Nonlinear Model Predictive Control with Probabilistic Models

Nonlinear Model Predictive Control bei Probabilistischen Modellen

Master-Thesis von Felix Schmitt aus Darmstadt

September 2013

TECHNISCHE UNIVERSITÄT DARMSTADT

Nonlinear Model Predictive Control with Probabilistic Models
Nonlinear Model Predictive Control bei Probabilistischen Modellen

Vorgelegte Master-Thesis von Felix Schmitt aus Darmstadt

1. Gutachten: Prof. Dr. Jan Peters
2. Gutachten: Prof. Dr. Stefan Ulbrich
3. Gutachten: Dr.-Ing. Marc Peter Deisenroth

Tag der Einreichung:

Erklärung zur Master-Thesis

Hiermit versichere ich, die vorliegende Master-Thesis ohne Hilfe Dritter 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.

Darmstadt, den 20. September 2013

(F. Schmitt)

Acknowledgements

I would like to thank Dr.-Ing. Marc Deisenroth, not only for the suggestion of the topic, but also for his long-standing, generous and patient support during my work on the thesis.

I am grateful to Prof. Dr. Jan Peters and the Intelligent Autonomous Systems Group for accepting this external master thesis and also for the possibility to work in the group and to use their facilities – especially the coffee-machine

Prof. Dr. Stefan Ulbrich I wish to thank that he agreed on providing the necessary supervision form side of the mathematics department and for the possibility to work into Nonlinear Model Predictive Control in his master-seminar on nonlinear optimization.

In implementation I used external open-source software from the Optimization in Engineering Group of the KU Leuven. Thanks to Joachim Ferreau and Milan Vukov for their quick and competent technical support.

I am also thankful to my colleagues in the lab, Sanket Kamthe, Rudolf Lioutikov, Marco Ewerton and Dominik Notz, it has been a great time in a nice environment.

Finally, I sincerely thank my girlfriend for her patience when I was absent-minded during my thesis work and my parents for their support that enabled me to study.

Abstract

Nonlinear Model Predictive Control (NMPC) is a powerful control framework, which strongly relies on a good model of the system dynamics. In the case, such a model is not available a-priori, non-parametric regression using Bayesian regression or *Gaussian Processes (GPs)* have been shown promising in inferring the dynamics from collected data. An advantage of Bayesian methods and GPs over other regression methods is the availability of a predictive distribution expressing the uncertainty about the true function induced by the finite amount of observations. Although recent work indicates that propagation of this uncertainty can be used to design robust controllers, it has not been considered in NMPC yet. This thesis presents an approach to robust *Semi-Implicit* NMPC of Bayesian linear models and Gaussian Process dynamics subject to control constraints. The propagation of the uncertainty is done by means of the *Moment-Matching (MM)* technique to track the central moments of the state distribution and a recent approximation framework is used for fast online NMPC. Although the approach has several advantages from a theoretical perspective, its performance on a highly nonlinear benchmark system is worse than expected. Several possible sources for the degradation are investigated and discussed.

Zusammenfassung

Nonlinear Model Predictive Control (NMPC) ist eine mächtige Methode der Steuerung und Regelung. Diese setzt allerdings ein gutes Modell des Systemverhaltens voraus. Wenn ein solches Modell nicht verfügbar ist, können Verfahren der *nicht-parametrischen* Regressionsschätzung, besonders *Gaussian Processes (GPs)*, erfolgreich eingesetzt werden um das Systemverhalten aus Beobachtungen zu schätzen. Bayes'sche Modelle und Gaussian Processes zeichnen sich gegenüber anderen Methoden dadurch aus, dass sie die Möglichkeit bieten mittels einer Wahrscheinlichkeitsverteilung die Unsicherheit in der Vorhersage, bei einer möglicherweise geringen Anzahl von Beobachtungen, zu quantifizieren. Obwohl in aktuellen Veröffentlichungen bereits gezeigt werden konnte, dass mithilfe der Fortpflanzung dieser Verteilung über Zeit robuste Regler entworfen werden können, wurde diese Methode bisher noch nicht in NMPC verwendet. In dieser Arbeit wird ein Ansatz zur *Semi-Impliziten* NMPC von Bayes'schen lineare Modellen sowie Gaussian Processes bei zusätzlichen Steuerungsbeschränkungen vorgestellt. Die Unsicherheit wird hierbei durch die *Moment Matching (MM)* Technik fortgesetzt, während die Online-Steuerung durch ein modernes Approximationsverfahren erfolgt. Obwohl der vorgestellte Algorithmus von theoretischer Seite vielversprechend ist, zeigte sich die tatsächliche Regelungsqualität auf einem nichtlinearen Testsystem als schlechter als erwartet. Die möglichen Ursachen für das schlechte Abschneiden werden deshalb dargestellt und diskutiert.

Acronyms

MPC Model Predictive Control

NMPC Nonlinear Model Predictive Control

GP Gaussian Process

QP Quadratic Program

NLP Nonlinear Programming

SQP Sequential Quadratic Programming

DP Dynamic Programming

LQR Linear Quadratic Regulator

KKT Karush-Kuhn-Tucker (Conditions)

LICQ Linear Independence Constraint Qualification

SEARD Squared Exponential (Kernel) with Automatic Relevance Detection

MM Moment-Matching

GP-NMPC (Full) Gaussian Process Nonlinear Model Predictive Control

D-GP-NMPC Deterministic Gaussian Process Nonlinear Model Predictive Control

AUG-GP-NMPC Augmented Gaussian Process Nonlinear Model Predictive Control

GP-MPC Gaussian Process (Linear) Model Predictive Control

Notation

To prevent misunderstandings due to the fact that this thesis consists of optimization and machine learning parts, in the following the notation is defined for controversial cases and less common operators.

General

- Bold symbols denote abstract tubles, e.g. $\boldsymbol{M}$ means the tuple (M_1, M_2, M_3). If the objects are vectors, involved into calculations, it means the concatenation into a single column, e.g.

$$M\boldsymbol{x} = M \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}.$$

- Matrices are always capital letters.
- Standard scalar operations, like $\exp, \sqrt{}, \log, \leq, \neq$, are applied element-wise if the arguments are vectors or matrices. Similar, operations on vectors $\phi(x) = y,\ x \in \mathbb{R}^n, y \in \mathbb{R}^m$ are extended to matrices, by applying them to each column $\phi(X) = Y,\ X \in \mathbb{R}^{n\times k}, Y \in \mathbb{R}^{m\times k}$.

Linear Algebra

- Symmetric matrices $\in \mathbb{R}^{n\times n}$ are denoted by $\mathbb{S}^n$.
- Positive definite matrices are denoted by $\succ 0$, whereas positive semi-definite matrices are indicated by $\succeq$.
- The Cholesky-factor L of a positive definite matrix $Q = LL^T$ is denoted by $Q^{\frac{1}{2}}$.
- The function tr is the sum of all diagonal elements.
- The operator diag extracts the diagonal of a matrix resulting in a column vector.
- The operator $\mathrm{blkdiag}$ builds the block-diagonal matrix, by diagonal concatenation of a tuple of matrices.
- The operator vec maps a matrix to a single column vector, consisting of the concatenation of the individual columns. vec^{-1} is the inverse map, which is generally not well-defined, but only used if the dimensionality of the original matrix is known.

- $\otimes$ is the *Kronecker Product* of two matrices, e.g.

$$A \in \mathbb{R}^{n \times m}, B \in \mathbb{R}^{l \times k} \Rightarrow A \otimes B = \begin{pmatrix} A_{1,1}B & \cdots & A_{n,1}B \\ \dots & \ddots & \dots \\ A_{1,m}B & \cdots & A_{n,m}B \end{pmatrix} \in \mathbb{R}^{nl \times mk}.$$

- The identity matrix is denoted by $\mathfrak{I}$, if the dimensionality is of importance, it will be denoted by a super-script. $\mathfrak{I}_i$ is the i-th unity vector.
- $\|.\|_2$ denotes the standard euclidean norm, whereas $\|.\|_Q$ for a positive definite matrix Q is defined as $\|Q^{\frac{1}{2}}.\|_2$.

Additionally, standard formulas from the Matrix Cookbook [Petersen and Pedersen, 2006] will be used in this thesis without further citation .

Probability Theory and Machine Learning

- If x_i, y_i are individual samples, then X, Y denote the matrices of a data set, were each column corresponds to a sample .
- $\mathbb{E}, \mathbb{V}$ denote the expectation and the covariance matrix of a random-variable or distribution.
- We use the common, yet sloppy, notation in machine learning of not distinguishing between distributions and their densities with respect to either the Dirac- or Lebesgue-measure. The standard notation is $p(x), q(x)$, whereas special distribution, namely GP-posteriors $\mathscr{GP}$ and Gaussian distributions $\mathcal{N}$, are in calligraphic symbols.
- $\underset{\text{i.i.d}}{\sim}$ is the abbreviation for "identical independent distributed"
- $\mathfrak{K}, \mathfrak{k}$ denote the kernel-matrix and the kernel-function of a GP.
- Integrals of several variables are denoted by $\int . \mathrm{d}(x_1, x_2)$ for sake of brevity.

Contents

1 Introduction

Controlling a complex nonlinear system in real-time is still a challenging task in process engineering. Standard *closed-loop* feedback controllers like fixed *Proportional Differential* regulators, which are fast enough for real-time application and straightforward to implement might be too crude to achieve sufficient stability. Often the reasons is that they only take into account the recent time-step. On the other hand, off-line *open-loop* optimal control can be inapplicable because the system can not be modeled sufficient well and is additionally subject to disturbances [Diehl, 2001].

Model Predictive Control (MPC) is a framework for real-time control, which tries to exploit the advantages of both closed-loop feedback and open-loop control [Diehl, 2001]. The high-level idea is to solve an optimal control problem on a finite *Prediction-Horizon* $[t_0, t_0 + T]$, measure the state x_{t_0}, apply the first control signal u_{t_0} and then move on to $[t_0 + \delta, t_0 + \delta + T]$ using the previous solution to initialize.

In the case of linear deterministic, possibly constrained systems already a variety of commercial applications (~ 4600) were reported [Qin and Badgwell, 2003] in 2003, which can be explained by the achieved maturity from both theoretical and numerical sides. However, in NMPC, commercial applications were still few in 2003 [Qin and Badgwell, 2003], due to the higher computational burden. Unfortunately, more recent statistics are not available, but the number of applications has for sure increased further.
In academics NMPC has long been studied from theoretical view, more recent algorithms for real-time application have been developed [Diehl et al., 2009] and been successfully tested on real systems. Especially the open source ACADO Toolkit[1] shows very promising performance as reported in [Houska et al., 2011]. The used method achieves high performance in terms of speed, as it is able to start numerical optimization even before the state is measured using *Parametric Optimization*. Additionally it only iterates once before moving to the next time-step, while still achieving a reasonable bound on sub-optimality [Diehl, 2001].

An key requirement with Model Predictive Control is the model of the system to control. If the physical laws the system obeys are known, appropriate equations of motion can be derived. Nevertheless, in many cases, these equations contain open parameters, which can not be defined directly, but must be estimated using sample data, e.g., by least-squares estimation. From a statistical view these point-estimates are problematic, especially when the amount of sample data is small. Hence, in statistics and Bayesian machine learning, confidence regions or posterior distributions are preferred. Additionally, in some real systems certain parameters have to be considered stochastic, e.g., normally distributed weight of the load carried by a robot in automatic luggage transport at an airport [Zantz, 2006]. Hence, for use in application, NMPC is desired to be robust against parameter uncertainty and external disturbances.

[1] `http://acadotoolkit.org/`

Recent approaches for robust control under parametric uncertainty, based on the concept of *Robust Positive Invariant Tubes*, have emerged [Houska, 2011]. A disadvantage of these techniques compared to probabilistic ones is that they do not use the entire distribution of the parameters but only a confidence region, often leading to “pessimistic descion[s]” [Marti and Stein, 2012] (section 1.2., last stanza).
Robot-trajectory planing under stochastic uncertainty has been studied by the group of K. Marti: Given a geometrical path, optimizing the joint trajectory subject to stochastic parameter uncertainty can be formulated as variational problem with uncertain parameters and solved by spline approximation and stochastic *Nonlinear Programming (NLP)* [Marti and Qu, 1998]. However, the computational costs are enormous, hence, for real-time application *Neural-Network* approximation is used [Zantz, 2006]. More recently MPC of linear affine systems, where the time-varying displacement is subject to stochastic parameter uncertainty, has been considered [Marti and Stein, 2012].

