerhard Neumann)
- Supervisor HiWi: Stefan Luthardt - Deep Learning for Artificial Skin
- Supervisor Robot Learning Project Fall 2014: Geukes, J. and Nakatenus, M. - Towards Balancing with the *iCub*
- Supervisor Robot Learning Project Fall 2013: Prediger, M. and Schnell, F. and Negoescu V. - Advanced Bayesian optimization models
- Supervisor Robot Learning Project 2012/2013: Wolter, E. and Baark, T. - Learning to bounce a ball with a robotic arm

Honors and Awards

- NIPS Travel Award 2015
- Robotics and Automation Society ICRA Student Travel Grant 2015
- DAAD Travel Grant to participate to the LION8 conference
- Travel Grant for the Graduate Summer School 2012
- Travel Grant for the Machine Learning Summer School 2012

Invited presentations

- 13 May 2015, CoDyCo third year review, TU Darmstadt, Darmstadt, Germany
- 02 May 2016, Universität Stuttgart, Stuttgart, Germany, **host:** Marc Toussaint, Machine Learning & Robotics Lab
- 16 Oct 2015, University College London, London, UK, **host:** Guy Lever
- 14 Oct 2015, University of Oxford, Oxford, UK, **host:** Michael Osborne, Machine Learning Research Group
- 13 Oct 2015, Imperial College London, London, UK, **host:** Stefan Leutenegger, Dyson Robotics Lab
- 03 Jun 2015, University of British Columbia, Vancouver, Canada, **host:** Mark Schmidt

- 02 Jun 2015, University of Washington, Seattle, US, **host:** Dieter Fox, Robotics and State Estimation Lab
- 15 May 2015, CoDyCo second year review, UPMC, Paris, France
- 01 Apr 2015, TU Freiburg, Freiburg, Germany, **host:** Frank Hutter
- 31 Mar 2015, TU Freiburg, Freiburg, Germany, **host:** Wolfram Burgard, Autonome Intelligente Systeme
- 22 Dec 2014, Università degli Studi di Palermo, Palermo, Italy
- 21 Nov 2013, CoDyCo meeting, Ljubliana
- 24 Apr 2014, Bosch Research, Stuttgart, Germany
- 15 May 2014, CoDyCo first year review, IIT, Genoa, Italy
- 21 Nov 2013, CoDyCo meeting, UPMC, Paris, France
- 13 Nov 2013, Imperial College London, London, UK
- 20 Jul 2013, iCub Summer School 2013

Professional Service

Organizer NIPS Workshop on Bayesian optimization (BayesOpt) 2015 with N. de Freitas (University of Oxford, Deepmind), R. P. Adams (Harvard University), B. Shahriari (University of British Columbia) and A. Shah (University of Cambridge)

Reviewer for Journals

- International Journal of Robotics Research (IJRR)
- Robotics and Autonomous Systems
- IEEE Transactions on Robotics (TRO)
- Neurocomputing
- IEEE Transactions on Cybernetics
- IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics
- Autonomous Robots (AuRo): Special Issue on Whole-body control for Humanoid Robots
- Autonomous Robots (AuRo): Special Issue on Assistive and Rehabilitation Robotics

Reviewer for Conferences & Workshops

- IEEE/ASME Internationl Conference on Advanced Intelligent Mechatronics (AIM): 2014
- International Conference on Artificial Intelligence and Statistics (AISTATS): 2014–2016
- International Conference on Humanoid Robots (HUMANOIDS): 2014–2015
- International Conference on Artificial Neural Networks (ICANN): 2014
- International Conference on Robotics and Automation (ICRA): 2015
- International Joint Conference on Artificial Intelligence (IJCAI): 2013, 2016
- International Conference on Intelligent Robots and Systems (IROS): 2013–2016
- Neural Information Processing Systems (NIPS): 2014–2015
- Robotics: Science and Systems (RSS): 2014-2016

- NIPS 2014 workshop on Bayesian optimization (BayesOpt)
- NIPS 2014 workshop on Autonomously Learning Robots

Reviewer for Books

- Wiley Book in Robotics
- CRC Press Book in Computer Science