If not only few parameters, but parts of the model are unknown,e.g., because they cannot be modeled analytically, non-parametric regression on sample trajectories can be used. Here Bayesian regression methods like GPs, offer the possibility of a "distribution over candidate functions" [Rasmussen and Williams, 2006] instead of a point estimate. GPs are state-of-the-art regression techniques and have been used for a variety of tasks in machine learning, among them depth estimation from stereo images [Sinz et al., 2004], approximation of stochastic differential equations [Archambeau et al., 2007] and the automatic generation of play-lists [Platt et al., 2001].
In the last years several authors started using GPs to learn dynamical models for control: [Nguyen-Tuong et al., 2008] used an online-learning approach for *Computed Torques*, a single-step feedback controller based on a system-model, with a GP and [Kocijan et al., 2004] presents several applications of GPs in control, among them NMPC on the mean-function and control with a learned auto-regressive forward-model.
Based on the article [Quinonero-Candela et al., 2003], which gives an approximation framework for the propagation of uncertainty in probabilistic dynamic models based on certain classes of GPs, applications to model-based *Reinforcement-Learning* [Deisenroth and Rasmussen, 2011] and *Imitation Learning* [Englert et al., 2013] as instances of stochastic optimal control under model-uncertainty have been proposed.

In this thesis, a new combination of the approximation framework with Bayesian models, especially GPs, is presented. Chapter 2 gives an introduction to MPC and then focuses on the aspects of the techniques developed by Moritz Diehl [Diehl et al., 2009] and his group members, including homotopy methods for hot-starting and theoretical results on sub-optimality. Bayesian linear regression in the context of learning system linearizations and Gaussian Processes are presented in chapter 3. In chapter 4, we show how both previous methods can be combined for robust NMPC with GP models, leading to our proposed algorithm. Finally, a benchmark system is presented in chapter 5 on which the method is tested , followed by a discussion of the performance and possible reasons in chapter 6.

2 Model Predictive Control

As mentioned in the introduction, MPC is an approach to closed loop control by iteratively solving open-loop problems in real-time.

Therefore, the definition of the MPC-framework requires:

- A state variable $x_t \in \mathbb{R}^{n_x}$, which is assumed to be given by direct measurement
- A control variable $u_t \in \mathbb{R}^{n_u}$, which is applied to the real system
- A planning horizon $T \in \mathbb{N}$
- A series of planning states $s_i^t \in \mathbb{R}^{n_s}, i \in \{1, 2, \ldots, T+1\}$, which are the predicted future system behavior, starting at time t.
- A series of planning controls $q_i^t \in \mathbb{R}^{n_q}, i \in \{1, 2, \ldots, T\}$, which are *not* applied but used for the planning of the next time steps.
- A model of the system dynamics $x_{t+1} = f(x_t, u_t)$, which is assumed to be twice continuously differentiable $f \in \mathscr{C}^2(\mathbb{R}^{n_x+n_u}, \mathbb{R}^{n_x})$.
- Inequality constraints $c_t(x_t, u_t) \leq 0 \quad \forall t$
- A tracking objective, defined by the sum of the squared deviation from a given reference $x_t^\star, u_t^\star$:

$$\|x_t^\star - x_t\|_{Q_t}^2 + \|u_t^\star - u_t\|_{R_t}^2 \tag{2.1}$$

with weight matrices $Q_t \in \mathbb{S}^{n_x \times n_x}, R_t \in \mathbb{S}^{n_u \times n_u}$, $Q_t, R_t \succ 0$

At every time step t in the MPC framework, the constrained optimal control problem

$$\text{S-MPC}(x_t) = \begin{cases} \min_{s,q} & \|s_{T+1}^t - x_{T+t}^\star\|_{Q_{t+T}}^2 \\ & \quad + \sum_{i=1}^{T} \left(\|x_{i-1+t}^\star - s_i^t\|_{Q_{t-1+i}}^2 + \|u_{i-1+t}^\star - q_i^t\|_{R_{t-1+i}}^2 \right) \\ \text{s.t.} & s_1^t - x_t = 0 \\ & i \in \{1, \ldots, T\} \\ & f(s_i^t, q_i^t) - s_{i+1}^t = 0 \\ & c(s_i^t, q_i^t) \leq 0 \end{cases} \tag{2.2}$$

has to be solved. The current control $u_t \equiv q_i^t$ is then applied to the system, while the next MPC-problem is initialized with the shifted control variables

$$(s^{t+1}_{1,\dots,T}, q^{t+1}_{1,\dots,T-1}) = (s^t_{2,\dots,T+1}, q^t_{2,\dots,T}). \tag{2.3}$$

The entire procedure can be summed up in a prototypic algorithm 1.

Algorithm 1: Prototypic Model Predictive Control algorithm

Input: Model of the system dynamics $f(.)$, constraints $c_t(.)$, reference $\boldsymbol{x}^\star, \boldsymbol{u}^\star$ and weights $\boldsymbol{Q}, \boldsymbol{R}$

for $t \in \{1, \dots, T_{end}\}$ **do**

- x_t =`Measure state()`
- $T \leftarrow \min(T, T_{\text{end}} - t - 1)$
- $(\boldsymbol{s}^t, \boldsymbol{q}^t) \leftarrow$`Solve MPC-Problem(`$x_t | t, T, \boldsymbol{x}^\star, \boldsymbol{u}^\star, \boldsymbol{Q}, \boldsymbol{R}, (\boldsymbol{s}^t, \boldsymbol{q}^t)$`)`
- `Apply control to system(`q_1^t`)`
- Initalize by shift: $(s^{t+1}_{1,\dots,T}, q^{t+1}_{1,\dots,T-1}) = (s^t_{2,\dots,T+1}, q^t_{2,\dots,T})$

end

Figure 2.1 shows the typical behaviour of a MPC algorithm on a tracking task: Because of the

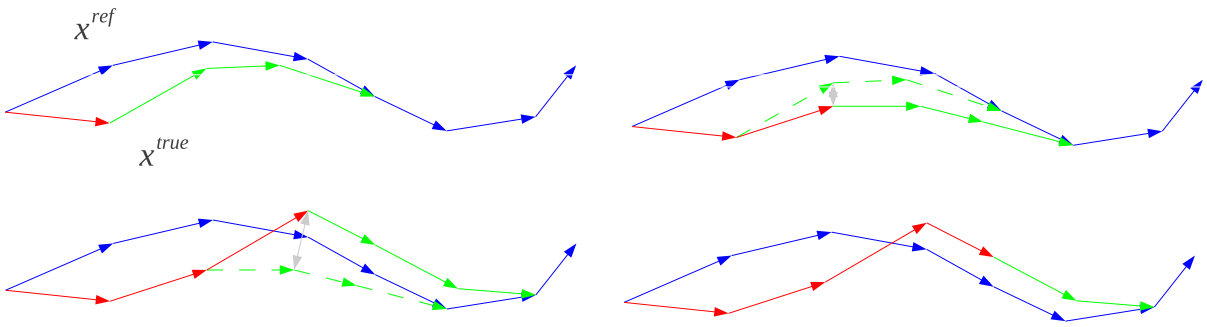

Figure 2.1: Model Predictive Control on a tracking task. To minimize the deviation from the reference [blue], a sequence of states [green] from a disturbed state [red] is planned. As additional disturbances [gray] occur during execution, replanning has to be done at every step.

planning of the future control signals, this approach often has superior performance in practice. In [Gruene and Pannek, 2011] exhaustive material on theoretical control properties can be found, including Lyapunov-stability and comparison to infinite-horizon dynamic programming.

2.1 Linear Quadratic Model Predictive Control

An important class of MPC-Problems are those with linear affine dynamics

$$x_{t+1} = f(x_t, u_t) := A_t x_t + B_t u_t + a_t,$$

$A_t \in \mathbb{R}^{n_x \times n_x}, B_t \in \mathbb{R}^{n_x \times n_u}, a_t \in \mathbb{R}^{n_x}$.

2.1.1 Linear Quadratic Regulator

With the quadratic cost (2.1) and without constraints, the corresponding optimal control problem (2.2) is defined as:

$$\begin{aligned} \min_{\boldsymbol{s},\boldsymbol{q}} \quad & ||s_{T_{\text{end}}+1} - x^\star_{T_{\text{end}}+1}||^2_{Q_{T_{\text{end}}+1}} + \sum_{t=1}^{T_{\text{end}}} \left(\|x^\star_t - s_t\|^2_{Q_t} + \|u^\star_t - q_t\|^2_{R_t} \right) \\ \text{s.t.} \quad & s_1 - x_0 = 0, \\ & A_t s_t + B_t q_t + a_t - s_{t+1} = 0, \quad t \in \{1, \dots, T_{\text{end}}\}. \end{aligned}$$

Although the optimization problem is a simple *Quadratic Program (QP)*, which could be solved by an arbitrary solver, a different approach will be presented.
The *Dynamic Programming (DP)* framework developed by R. Bellman in the 1950s [Bellman, 1954] is not only one of the first practical optimal control algorithms, but also gives, if tractable, the solution of the problem *dependent* on the initial value. Therefore we can computed all optimals controllers *offline* for the entire time-horizon T_{end}. The first application of dynamic programming in form of the Hamilton-Bellman-Jacobi equation to the *Linear Quadratic Regulator (LQR)* control problems is R.E. Kalman's paper [Kalman, 1960] "Contributions to the theory of optimal control" from 1960.

Dynamic Programming for LQR

DP can be described as a procedure, which solves the optimization problem by recursively solving parametric sub-problems.

First a *Value Function* $V_t(x_t)$ is defined as the optimal value of the shrunken optimization problem

$$\begin{aligned} \mathrm{V}_t(x_t) = \min_{\boldsymbol{s},\boldsymbol{q}} \quad & ||s_{T_{\text{end}}+1} - x^\star_{T_{\text{end}}+1}||^2_{Q_{T_{\text{end}}+1}} + \sum_{\underline{k=t}}^{T_{\text{end}}} \left(\|x^\star_k - s_k\|^2_{Q_k} + \|u^\star_k - q_k\|^2_{R_k} \right) \\ \text{s.t.} \quad & \underline{s_t - x_t} = 0, \\ & A_k s_k + B_k q_k + a_k - s_{k+1} = 0, \quad \underline{k \in \{t, \dots, T_{\text{end}}\}}. \end{aligned}$$

It is easy to verify that the *Bellman-Equation*

$$V_t(x_t) = \min_{q_t} \left[\|x_t^\star - x_t\|_{Q_t}^2 + \|u_t^\star - q_t\|_{R_t}^2 + V_{t+1}(A_t x_t + B_t q_t + a_t) \right]$$

holds for all value functions.
Using $V_{T_{\text{end}}}(x_{T_{\text{end}}}) = ||s_{T_{\text{end}}+1} - x^\star_{T_{\text{end}}+1}||^2_{Q_{T_{\text{end}}+1}}$, which can be written in the form $x^T_{T_{\text{end}}} P_{T_{\text{end}}} x_{T_{\text{end}}} + p^T_{T_{\text{end}}} x_{T_{\text{end}}} + v_{T_{\text{end}}}$, and iteratively applying the Bellman-equation, the optimal controller

$$\pi_t(x_t) = \arg\min_{q_t} \left[\|x_t^\star - x_t\|_{Q_t}^2 + \|u_t^\star - q_t\|_{R_t}^2 + V_{t+1}(\underbrace{A_t x_t + B_t q_t + a_t}_{x_{t+1}}) \,\middle|\, x_t \right]$$

can be computed:

Assuming $V_t(x_t) = x_t^T P_t x_t + p_t^T x_t + v_t$ the Bellman-Equation turns into

$$\begin{aligned} V_{t-1}(x_{t-1}) = \min_{q_t} \Big[& \|x_{t-1}^\star - x_{t-1}\|_{Q_{t-1}}^2 + \|u_t^\star - q_{t-1}\|_{R_{t-1}}^2 \\ & + (A_{t-1}x_{t-1} + B_{t-1}q_{t-1} + a_{t-1})^T P_t (A_{t-1}x_{t-1} + B_{t-1}q_{t-1} + a_{t-1}) \\ & + p_t^T (A_{t-1}x_{t-1} + B_{t-1}q_{t-1} + a_{t-1}) + v_t \Big]. \end{aligned} \tag{2.4}$$

The necessary first-order optimality condition of (2.4) with respect to q_{t-1} yields

$$0 = 2R_{t-1}(u_t^\star - q_{t-1}) + 2B_{t-1}^T P_t B_{t-1} q_{t-1} + 2B_{t-1}^T P_t A_{t-1} x_{t-1} + 2B_{t-1}^T P_t a_t + B_{t-1}^T p_t,$$

showing that the optimal controller is linear affine in the state variable:

$$\begin{aligned} q_{t-1} &= (R_{t-1} - B_{t-1}^T P_t B_{t-1})^{-1} (B_{t-1}^T P_t A_{t-1} x_{t-1} + R_{t-1} u_t^\star + B_{t-1}^T P_t a_t + \frac{1}{2} B_{t-1}^T p_t) \\ &= K_t x_{t-1} + k_t, \end{aligned}$$

$$\begin{aligned} K_t &:= (R_{t-1} - B_{t-1}^T P_t B_{t-1})^{-1} B_{t-1}^T P_t A_{t-1}, \\ k_t &:= (R_{t-1} - B_{t-1}^T P_t B_{t-1})^{-1} (R_{t-1} u_t^\star + B_{t-1}^T P_t a_t + \frac{1}{2} B_{t-1}^T p_t). \end{aligned}$$

The results can finally be used to compute the value function $V_{t-1}(.)$:

$$\begin{aligned} V_{t-1}(x_{t-1}) =& \|x^\star_{t-1} - x_{t-1}\|^2_{Q_{t-1}} + \|u^\star_t - K_{t-1}x_{t-1}\|^2_{R_{t-1}} \\ &+ (A_{t-1}x_{t-1} + B_{t-1}K_{t-1}x_{t-1} + a_{t-1})^T P_t (A_{t-1}x_{t-1} + B_{t-1}K_{t-1}x_{t-1} + a_{t-1}) \\ &+ p_t^T (A_{t-1}x_{t-1} + B_{t-1}K_{t-1}x_{t-1} + a_{t-1}) + v_t \\ =& x^T_{t-1}P_{t-1}x_{t-1} + p^T_{t-1}x_{t-1} + v_{t-1} \end{aligned}$$

$$\begin{aligned} P_{t-1} =& Q_{t-1} + K^T_{t-1}R_{t-1}K_{t-1} + A^T_{t-1}P_tA_{t-1} + K^T_{t-1}B^T_{t-1}P_tB_{t-1}K_{t-1} \\ &+ K^T_{t-1}B^T_{t-1}P_tA_{t-1} + A^T_{t-1}P_tB_{t-1}K_{t-1} \end{aligned}$$

$$\begin{aligned} p_{t-1} =& -2Q_{t-1}x^\star_{t-1} - 2K^T_{t-1}u^\star_t + 2A^T_{t-1}P_ta_{t-1} + 2K^T_{t-1}B^T_{t-1}P_ta_{t-1} \\ &+ (A^T_{t-1} + K^T_{t-1}B^T_{t-1})p_t \end{aligned}$$

The problem can be further extended to stochastic dynamics of the type

$$x_{t+1} = A_tx_t + B_tu_t + a_t + \epsilon_t \qquad \epsilon_t \underset{\text{i.i.d}}{\sim} p(\epsilon), \quad \mathbb{E}[\epsilon] = 0,\ \mathbb{V}[\epsilon] = \sigma^2\mathfrak{I},$$

where ϵ_t is an external noise.

Changing (2.4) to the expectation of the cost,

$$\begin{aligned} V_{t-1}(x_{t-1}) = \min_{q_t}\ \mathbb{E}_{\epsilon_t}\Big[& \|x^\star_{t-1} - x_{t-1}\|^2_{Q_{t-1}} + \|u^\star_t - q_{t-1}\|^2_{R_{t-1}} \\ & + (A_{t-1}x_{t-1} + B_{t-1}q_{t-1} + a_{t-1} + \epsilon_t)^T \\ & \cdot P_t(A_{t-1}x_{t-1} + B_{t-1}q_{t-1} + a_{t-1} + \epsilon_t) \\ & + p_t^T(A_{t-1}x_{t-1} + B_{t-1}q_{t-1} + a_{t-1} + \epsilon_t) + v_t \Big], \end{aligned} \tag{2.5}$$

all calculations stay the same, as the noise contribution cancels out due to zero mean.

2.1.2 Constrained Linear Quadratic Model Predictive Control

The general linear constrained quadratic problem is defined as

$$\begin{aligned} \min_{\boldsymbol{s},\boldsymbol{q}} \quad & \sum_{t=1}^{T} \left(\|x_t^\star - s_t\|_{Q_t}^2 + \|u_t^\star - q_t\|_{R_t}^2 \right) + \|s_{T+1} - x_{T+1}^\star\|_{Q_{T+1}}^2 \\ \text{s.t.} \quad & s_t - x_0 = 0, \\ & t \in \{1, \dots, T\}, \\ & A_t s_t + B_t q_t + a_t - s_{t+1} = 0, \\ & C_t^s s_t + C_t^q q_t \leq c_t, \\ & u_{\min} \leq q_t \leq u_{\max}, \end{aligned} \tag{2.6}$$

$H_t^s \in \mathbb{R}^{n_c \times n_s}, H_t^q \in \mathbb{R}^{n_c \times n_q}$.

Although DP is very efficient in the case of LQR-problems, it is less useful in presence of constraints. The reason is that the value function

$$V_t(x_t) = \begin{cases} \min_{q_t} & \|x_t^\star - x_t\|_{Q_t}^2 + \|u_t^\star - q_t\|_{R_t}^2 + V_{t+1}(A_t x_t + B_t q_t + a_t) \\ \text{s.t.} & C_t^s s_t + C_t^q q_t \leq c_t \\ & u_{\min} \leq q_t \leq u_{\max} \end{cases}$$

is in general not quadratic and can be shown to lead to piecewise linear-affine optimal control-laws [Bemporad et al., 2002]. As the domain of each of these controllers is a polyhedron defined by the inactive and active constraints, explicit calculation usually becomes unwieldy even for small state space dimensions [Ferreau et al., 2008].
Hence, for online control of linear systems subject to constraints, the quadratic programs (2.6) have to be solved online. Compared to the explicit offline calculation of the optimal controllers, this approach is sometimes referred to as *implicit* MPC because the controller is only implicitly given by the solution of the QPs.

As the online solution of the MPC problems has to be computed in real-time, tailored techniques, which exploit the special structure of the QPs, are used. Especially in NMPC this is crucial as linear MPC appears as sub-problem of the *Sequential Quadratic Programming (SQP)* method.

The first step for efficient online computation is the reduction of the MPC-QP (2.6) to a QP in the control-variables $\boldsymbol{q}$ and the first state s_1 only using the so called *Condensing* approach.
An important observation is that all states besides the first one can be expressed by the controls and the initial state

$$\bar{\boldsymbol{s}} := \begin{pmatrix} s_2 \\ \vdots \\ s_{T+1} \end{pmatrix}, \quad \bar{q} := \begin{pmatrix} s_1 \\ \boldsymbol{q} \end{pmatrix}, \quad \bar{\boldsymbol{s}} = \boldsymbol{M}\bar{\boldsymbol{q}} + \boldsymbol{m},$$

$$\boldsymbol{M} = \begin{pmatrix} M_{2|0} & M_{2|1} & & & \\ M_{3|0} & M_{3|1} & M_{3|2} & & \\ \vdots & \vdots & \vdots & \ddots & \\ M_{T+1|0} & M_{T+1|1} & M_{T+1|2} & \dots & M_{T+1|T} \end{pmatrix}, \quad \boldsymbol{m} = \begin{pmatrix} m_2 \\ m_3 \\ \vdots \\ m_{T+1} \end{pmatrix}.$$

The entries of both $\boldsymbol{M}, \boldsymbol{m}$ can be computed recursively using the equality constraint of the dynamic equation $A_t s_t + B_t q_t + a_t - s_{t+1} = 0$:

$$\begin{aligned} M_{2|0} &= A_1, \\ M_{i|0} &= A_{i-1}\, M_{i-1|0} \quad i \in \{3, \dots, T+1\}, \end{aligned}$$

$$\begin{aligned} \forall_{j=1}^{T}\, M_{i+1|j} &= B_j, \\ \forall_{j=1}^{T} \quad M_{i|j} &= A_{i-1}\, M_{i-1|j} \quad i \in \{j+2, \dots, T+1\}, \end{aligned}$$

$$\begin{aligned} m_2 &= a_1, \\ \forall_{i=3}^{T+1} \quad m_i &= A_{i-1} m_{i-1} + a_i. \end{aligned}$$

Finally, let $\boldsymbol{p}_1, \boldsymbol{P}_{11}$ the gradient and Hessian of the MPC-objective with respect to $\bar{\boldsymbol{s}}$, $\boldsymbol{p}_2, \boldsymbol{P}_{22}$ gradient and Hessian w.r.t. $\bar{\boldsymbol{q}}$ and $\boldsymbol{P}_{12}$ denote the joint part of the Hessian. Define $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$ as the concatenation of the constraint matrices w.r.t. to $\bar{\boldsymbol{s}}$ and $\bar{\boldsymbol{q}}$.
Applying $\bar{\boldsymbol{s}} = \boldsymbol{M}\bar{\boldsymbol{q}} + \boldsymbol{m}$ the substitution yields a equivalent condensed QP in $\bar{\boldsymbol{q}}$ only:

$$\begin{aligned} \min_{\bar{q}} \quad & \bar{\boldsymbol{q}}^T \bar{\boldsymbol{P}} \bar{\boldsymbol{q}} + \bar{\boldsymbol{p}}^T \bar{\boldsymbol{q}} \\ \text{s.t.} \quad & s_t - x_0 = 0, \\ & \bar{\boldsymbol{C}} \bar{\boldsymbol{q}} + \bar{\boldsymbol{c}} \leq 0, \\ & \boldsymbol{u}_{\min} \leq \boldsymbol{q} \leq \boldsymbol{u}_{\max}, \end{aligned} \tag{2.7}$$

where we define

$$\begin{aligned}
\bar{\boldsymbol{P}} &= \boldsymbol{M}^T \boldsymbol{P}_{11} \boldsymbol{M} + \boldsymbol{M}^T \boldsymbol{P}_{12} + \boldsymbol{P}_{21} \boldsymbol{M} + \boldsymbol{P}_{22}, \\
\bar{\boldsymbol{p}} &= \boldsymbol{M}^T \boldsymbol{p}_1 + \boldsymbol{p}_2 + 2\boldsymbol{P}_{21} \boldsymbol{m}, \\
\bar{\boldsymbol{C}} &= \boldsymbol{C}_1 \boldsymbol{M} + \boldsymbol{C}_2, \\
\bar{\boldsymbol{c}} &= \boldsymbol{c} + \boldsymbol{C}_1 \boldsymbol{m}.
\end{aligned}$$

[Bock and Plitt, 1984] describe an efficient algorithm for the computation of the parameters of the condensed QP $\bar{\boldsymbol{P}}, \bar{\boldsymbol{p}}, \bar{\boldsymbol{C}}, \bar{\boldsymbol{c}}$ exploiting the block-diagonal structure of $\boldsymbol{P}_{11}, \boldsymbol{P}_{22}, \boldsymbol{P}_{12}, \boldsymbol{C}_1, \boldsymbol{C}_2$ and the block-triangular structure of $\boldsymbol{M}$, which is also used in the implementation of the MPC.

Solving the condensed QP (2.7) produces additionally to $\bar{\boldsymbol{q}}$, the multipliers $\boldsymbol{\mu}, \boldsymbol{\beta}$ of the linear and the bound constraints. If additionally the multipliers of the dynamics of the MPC (2.6) $\boldsymbol{\lambda}$ have to be computed, the so called *Adjoint Variables*, also a recursive technique can be used:
Deriving the Langrangian $\mathscr{L}(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\mu}, \boldsymbol{\beta}, \boldsymbol{\lambda})$ of (2.6) with respect to a certain state s_t leads to

$$\nabla_{s_t} \mathscr{L}(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\mu}, \boldsymbol{\beta}, \boldsymbol{\lambda}) = 2Q_t(s_t - x_t^\star) + A_t^T \lambda_t + [C_t^s]^T \mu_t - \lambda_{t-1} - \beta_t^- + \beta_t^+.$$

From the necessary optimality condition $\nabla \mathscr{L}(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\mu}, \boldsymbol{\beta}, \boldsymbol{\lambda}) \equiv 0$ all adjoints $\boldsymbol{\lambda}$ can be computed by the backwards recursion

$$\lambda_T = 2Q_{T+1}(s_{T+1} - x_{T+1}^\star),$$

$$\lambda_t = 2Q_{t+1}(s_{t+1} - x_{t+1}^\star) + A_{t+1}^T \lambda_{t+1} + [C_{t+1}^s]^T \mu_{t+1}, \qquad t = T-1, \ldots, 1.$$

Parametric Quadratic Programs, Ferreaus' Homotopy-Method and qpOASES

The performance of the MPC-algorithm can be increased further by first solving (2.7) with a prediction of the next start x_0^{pred} and preparing the solver for the measurement. This section introduces the the open-source solver qpOASES[1] developed by Ferreau et al. at Optimization in Engineering Center of KU Leuven published in [Ferreau et al., 2008], which uses a *homotopy method*, to implement this idea.
The method shall be presented in the application to a prototypical QP (the MPC-QP (2.7) can easily brought into the same form), $y \in \mathbb{R}^n, H \in \mathbb{S}^n, g(x_0) \in \mathbb{R}^n, x_0 \in \mathbb{R}^p, c(x_0) \in \mathbb{R}^m, C \in \mathbb{R}^{m \times n}$:

$$\text{QP}(x_0) = \begin{cases} \min_y & \frac{1}{2} y^T H y + y^T g(x_0) \\ \text{s.t.} & Cy \leq c(x_0) \end{cases}. \tag{2.8}$$

[1] www.qpoases.org

First a few prerequisites are needed:
If the constraints of QP(x_0) can be fulfilled, the set of all *feasible* Points

$$\mathscr{F}(x_0) = \{y \in \mathbb{R}^n | Cy \leq c(x_0)\}$$

is not empty, the QP(x_0) is called *feasible*. An important property of $\mathscr{F}(.)$ is that all sets

$$\mathscr{F}(tx_0 + (1-t)x_1), \quad t \in [0,1],$$

are not empy given that $\mathscr{F}(x_0)$ and $\mathscr{F}(x_1)$ are not empty [Bemporad et al., 2002].
Furthermore, for $y \in \mathscr{F}(x_0)$ let $\mathbb{A}(y; x_0) \subset \mathbb{M} := \{1, \ldots, m\}$ be the set of *active* constraints where $C_i y = c_i(x_0)$ holds. The set of *inactive* constraints $C_i y < c_i(x_0)$ is denoted by $\mathbb{I}(y; x_0)$.

The following optimality conditions can be found in any textbook on optimization, e.g. [Geiger and Kanzow, 2002].

Karush-Kuhn-Tucker conditions for convex QPs (KKT) Let H be a positive-definite Matrix and $\mathscr{F}(x_0) \neq \emptyset$. Then there exists a unique $y^\star \in \mathscr{F}(x_0)$ and at least one $\mathbb{A}(y^\star; x_0)$ and one multiplier $\mu^\star \in \mathbb{R}^m$ satisfying the conditions

$$\begin{aligned} Hy^\star + C_{\mathbb{A}}^T \mu_{\mathbb{A}}^\star &= -g(x_0), \\ Cy^\star &\leq c(x_0), \\ \mu_{\mathbb{I}} &= 0, \\ \mu_{\mathbb{A}} &\geq 0. \end{aligned}$$

$y^\star$ is unique minimizer of QP (2.8).

Additionally, for simplicity it is assumed that also $\mathbb{A}(y^\star, x_0)$ is unique and $C_{\mathbb{A}}$ has independent rows, which is called *Linear Independence Constraint Qualification* (LICQ) leading to a unique pair $(y^\star, \mu^\star)$. This is indeed a simplification, but the management of possible exceptions can be found in [Ferreau et al., 2008].
Assuming that the solution of the initial QP(x_0), $(y_0^\star, \mu_0^\star)$, is known and that a new QP(x_1) $\mathscr{F}(x_1) \neq \emptyset$ is to solve. We define

$$\Delta x = x_1 - x_0, \qquad \Delta g = g(x_1) - g(x_0), \qquad \Delta c = c(x_1) - c(x_0)$$

and for $t \in [0,1]$

$$x(t) := x_0 + t\Delta x, \qquad g(t) := g(x_0) + t\Delta g, \qquad c(t) := c(x_0) + t\Delta c.$$

Since $\mathscr{F}(x(t)) \neq \emptyset$, the idea to solve QP($x_1$) for $(y_1^\star, \mu_1^\star)$, is to move from $t = 0$ to $t = 1$ while keeping the KKT-conditions fulfilled, resulting in the *homotopies*

$$\begin{aligned} y^\star(t): &\quad y^\star(0) = y_0^\star, \quad y^\star(1) = y_1^\star, \\ \mu^\star(t): &\quad \mu^\star(0) = \mu_0^\star, \quad \mu^\star(1) = \mu_1^\star, \\ \mathbb{A}(t) := \mathbb{A}(y^\star(t); x(t)): &\quad \mathbb{A}(0) = \mathbb{A}(y_0^\star; x_0), \quad \mathbb{A}(1) = \mathbb{A}(y_1^\star; x_1). \end{aligned}$$

The KKT conditions (2.9) with LICQ result in well defined, piecewise linear and continuous function $y^\star(t)$ and $\mu^\star(t)$ [Bemporad et al., 2002].

$$\begin{aligned} \begin{pmatrix} H & C_{\mathbb{A}(t)}^T \\ C_{\mathbb{A}(t)} & 0 \end{pmatrix} \begin{pmatrix} y^\star(t) \\ \mu_{\mathbb{A}(t)}^\star(t) \end{pmatrix} &= \begin{pmatrix} -g(t) \\ c(t)_{\mathbb{A}(t)} \end{pmatrix}, \\ \mu_{\mathbb{I}(t)}^\star &= 0, \\ C_{\mathbb{I}(t)} y^\star(t) &= c_{\mathbb{I}(t)}(t), \\ \mu^\star(t) &\geq 0 \end{aligned} \tag{2.9}$$

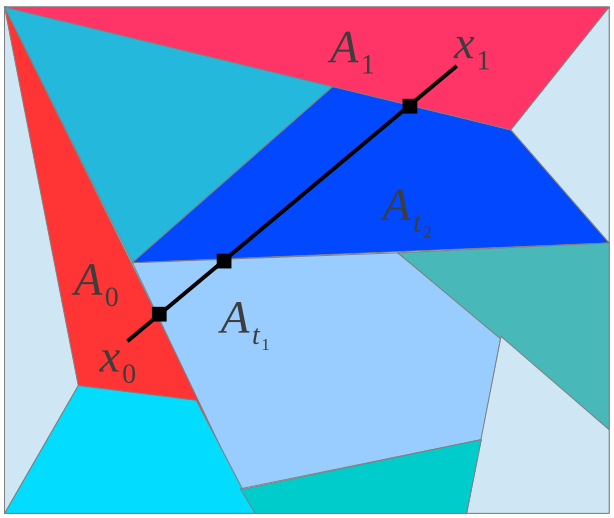

Figure 2.2: Homotopy from x_0 to x_1 with two intermediate changes in the active set $\mathbb{A}(t_1)$ and $\mathbb{A}(t_2)$. Each active set defines a polyhedron with a linear affine optimal feedback-controller [Bemporad et al., 2002]

Starting at $t = 0$ the homotopies are of the form

$$\begin{aligned} y^\star(t) &= y_0^\star + t\Delta y^\star, \\ \mu_{\mathbb{A}(0)}^\star(t) &= \mu_{\mathbb{A}(0)}^\star + t\Delta\mu_{\mathbb{A}(0)}^\star, \\ \mu_{\mathbb{I}(0)}^\star(t) &= \mu_{\mathbb{I}(0)}^\star, \end{aligned}$$

$$\begin{pmatrix} H & C_{\mathbb{A}(0)}^T \\ C_{\mathbb{A}(0)} & 0 \end{pmatrix} \begin{pmatrix} \Delta y^\star \\ \Delta \mu_{\mathbb{A}(0)}^\star \end{pmatrix} = \begin{pmatrix} -\Delta g \\ \Delta c_{\mathbb{A}(0)} \end{pmatrix}, \tag{2.10}$$

as long as the active set $\mathbb{A}(t)$ stays the same .
There are two possible events when a change occurs:

1. An inactive constraint becomes active

$$C_i(y_0^\star + \tau \Delta y^\star) = c_i(0) + \tau \Delta c_i, \tag{2.11}$$

2. An active constraint $\mu_i^\star(0) > 0$ becomes inactive

$$\mu_i^\star(0) + \tau \Delta \mu_i^\star = 0, \tag{2.12}$$

which determines the homotopy stepsize τ as the biggest step until either (2.11) or (2.12) is fulfilled.

The resulting Algorithm can be summarized by

Algorithm 2: Ferreau's Homotopy-Algorithm [Ferreau et al., 2008]

Input : QP(x_0) and solution $(y_0^\star, \mu_0^\star)$, $\mathbb{A}(0)$ and new parameter x_1
Output: Solution $(y_1^\star, \mu_1^\star)$ and $\mathbb{A}(1)$ of QP(x_1)
while $\tau < 1$ **do**
  Calculate Δg and $\Delta c_{\mathbb{A}(0)}$;
  Solve (2.10) for $\Delta y^\star$ and $\Delta \mu_{\mathbb{A}(0)}^\star$ using $\Delta g, \Delta c_{\mathbb{A}(0)}$;
  Determine step τ by checking (2.1.2);
  $\tilde{x} \leftarrow x_0 + \tau \Delta x$
  $\tilde{y}^\star \leftarrow y_0^\star + \tau \Delta y^\star$
  $\tilde{\mu}^\star \leftarrow \mu_0^\star + \tau \Delta \mu^\star$
  if $\tau < 1$ **then**
    Adapt $\mathbb{A}(0)$ according to (2.11) or (2.12);
    Set $x_0 \leftarrow \tilde{x}$, $y_0^\star \leftarrow \tilde{y}^\star$, and $\mu_0^\star \leftarrow \tilde{\mu}^\star$;
  else
    Set $\mathbb{A}(1) \leftarrow \mathbb{A}(0)$, $x_0 \leftarrow \tilde{x}$, $y_0^\star \leftarrow \tilde{y}^\star$ and $\mu_0^\star \leftarrow \tilde{\mu}^\star$;
    STOP;
  end
end

Using a sophisticated management of the factorization of the KKT-System (2.9), the solution to a new QP(x_1) can be computed fast even in case of large problems [Ferreau et al., 2008]. This is especially efficient if x_0 is "close" to x_1 as prediction and measurement in MPC.

The general MPC-Problem

$$\mathrm{MPC}(x) = \begin{cases} \min_{\boldsymbol{s},\boldsymbol{q}} & ||s_{T+1} - x^{\star}_{T+1}||^2_{Q_T} + \sum_{t=1}^{T} \left(\|x^{\star}_t - s_t\|^2_{Q_t} + \|u^{\star}_t - q_t\|^2_{R_t} \right) \\ \text{s.t.} & s_1 - x = 0, \\ & t \in \{1, \dots, T\}, \\ & f(s_t, q_t) - s_{t+1} = 0, \\ & c(s_t, q_t) \leq 0 \end{cases} \tag{2.13}$$

is an instance of a NLP, which can be solved by a variety of algorithms, e.g., interior-point methods. Another approach, which is especially interesting in the case of NMPC, is sequential-quadratic-programming.
Given an initial guess $\boldsymbol{s}^k, \boldsymbol{q}^k$ the next iterate in SQP $\boldsymbol{s}^{k+1} = \boldsymbol{s}^k + \tau \Delta \boldsymbol{s}, \quad \boldsymbol{q}^{k+1} = \boldsymbol{q}^k + \tau \Delta \boldsymbol{q}$ is computed solving the QP

$$\mathrm{QP}_k(x) = \begin{cases} \min_{\Delta \boldsymbol{s}, \Delta \boldsymbol{q}} & \frac{1}{2} \Delta(\boldsymbol{s}^T, \boldsymbol{q}^T) \mathrm{H} \, \Delta(\boldsymbol{s}^T, \boldsymbol{q}^T)^T + \Delta(\boldsymbol{s}^T, \boldsymbol{q}^T) g \\ \text{s.t.} & s^k_1 + \Delta s_1 - x = 0, \\ & i \in \{1, \dots, T\}, \\ & f(s^k_t, q^k_t) + F_s(s^k_t, q^k_t) \Delta s_t + F_q(s^k_t, q^k_t) \Delta q_t - s^k_{t+1} - \Delta s_{t+1} = 0, \\ & c(s^k_t, q^k_t) + C_s(s^k_t, q^k_t) \Delta s_t + C_q(s^k_t, q^k_t) \Delta q_t \leq 0 \end{cases} \tag{2.14}$$

using the Hessian $\mathrm{H} = \nabla^2_{\boldsymbol{s},\boldsymbol{q}} \mathscr{L}$ and gradient $g = \nabla_{\boldsymbol{s},\boldsymbol{q}} \mathscr{L}$ of the Lagrangian

$$\begin{aligned} \mathscr{L}(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = & \sum_{t=1}^{T} \left(\|x^{\star}_t - s_t\|^2_{Q_t} + \|u^{\star}_t - q_t\|^2_{R_t} + \lambda^T_t (f(s_t, q_t) - s_{t+1}) + \mu^T_t c(s_t, q_t) \right) \\ & + ||s_{T+1} - x^{\star}_{T+1}||^2_{Q_T} \end{aligned}$$

defined by the adjoints $\boldsymbol{\lambda}$ of the dynamics and the multipliers $\boldsymbol{\mu}$ of the constraints. For globalization a step size rule to determine τ is needed. The algorithm can be shown to converge globally to a local minimizer and locally quadratic in non-degenerate problems [Geiger and Kanzow, 2002].

2.2.1 Diehl's Suboptimal Scheme for Online NMPC

In real-world applications the exact solution of (2.13) is often computationally too costly. Because usually multiple QPs (2.14) have to be solved, the approach is likely to exceed the time constraints even if efficient solvers are used.
On the other hand, a good initial guess of the NMPC problem is available: The open-loop optimal solution $\boldsymbol{s}^\star, \boldsymbol{q}^\star$ is close to the reference if only small disturbances occur. In this case [Diehl, 2001] suggested to calculate only one SQP step per sampling instance before proceeding to the next instance using the variable-shift initialization (2.3). Although this might seem heuristic, bounds on suboptimality and the deviation from the optimal solution can be derived [Diehl, 2001] and are briefly discussed below:

The first important property of the SQP-method on (2.13) concerns the *solution-manifold*

$$\mathscr{M} = \left\{ (x, \boldsymbol{s}, \boldsymbol{q}) \in \mathbb{R}^{n_x} \times \mathbb{R}^{n_s(T+1)} \times \mathbb{R}^{n_q T} \;\middle|\; (\boldsymbol{s}, \boldsymbol{q}) = \arg\min_{\boldsymbol{s},\boldsymbol{q}} \mathrm{MPC}(x) \right\}.$$

Given the solution $(\boldsymbol{s}, \boldsymbol{q})_0$ of an instance MPC(x_0), the first SQP step $\Delta(\boldsymbol{s}, \boldsymbol{q})$, using $(\boldsymbol{s}, \boldsymbol{q})_0$ as initial guess, on a "neighboring" MPC(x) is proportional to the directional derivative of $(x, \boldsymbol{s}, \boldsymbol{q})_0$ to $(x, \boldsymbol{s}, \boldsymbol{q})$ along $\mathscr{M}$:

$$\|(\boldsymbol{s}, \boldsymbol{q}) - \Delta(\boldsymbol{s}, \boldsymbol{q}) + (\boldsymbol{s}, \boldsymbol{q})_0\| = \mathscr{O}(\|x - x_0\|^2), \tag{2.15}$$

even if the active set changes ([Diehl, 2001], pp.45–48).
This indicates that given the optimal solution to an undisturbed MPC(x_0) problem, one SQP step might already suffice to adapt to moderate disturbances $x_0 + \epsilon$. For linear constraint MPC-problems the manageable disturbances are, intuitively speaking, those in which the linearization in x_0 still captures the true dynamics well enough.

In order to establish a bound on sub-optimality, a series of MPC problems with *shrinking horizon* SH-MPC$_t$ has to be considered:

$$\text{SH-MPC}_t(x_t; x_0) = \begin{cases} \min_{\boldsymbol{s},\boldsymbol{q}} & \|s_{T+1} - x^\star_{T+1}\|^2_{Q_{T+1}} + \sum_{\underline{i=t}}^{T} \left(\|x^\star_i - s_i\|^2_{Q_i} + \|u^\star_i - q_i\|^2_{R_i} \right) \\ \text{s.t.} & s_t - x_t = 0, \\ & i \in \underline{\{t, \ldots, T\}}, \\ & f(s_i, q_i) - s_{i+1} = 0, \\ & c(s_i, q_i) \leq 0. \end{cases}$$

Compared are the exact solution (including multipliers) $(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\mu})^\star$ of SH-MPC$_0(x_0; x_0)$ and the *real-time* solution $(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\mu})^{\text{real}}$, starting at an initial guess $(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\mu})^0$.

The real-time approach consists of one SQP-step for SH-MPC$_t(x_t; x_0)$ and passing over all but the first variables $(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\mu})_{t+1}$ to the next problem SH-MPC$_{t+1}(x_{t+1} = s_{t+1}; x_0)$.

Given a sufficient good initial guess, considering the variables starting at t, it holds:

$$\|(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\mu})_t^\star - (\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\mu})_t^{\text{real}}\| \leq 2\frac{\delta^{t+1}}{1-\delta}\|\Delta(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\mu})^0\|, \qquad \delta = c_1 + c_2\|\Delta(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\mu})^0\|, \tag{2.16}$$

with constants c_1, c_2, where $c_1 \equiv 0$ for exact Hessian and $c_1 > 1$ for SQP based on an approximation to the Hessian and $\|\Delta(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\mu})^0\|$ the size of the first step ([Diehl, 2001], pp.74–87 “The On-Line Problem”). This bound also shows that the approximation framework can increase in performance if the time horizon T is increased.

Both previous results (2.15), (2.16) can be combined to prove a bound on the sub-optimality when using the real-time algorithm on the disturbed problem MPC$(x_0 + \epsilon)$ initializing with $(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\mu})^\star$ of the reference problem MPC(x_0). Comparing the cost of the real-time approach V^{real} to the optimal cost $V^\star$ it holds

$$V^{\text{real}} - V^\star \leq c_2^2\|\epsilon\|^4 \tag{2.17}$$

for an exact Hessian SQP and

$$V^{\text{real}} - V^\star \leq (c_1 + c_2\|\epsilon\|)^2\|\epsilon\|^2 \tag{2.18}$$

for a SQP with an approximation to the Hessian, latent in the constant c_1 ([Diehl, 2001], pp. 87–89).
These theoretical results also have practical consequences for the performance of NMPC algorithm in application and will be readdressed in the discussion (see chapter 6).

2.2.2 Efficient Approximation of the Hessian

So far SQP was only considered in the case that the Hessian H of the Lagrange function $\mathscr{L}(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\mu})$ is known and tractable to compute. However, in NMPC, the computation of the full Hessian

$$\nabla_{s,q} \begin{pmatrix} 2Q_1(x_1^\star - s_1) + F_s^T(s_1, q_1)\lambda_1 + C_s^T(s_1, q_1)\mu_1 - \lambda_0 \\ \vdots \\ 2Q_{T+1}(x_{T+1}^\star - s_{T+1}) - \lambda_T \\ 2R_1(u_1^\star - q_1) + F_q^T(s_1, q_1)\lambda_1 + C_q^T(s_1, q_1)\mu_1 \\ \vdots \\ 2R_T(u_T^\star - q_T) + F_q^T(s_T, q_T)\lambda_T + C_q^T(s_T, q_T)\mu_T \end{pmatrix} \tag{2.19}$$

is too demanding (especially if already the Jacobians $F(s_t, q_t)$ involve numerical computations). In the following, we will discuss two standard approximations to the Hessian, which have be suggested for NMPC.

Broyden-Fletcher-Goldfarb-Shanno Update

A widely used technique in nonlinear programming is the *Broyden-Fletcher-Goldfarb-Shanno* (BFGS) rank 2-Update for a approximation $\tilde{\mathrm{H}}$ to the Hessian [Broyden, 1970] [Fletcher, 1970] [Goldfarb, 1970] [Shanno, 1970]. This rule is well suited for NMPC as the dependence-structure in the gradient of the Lagrangian can be exploited. As only s_t and q_t cooccur in the gradient the Hessian has a block-diagonal structure and, therefore, updates can be done block-wise:
Let d_t be the recent step in both s_t, q_t and y_t be the difference of the gradients of the Lagrangian

$$y_t := \nabla_{s_t,q_t}\mathscr{L}((\boldsymbol{s}, \boldsymbol{q}) + \boldsymbol{d}, \boldsymbol{\lambda}, \boldsymbol{\mu}) - \nabla_{s_t,q_t}\mathscr{L}((\boldsymbol{s}, \boldsymbol{q}), \boldsymbol{\lambda}, \boldsymbol{\mu}).$$

The approximation to the Hessian block $\tilde{\mathrm{H}}_t$ is updated using

$$\tilde{\mathrm{H}}_t \leftarrow \tilde{\mathrm{H}}_t + \frac{y_t y_t^T}{d_t^T y_t} - \frac{\tilde{\mathrm{H}}_t d_t d_t^T \tilde{\mathrm{H}}_t}{d_t^T \tilde{\mathrm{H}}_t d_t}.$$

If $\tilde{\mathrm{H}}_t$ is positive definite, the update is also positive definite, provided $d_t^T y_t > 0$.
In implementation, Powell's safeguard-rule

$$\begin{aligned} \tilde{y}_t &= \gamma y_t + (1-\gamma)\tilde{\mathrm{H}}_t d_t \\ \gamma &= \begin{cases} 1 & \text{if} \quad d_t^T y_t \geq 0.2\, d_t^T \tilde{\mathrm{H}}_t d_t \\ \frac{0.8\, d_t^T \tilde{\mathrm{H}}_t d_t}{d_t^T \tilde{\mathrm{H}}_t d_t - d_t^T y_t} & \text{else} \end{cases}, \end{aligned}$$

$$\tilde{\mathrm{H}}_t \leftarrow \tilde{\mathrm{H}}_t + \frac{\tilde{y}_t \tilde{y}_t^T}{d_t^T \tilde{y}_t} - \frac{\tilde{\mathrm{H}}_t d_t d_t^T \tilde{\mathrm{H}}_t}{d_t^T \tilde{\mathrm{H}}_t d_t},$$

ensures positive definite updates. The approximation can be proven to converge to the true Hessian in direction of the step leading to super-linear local convergence [Geiger and Kanzow, 2002]. For the real-time iteration this results in constants c_1 in (2.16), (2.18) approaching 0 with increasing optimization horizon T as a direct consequence of the proofs in [Diehl, 2001] chapter "Contractivity of the Real-Time Iterations".

Gauss-Newton Approximation

Another approximation method, especially suited for NMPC, is the Gauss-Newton approach. Given that the NMPC problem has only box-constraints $\boldsymbol{u}_{\min} \leq \boldsymbol{q} \leq \boldsymbol{u}_{\max}$, it holds

$$\nabla_{\boldsymbol{s},\boldsymbol{q}} \begin{pmatrix} 2Q_1(x_1^\star - s_1) + F_s^T(s_1, q_1)\lambda_1 - \lambda_0 \\ \vdots \\ 2Q_{T+1}(x_{T+1}^\star - s_{T+1}) - \lambda_T \\ 2R_1(u_1^\star - q_1) + F_q^T(s_1, q_1)\lambda_1 + \beta_t^+ - \beta_t^- \\ \vdots \\ 2R_T(u_T^\star - q_T) + F_q^T(s_T, q_T)\lambda_T + \beta_T^+ - \beta_T^- \end{pmatrix} \approx \nabla_{\boldsymbol{s},\boldsymbol{q}} \begin{pmatrix} 2Q_1(x_1^\star - s_1) \\ \vdots \\ 2Q_{T+1}(x_{T+1}^\star - s_{T+1}) \\ 2R_1(u_1^\star - q_1) \\ \vdots \\ 2R_T(u_T^\star - q_T) \end{pmatrix} = \tilde{\mathrm{H}}_{\mathrm{GN}}.$$

From the necessary optimality condition it follows:

$$\begin{pmatrix} 2Q_t(x_t^\star - s_t) + F_s^T(s_t, q_t)\lambda_t - \lambda_{t-1} \\ 2R_t(u_t^\star - q_t) + F_q^T(s_t, q_t)\lambda_t + \beta_t^+ - \beta_t^- \end{pmatrix} \equiv 0.$$

Assuming full rank of $\boldsymbol{Q}, \boldsymbol{F}_s^T$ this, starting in $2Q_{T+1}(x_{T+1}^\star - s_{T+1}) - \lambda_T$, recursively yields $\lambda_t = \mathcal{O}(\|x_{t,\dots,T+1}^\star - s_{t,\dots,T+1}\| + \|u_{t,\dots,T}^\star - q_{t,\dots,T}\|)$. Therefore it holds

$$\|\mathrm{H}_t - [\tilde{\mathrm{H}}_{\mathrm{GN}}]_t\| = \|\sum_{i=1}^{n_s} [\lambda_t]_i \nabla^2 [F_s]_i(s_t, q_t)\| = \mathcal{O}(\|\boldsymbol{x}^\star - \boldsymbol{s}\| + \|\boldsymbol{u}^\star - \boldsymbol{q}\|).$$

Therefore, the approximation quality is good if planned states and controls are close to the reference, which is also the goal of the NMPC approach.
[Kosmol, 1993] shows local quadratic convergence for optimal residuals $\|\boldsymbol{x}^\star - \boldsymbol{s}^\star\| + \|\boldsymbol{u}^\star - \boldsymbol{q}^\star\| = 0$ and [Stoer et al., 2004] linear convergence for small residuals. Especially in [Diehl, 2001] it is shown that this approach is a Newton-type algorithm in the sense that it is suitable for the realtime-iteration ("The Constrained Gauss-Newton Method" pp. 68–73).
In the realtime-approach the constants c_1 in (2.16), (2.18) can be shown to decrease in $\|\boldsymbol{x}^\star - \boldsymbol{s}\| + \|\boldsymbol{u}^\star - \boldsymbol{q}^\star\|$.

3 Bayesian Regression

Regression is the task of inferring a model $f : \mathbb{R}^{n_{\text{input}}} \to \mathbb{R}^{n_{\text{target}}}$

$$y = f(x) + \epsilon$$

$y, \epsilon \in \mathbb{R}^{n_{\text{target}}}, x \in \mathbb{R}^{n_{\text{input}}}$, $\epsilon \sim \mathcal{N}(0, \sigma^2 \mathfrak{I})$ from observed Data $D = (Y, X)$, where $Y \in \mathbb{R}^{n_{\text{target}} \times d}, X \in \mathbb{R}^{n_{\text{input}} \times d}$. The joint density x, ϵ is assumed to be of the form $p(x, \epsilon) = p(x)p(\epsilon)$. Therefore, ϵ and x are independent, and ϵ is considered an unknown external influence. The data is assumed to be *independently* distributed

$$p(Y, X) = \prod_{i=1}^{d} p(y_i, x_i). \tag{3.1}$$

The problem of regression is amongst the oldest in statistics and the basic approach linear *Least-Squares-Estimation* can be dated back to works of Legendre and Gauss in 1805 and 1809, respectively.
The basic assumption is that the true function is of the form $f(x)_i = \theta_i^T \phi(x)$, where the *feature-map* $\phi : \mathbb{R}^{n_{\text{input}}} \to \mathbb{R}^p$ serves to extract important information of the input and $\Theta := (\theta_1, \ldots, \theta_{n_{\text{target}}})^T \in \mathbb{R}^{n_{\text{target}} \times p}$ is a matrix containing all parameters.
It is easy to prove that for the conditional expectation $\mathbb{E}[y|x] = f(x)$ it holds

$$f = \underset{g : \mathbb{R}^{n_{\text{input}}} \to \mathbb{R}^{n_{\text{target}}}}{\arg\min} \int \|g(x) - y\|^2 p(y|x) p(x) \, \mathrm{d}(x, y), \tag{3.2}$$

for sufficient regular (measurable) g.
The least-squares objective

$$\min_{\Theta} \sum_{i=1}^{d} \|y_i - \Theta \phi(x_i)\|^2 \tag{3.3}$$

can be seen as a finite sample approximation to (3.2). Approximation quality and convergence in the limit of infinitely many samples can be found in textbooks on mathematical statistics, e.g., [Van De Geer, 2000] (147 ff., Convergence Rates for Least Squares Estimation).
Using the operator vec and $\Phi(X) = (\phi(X)^T \otimes \mathfrak{I}^{n_{\text{target}}})$ the solution can be written in the form

$$\begin{aligned} \operatorname{vec}(\Theta^\star) &= \arg\min_{\Theta} \|\operatorname{vec}(X) - \Phi(X) \operatorname{vec}(\Theta)\|^2, \\ \operatorname{vec}(\Theta^\star) &= \left(\Phi(X)^T \Phi(X)\right)^{-1} \Phi(X)^T \operatorname{vec}(Y). \end{aligned}$$

If regularization of Θ is desired, the equations become

$$\begin{aligned} \operatorname{vec}(\Theta^\star) &= \arg\min_{\operatorname{vec}(\Theta)} \|\operatorname{vec}(Y) - \Phi(X)\operatorname{vec}(\Theta)\|^2 + \rho\|\operatorname{vec}(\Theta)\|^2, \\ \operatorname{vec}(\Theta^\star) &= \left(\Phi(X)^T\Phi(X) + \rho\mathfrak{I}\right)^{-1}\Phi(X)^T\operatorname{vec}(Y) \end{aligned} \tag{3.4}$$

with a regularization parameter ρ.

3.1 Bayesian Linear Regression

Bayesian Linear Regression is motivated differently from the minimization approaches to least-squares (3.3),(3.4), but will end up in the same results.

Again the assumption is that $f(x) = \Theta\phi(x)$. In addition to that a-priory information of the parameters, $\operatorname{vec}(\Theta) \sim \mathcal{N}(0, \Sigma_\Theta)$ is available. Instead of minimizing a loss-objective Bayes' rule is applied to calculate the posterior $p(\Theta|Y,X)$:

$$\underbrace{p(\Theta|Y,X)}_{posterior} = \frac{\overbrace{p(Y|\Theta,X)}^{likelihood}\overbrace{p(\Theta)}^{prior}}{\int p(Y|\Theta,X)p(\Theta)\,\mathrm{d}\,\Theta}.$$

As $p(Y|\Theta,X) = \prod_i^d \mathcal{N}(\Theta\phi(x_i), \sigma^2\mathfrak{I})$ the posterior results in

$$p(\operatorname{vec}(\Theta)|Y,X) = \mathcal{N}\left(\operatorname{vec}(\Theta)^\star, \Sigma^\star\right)$$

$$\begin{aligned} \operatorname{vec}(\Theta)^\star &= S\operatorname{vec}(Y), \\ \Sigma^\star &= \Sigma_\Theta - S\Phi(X)\Sigma_\Theta, \\ S &= \Sigma_\Theta\Phi(X)^T\left(\Phi(X)^T\Sigma_\Theta\Phi(X) + \sigma^2\mathfrak{I}\right)^{-1}, \end{aligned}$$

using well-know Gaussian identities [Bishop and Nasrabadi, 2006] [Barber, 2012].

As this result seems cryptic on the first glance, the role of the parameters involved will be explained:

- Choosing $\Sigma_\Theta = \frac{\sigma^2}{\rho}\mathfrak{I}$, $\operatorname{vec}(\Theta)^\star$ coincides with the minimizer of the regularized least-squares (3.4), therefore, the *Signal-to-Noise Ratio* $\frac{\sigma^2}{\|\Sigma_\theta\|}$ determines the amount of regularization.
- The covariance $\Sigma^\star$ of the prediction is decreasing in the prior covariance $\Sigma_\theta - \Sigma^\star \succeq 0$.
- $\Sigma^\star$ decreases, if the norm of the *Gramian* $\Phi(X)^T\Phi(X)$ increases. This happens when the number of observations d grows, or if the $\phi(x_i)$ are spread. Therefore, increasing the cover of the input space decreases the predictive uncertainty.

Hence, using the Bayesian approach has the advantage that $\Sigma^\star$ provides additional information of the data. Especially in cases where the amount of data is limited, exploitation of the predictive uncertainty can make further steps building on the linear regression model more robust and reliable.

Kalman-Learning of Series of Linear Systems on Trajectory Data

In a control task, where the dynamical model is not known, regression can be used to infer the true model for trajectory data $D_t = (X_t, U_t)$. However, there are problems when using linear Bayesian regression:

- The assumption of independence of the data is no longer valid (3.1), as x_{t+1} is strongly dependent on x_t, u_t and, therefore, using both $(x_{t+2}; x_{t+1}, u_{t+1})$ and $(x_{t+1}; x_t, u_t)$ in the data is a violation from a theoretical perspective
- In the case that the true dynamics are nonlinear, a single linear model for the entire time horizon might not be flexible enough.
- There might also be not enough data to learn a separate model for each time step.

A simple idea to learn a series of linear models on trajectory data, *that hasn't been considered so far in literature* to the best of our knowledge, is to use the Rauch-Tung-Striebel smoother [Rauch et al., 1965] and utilize that the system matrices usually do not change much in one time step.

Let us assume a state-space model for the system matrices:

$$\mathrm{vec}(\Theta_{t+1}) = \mathfrak{I}\,\mathrm{vec}(\Theta_{t+1}) + \xi_t, \qquad \xi_t \underset{\text{i.i.d}}{\sim} \mathcal{N}(0, \Sigma_\xi),$$

codes the knowledge about the slowly varying system matrix starting from an initial distribution $\mathcal{N}(\mu_0^\Theta, \Sigma_0^\Theta)$.
The likelihood

$$p(X_{t+1}|\Theta_t, X_t, U_t) \quad = \quad \prod_{i=1}^{d} \mathcal{N}(x_{t+1}^i|\Theta_t \begin{pmatrix} x_t^i \\ u_t^i \end{pmatrix}, \Sigma_t)$$

can be used to define an observation model:

$$\mathrm{vec}(X)_{t+1} = G_t\,\mathrm{vec}(\Theta)_t + \zeta_t, \qquad \zeta_t \underset{\text{i.i.d}}{\sim} \mathcal{N}(0, \Sigma_\zeta^t), \qquad G_t := ((X_t^T, U_t^T) \otimes \mathfrak{I}^{n_x})$$
$$\Sigma_\zeta^t = \underbrace{\mathrm{diag}(\Sigma_\zeta, \ldots, \Sigma_\zeta)}_{d_t := \#\text{ of observations at } t}.$$

Thus, the filter and smoothing equations can be derived from the general solution [Bishop and Nasrabadi, 2006] (pp. 637- 641):

1. **Forward-Pass:**

   a) Prediction:

$$\begin{aligned}\mu_{t|t-1} &= \mu_{t-1},\\ \Sigma_{t|t-1} &= \Sigma_{t-1} + \Sigma_\xi.\end{aligned}$$

   b) Update:

$$\begin{aligned}\mu_t &= \mu_{t|t-1} + K_t r_t,\\ \Sigma_t &= (\mathfrak{I} - K_t G_t)\Sigma_{t|t-1},\end{aligned}$$

$$\begin{aligned}r_t &:= \mathrm{vec}(X)_{t+1} - G_t \mu_{t|t-1},\\ K_t &:= \Sigma_{t|t-1} G_t^T \left(\Sigma_\zeta + G_t \Sigma_{t|t-1} G_t^T\right)^{-1}.\end{aligned}$$

2. **Backward-Pass:**

$$\begin{aligned}\mu_{t|T} &= \mu_t + J_t(\mu_{t+1|T} - \mu_{t+1|t}),\\ \Sigma_{t|T} &= \Sigma_t + J_t(\Sigma_{t+1|T} - \Sigma_{t+1|t})J_t^T,\\ J_t &:= \Sigma_t \Sigma_{t+1|t}^{-1}.\end{aligned}$$

The open parameters $\mu_0^\Theta, \Sigma_0^\Theta, \Sigma_\xi, \Sigma_t$ can be inferred using *Expectation-Maximization (EM)* as suggested by [Ghahramani and Hinton, 1996]:

To maximize the *so-called* incomplete log-likelihood

$$\log \int p(\boldsymbol{X}, \boldsymbol{\Theta} | \boldsymbol{X}; \mu_0^\Theta, \Sigma_0^\Theta, \Sigma_\xi, \Sigma_t)\, \mathrm{d}\,\boldsymbol{\Theta}$$

w.r.t. $\mu_0^\Theta, \Sigma_0^\Theta, \Sigma_\xi, \Sigma_\zeta$, EM iterates switching between a forward and backward-pass (E-Step) and the update (M-Step)

$$\begin{aligned}\mu_0^\Theta &= \mathbb{E}[\Theta_0], \quad \Sigma_0^\Theta = \mathbb{V}[\Theta_0],\\ \Sigma_\xi &= \frac{1}{N}\sum_{t=1}^{N}\Big(\mathbb{E}[\mathrm{vec}(\Theta)_t \,\mathrm{vec}(\Theta)_t^T] - \mathbb{V}[\mathrm{vec}(\Theta)_t] J_{t-1}^T - \mathbb{E}[\mathrm{vec}(\Theta)_t]\mathbb{E}[\mathrm{vec}(\Theta)_{t-1}]^T\\ &\quad - (\mathbb{V}[\Theta_t] J_{t-1}^T + \mathbb{E}[\mathrm{vec}(\Theta)_t]\mathbb{E}[\mathrm{vec}(\Theta)_{t-1}]^T)^T + \mathbb{E}[\mathrm{vec}(\Theta)_{t-1}\,\mathrm{vec}(\Theta)_{t-1}^T]\Big),\end{aligned}$$

$$\Sigma_\zeta = \frac{1}{N}\sum_{t=1}^{N}\frac{1}{d_t}\sum_{j=1}^{d_t}\Big([x_{t+1}]_j[x_{t+1}]_j^T - G_{t,j}\mathbb{E}[\mathrm{vec}(\Theta)_t][x_{t+1}]_j^T$$
$$- [x_{t+1}]_j\mathbb{E}[\mathrm{vec}(\Theta)_t]^T G_{t,j}^T + G_{t,j}\mathbb{E}[\mathrm{vec}(\Theta)_t\,\mathrm{vec}(\Theta)_t^T]G_{t,j}^T\Big),$$
$$G_{t,j} = (([x_t]_j^T, [u_t]_j^T)\otimes \mathfrak{I}^{n_x}),$$

which results in a sequence converging to a local maximizer [Dempster et al., 1977], [Wu, 1983] ([Bishop and Nasrabadi, 2006], chapter "Mixture Models and EM" for a illustrative introduction).

3.2 Gaussian Processes

GPs are a family of non-parametric regression methods, which extend the ideas of Bayesian linear regression. Their basis was founded by the works of Wiener [Wiener, 1949] and Kolmorgorov [Kolmogorov, 1941]. The first applications came up in geostatistics in the 1950s [Krige, 1951] and they are nowadays one of the state-of-the-art approaches in machine learning [Rasmussen and Williams, 2006] for regression and classification.

3.2.1 Gaussian Process Regression

Similar to linear Bayesian regression, Gaussian Process Regression is not motivated by minimization of a certain risk but from an assumption on the conditional distribution $p(Y|\boldsymbol{x})$ of the observations. Given a dataset $D = (Y,X)$ with *one-dimensional* targets $Y \in \mathbb{R}^{d\times 1}$ and multidimensional inputs $X \in \mathbb{R}^{n\times d}$ the following joint conditional distribution is assumed:

$$p(y|X) = \mathcal{N}(\mathbf{0}, \mathfrak{K})$$
$$\mathfrak{K} = \begin{pmatrix} \mathfrak{k}(x_1,x_1) & \mathfrak{k}(x_1,x_2) & \cdots & \mathfrak{k}(x_1,x_n) \\ \mathfrak{k}(x_2,x_1) & \mathfrak{k}(x_2,x_2) & \cdots & \mathfrak{k}(x_2,x_n) \\ \vdots & \vdots & \ddots & \vdots \\ \mathfrak{k}(x_n,x_1) & \mathfrak{k}(x_n,x_2) & \cdots & \mathfrak{k}(x_n,x_n) \end{pmatrix}, \tag{3.5}$$

where the symmetric $\mathfrak{k}(.,.) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_0^+$ is a *kernel-function*, i.e., the kernel-matrix $\mathfrak{K}$ (3.5) of any finite collection of $\tilde{X} \subset \mathbb{R}^n$ is positive-definite. The kernel-function $\mathfrak{k}$ is chosen, such that it captures the important correlation in the data.

Given a new input x', applying Gaussian conditioning [Bishop and Nasrabadi, 2006], the GP yields a *posterior* distribution

$$p\left(y'|x',Y,X\right) = \mathcal{N}\left(\mathfrak{k}(x',X)\mathfrak{K}^{-1}Y \,\middle|\, \mathfrak{k}(x',x') - \mathfrak{k}(x',X)\mathfrak{K}^{-1}\mathfrak{k}(X,x')\right), \tag{3.6}$$

which expresses the predictive uncertainty dependent on the correlations of x' and the given data X.

The widely used *Squared Exponential (Kernel) with Automatic Relevance Detection (SEARD)* is defined as

$$\mathfrak{k}(x_1, x_2) = \exp\big(-\frac{1}{2}(x_1 - x_2)^T L^{-1}(x_1 - x_2)\big),$$

where $L = \mathrm{diag}(l_1^2, \ldots, l_n^2)$ contains the squared characteristic scales of the different input-dimensions. Figure 3.1, taken from [Rasmussen and Williams, 2006], illustrates the equations

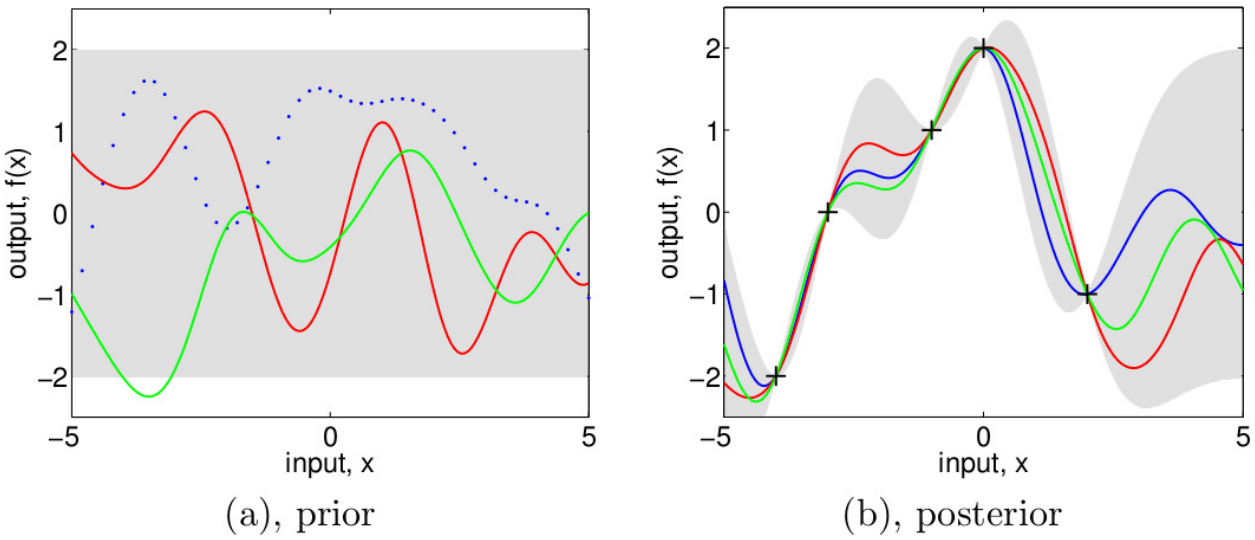

Figure 3.1: [Rasmussen and Williams, 2006] (a) Prior samples $f(x)$ of a dense set X and (b) posterior samples given some observations, using the SEARD kernel. Gray indicates the $\mu \pm 2\sigma$ area.

from above. In absence of observations the sampled values of a dense set $X \in \mathbb{R}$ form the shape of a smooth function[1], which can be seen as an underlying prior assumption on the candidates of interpolation functions. If observations D are given, the functions interpolate the data and vary in regions distant from the observed points. This behavior is an immediate consequence from the predictive variance, which can, therefore, be used to asses the predictive uncertainty.

3.2.2 Model Selection

In application, the observed data often contains additional unknown measurement noise, which can be incorporated by a slight modification of the kernel matrix of the data $\mathfrak{K}(\boldsymbol{x})$:

$$\tilde{\mathfrak{K}}(X) = \alpha^2 \mathfrak{K}(X; L) + \sigma^2 \mathfrak{I}.$$

As α, σ and also L have an influence (compare figure 3.2) on the quality of the GP prediction, a natural question is how to set them best. A commonly used method, first suggested in [Mardia and

[1] This should only serve as an intuition. In fact even a very dense input set will produce sample functions with oscillation, which are simply not visible in this figure. But as never explicit samples are used and because of the smooth predictive mean function $\mathbb{E}(y'|x')$, this is neglect-able in application.

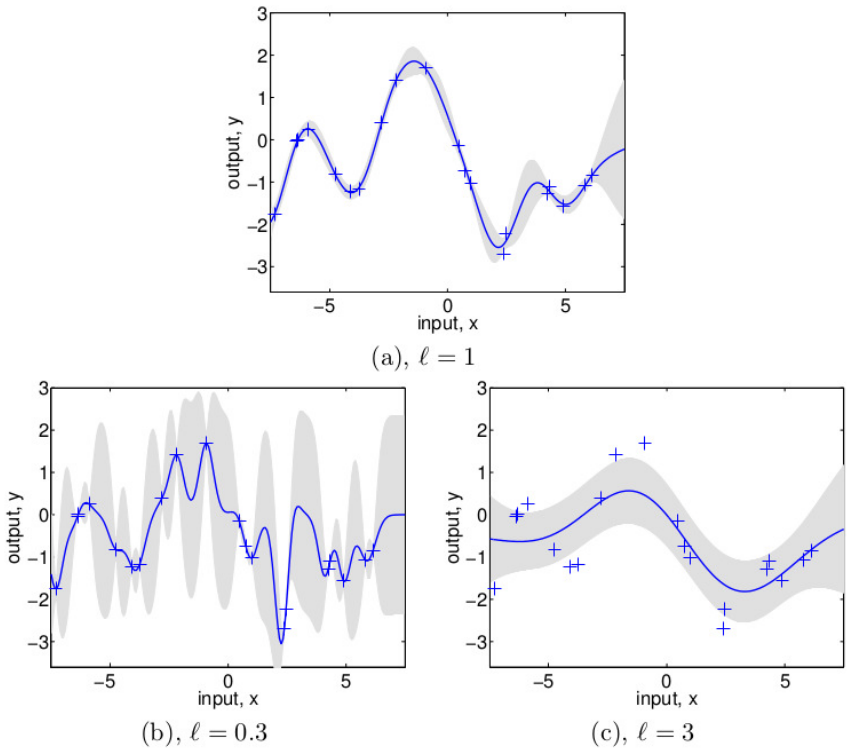

Figure 3.2: [Rasmussen and Williams, 2006] (a)–(c) show the influence of the length-scale l on the model quality. A Small l results in strong oscillations, whereas a big l smooths out fine structure .

Marshall, 1984], to find the best *hyper-parameters* is to maximize the *marginal likelihood* of the observed data

$$\log p(Y|X;\alpha,\sigma,L) = -\frac{1}{2}Y^T\tilde{\mathfrak{K}}(X;\alpha,\sigma,L)Y - \frac{1}{2}\det\left(\tilde{\mathfrak{K}}(X;\alpha,\sigma,L)\right) - \frac{n}{2}\log 2\pi. \tag{3.7}$$

Given the derivatives of the kernel $\tilde{\mathfrak{K}}$ with respect to L,α,σ, standard solvers for unconstrained optimization problems can be used to compute the optimal $(\alpha,\sigma,L)^\star$. As the objective is not concave, however a good initialization has to be provided.

4 Control with Probabilistic Models

In the case that an unknown system has to be controlled along a predefined reference, the methods of both previous chapters can be combined. However, as MPC relies on a good model of the dynamics, using a regression model on a small amount of trajectory data can be dangerous. In model-based reinforcement learning, which consists roughly speaking of learning a model and optimizing a controller, it is reported that this approach can induce bias and lead to non-robust controllers [Atkeson and Santamaria, 1997].
Recently, GPs have been used for different control tasks . Although Bayesian regression techniques give posterior distributions most of these methods do not use this additional information for robust control. However, in [Deisenroth and Rasmussen, 2011] a method for reinforcement learning is presented that strongly exploits the predictive uncertainty leading to a very data-efficient learning, also in real application. This approach was further extended to imitation learning [Englert et al., 2013].
The next sections show how the same ideas can be used for MPC with both Bayesian linear models and GPs.

4.1 Stochastic Optimal Control

The high level idea is to solve *stochastic* instead of deterministic optimal control problems

$$\text{S-MPC}(x_t) = \begin{cases} \min_{\boldsymbol{q},\boldsymbol{\epsilon}} & \mathbb{E}_{\boldsymbol{\epsilon}}\Big[||s^t_{T+1} - x^\star_{T+t}||^2_{Q_{t+T}} \\ & \qquad + \sum_{i=1}^{T} \left(\|x^\star_{i-1+t} - s^t_i\|^2_{Q_{t-1+i}} + \|u^\star_{i-1+t} - q^t_i\|^2_{R_{t-1+i}} \right) \Big] \\ \text{s.t.} & s^t_1 - x_t = 0, \\ & i \in \{1, \dots, T\}, \\ & f(s^t_i, q^t_i) + \epsilon_t = s^t_{i+1}, \qquad \epsilon_t \underset{\text{i.i.d}}{\sim} \mathcal{N}(0, \Sigma_\epsilon), \end{cases} \tag{4.1}$$

where ϵ_t is a external noise.

A very important difference to deterministic open-loop optimal control is, that $q^\star_t$ always depends on the realizations of $\boldsymbol{\epsilon}$: The solution of (4.1) is necessarily a controller $q_t = \pi_t(\epsilon_{1,\dots,t})$. A basic result from the theory of stochastic control [Bertsekas and Shreve, 1978] (see also [Gruene, 2007]), is that the optimal controller depends on the recent only state:

$$\pi^\star_t(\epsilon_{1,\dots,t}) \equiv \pi^\star_t(s_t).$$

Although there exists a stochastic maximum principle it is only applicable in special problems [Gruene, 2007] as the random variables induce complex dependencies. Therefore, the standard approach is DP. For nonlinear dynamics hereby usually approximation techniques, which discretize the state space and take use of function approximation, must be used [Bertsekas and Tsitsiklis, 1996].

The problem can be further extended to dynamical models with stochastic parameter uncertainty

$$\text{BS-MPC}(x_t) = \begin{cases} \min_{\boldsymbol{q}(\boldsymbol{s})} & \mathbb{E}_{\boldsymbol{\epsilon},\boldsymbol{\theta}}\Big[||s^t_{T+1} - x^\star_{T+t}||^2_{Q_{t+T}} \\ & \quad + \sum_{i=1}^{T} \left(\|x^\star_{i-1+t} - s^t_i\|^2_{Q_{t-1+i}} + \|u^\star_{i-1+t} - q^t_i\|^2_{R_{t-1+i}} \right) \Big] \\ \text{s.t.} & s^t_1 - x_t = 0, \\ & i \in \{1, \dots, T\}, \\ & f(s^t_i, q^t_i; \theta_t) + \epsilon_t = s^t_{i+1}, \end{cases} \tag{4.2}$$

if we include the posterior distribution on the independent parameters θ_t[1]. If the problem (4.2) can be solved for an optimal controller $\pi^\star_t(.)$, we can expect it to be robust with respect to *both* external disturbances and model errors.

4.2 Uncertain Linear Quadratic Regulator

Although the general problem (4.2) is in most cases intractable to solve exactly, DP for distributions over linear system matrices, e.g., obtained by Bayesian linear regression, is a straightforward extension of the LQR section 2.1.1 [De Koning, 1982].
In (2.5) now the expectations w.r.t. A_t, B_t, a_t have to be considered,

$$\begin{aligned} V_{t-1}(x_{t-1}) = \min_{q_t} \mathbb{E}_{\epsilon_t, A_t, B_t, a_t}\Big[& \|x^\star_{t-1} - x_{t-1}\|^2_{Q_{t-1}} + \|u^\star_t - q_{t-1}\|^2_{R_{t-1}} \\ & + (A_{t-1}x_{t-1} + B_{t-1}q_{t-1} + a_{t-1} + \epsilon_t)^T P_t \\ & \quad \cdot (A_{t-1}x_{t-1} + B_{t-1}q_{t-1} + a_{t-1} + \epsilon_t) \\ & + p_t^T (A_{t-1}x_{t-1} + B_{t-1}q_{t-1} + a_{t-1} + \epsilon_t) + v_t \Big], \end{aligned}$$

[1] From a theoretical perspective BS-MPC 4.2 is not exactly the problem, which we would like to solve, a single fixed random variable θ seems more sound. However, a fixed variable would make the problem even more complicated as the optimal controller $\pi^\star_t$ might not depend on the recent state only.

The changes appear in the expectations

$$\begin{aligned} &\mathbb{E}[A_{t-1}^T P_{t-1} A_{t-1}], \quad \mathbb{E}[B_{t-1}^T P_{t-1} B_{t-1}], \\ &\mathbb{E}[B_{t-1}^T P_{t-1} A_{t-1}], \quad \mathbb{E}[a_{t-1}^T P_{t-1} a_{t-1}], \\ &\mathbb{E}[A_{t-1}^T P_{t-1} a_{t-1}], \quad \mathbb{E}[B_{t-1}^T P_{t-1} a_{t-1}], \end{aligned} \tag{4.3}$$

which can be computed using the bias-variance decomposition for the expectation of quadratic forms:

$$\begin{aligned} \mathbb{E}[\mathfrak{I}_j^T A_{t-1}^T P_{t-1} A_{t-1} \mathfrak{I}_i] &= \mathfrak{I}_j^T \mathbb{E}[A_{t-1}]^T P_{t-1} \mathbb{E}[A_{t-1}] \mathfrak{I}_i + \operatorname{tr}\left(P_{t-1} \mathbb{V}[A_{t-1}\mathfrak{I}_j, A_{t-1}\mathfrak{I}_i]\right) \\ &= \mathfrak{I}_j^T \mathbb{E}[A_{t-1}]^T P_{t-1} \mathbb{E}[A_{t-1}] \mathfrak{I}_i + \mathfrak{I}_j^T M_{t-1} \mathfrak{I}_i, \qquad M_{t-1} \in \mathbb{S},\, M_{t-1} \succeq 0, \\ \\ \Rightarrow \quad \mathbb{E}[A_{t-1}^T P_{t-1} A_{t-1}] &= \mathbb{E}[A_{t-1}]^T P_{t-1} \mathbb{E}[A_{t-1}] + M_{t-1}. \end{aligned} \tag{4.4}$$

Here, $\operatorname{tr}\left(P_{t-1} \mathbb{V}[A_{t-1}\mathfrak{I}_j, A_{t-1}\mathfrak{I}_i]\right)$ can be represented by $\mathfrak{I}_j^T M_{t-1} \mathfrak{I}_i$ and $M_{t-1} \succeq 0$ because the term is a symmetric bilinear form and always non-negative for $j = i$. All other terms in (4.3) can be computed in similar fashion.
An interesting property of this approach is that it does not *depend* on the distribution of ϵ and therefore, only on the moments of the state distribution.

4.3 Moment-Matching and Linear Affine Controllers

A direct approach to solving (4.2) is to directly optimize a controller from a certain class $\pi_t \in \Pi$, propagating the necessary information of the state distribution.

The first idea would be to discretize the entire state and solve for the optimal density and control signal. Although this approach can also be done in continuous time using a *Fokker-Planck* framework [Annunziato and Borzi, 2013], it quickly becomes computationally too expensive with increasing dimensionality.

An important observation in problem (4.2) is that the objective

$$\mathbb{E}_{\boldsymbol{\epsilon},\boldsymbol{\theta}}\left[||s_{T+1}^t - x_{T+t}^\star||_{Q_{t+T}}^2 + \sum_{i=1}^{T} \left(\|x_{i-1+t}^\star - s_i^t\|_{Q_{t-1+i}}^2 + \|u_{i-1+t}^\star - q_i^t\|_{R_{t-1+i}}^2 \right) \right]$$

$$\begin{aligned} &= ||\mathbb{E}[s_{T+1}^t] - x_{T+t}^\star||_{Q_{t+T}}^2 + \operatorname{tr}(Q_{t+T} \mathbb{V}[s_{T+1}^t]) \\ &\quad + \sum_{i=1}^{T} \Big(\|x_{i-1+t}^\star - \mathbb{E}[s_i^t]\|_{Q_{t-1+i}}^2 + \operatorname{tr}(Q_{t-1+i} \mathbb{V}[s_i^t]) \\ &\qquad + \|u_{i-1+t}^\star - \mathbb{E}[q_i^t]\|_{R_{t-1+i}}^2 + \operatorname{tr}(R_{t-1+i} \mathbb{V}[q_i^t]) \Big) \end{aligned}$$

only depends on the *Central Moments* $\mathbb{E}, \mathbb{V}$. Therefore, knowing the moments would suffice to solve the optimization problem.
The main difficulty of this approach for Bayesian dynamical models is that the nested integrals

$$\mathbb{E}[s_{t+1}] = \int s_{t+1} p(s_{t+1}|s_t, q_t; \theta_t) p(s_t, q_t) p(\theta_t) \,\mathrm{d}(s_{t+1}, s_t, q_t, \theta_t) \tag{4.5}$$

as well as

$$\mathbb{V}[s_{t+1}] + \mathbb{E}[s_{t+1}]\mathbb{E}[s_{t+1}]^T = \int s_{t+1} s_{t+1}^T p(s_{t+1}|s_t, q_t; \theta_t) p(s_t, q_t) p(\theta_t) \,\mathrm{d}(s_{t+1}, s_t, q_t, \theta_t), \tag{4.6}$$

$$p(s_t, q_t) = p(q_t|s_t; \pi_t) p(s_t), \tag{4.7}$$

$$p(s_t) = \int p(s_t|s_{t-1}, q_{t-1}; \theta_{t-1}) p(s_{t-1}, q_{t-1}) p(\theta_{t-1}) \,\mathrm{d}(s_t, s_{t-1}, q_{t-1}, \theta_{t-1}) \tag{4.8}$$

have to be evaluated.

In the case that $s_t \sim \mathcal{N}(\mu_t^s, \Sigma_t^s)$ and $p(q_t|s_t; \theta_t) = K_t s_t + k_t$ both integrals (4.5) and (4.6) can be solved in closed form for the GP with the SEARD and Bayesian linear systems. However, equation (4.8) still remains intractable even for a Gaussian input distribution. The idea of MM is to use the tractable equations (4.5), (4.6) and approximate the true density with a Gaussian defined by those moments. In [Kollar and Friedman, 2009] (chapter 8, Theorem 8.6), it is shown that this approach is an M-inference, meaning that the Gaussian with the same moments q minimizes the *Kullback-Leiber Divergence* $\mathrm{KL}(p||q) = \int p(x) \log \frac{p(x)}{q(x)} \,\mathrm{d}\, x$, a common distance measure for probability densities, with respect to the true distribution p in the class of Gaussians. Besides the simplicity and the closure properties of Gaussians, the approximation can also be motivated by the variational problem

$$\begin{aligned} &\arg\min_{\tilde{q} \in \mathscr{C}} \; -\mathscr{H}(\tilde{q}) \\ &\text{s.t.} \quad \mathbb{E}[\tilde{q}] = \int s_{t+1} p(s_{t+1}|s_t, q_t; \theta_t) p(s_t, q_t) p(\theta_t) \,\mathrm{d}(s_{t+1}, s_t, q_t, \theta_t), \\ &\qquad \mathbb{V}[\tilde{q}] + \mathbb{E}[\tilde{q}]\mathbb{E}[\tilde{q}]^T = \int s_{t+1} s_{t+1}^T p(s_{t+1}|s_t, q_t; \theta_t) p(s_t, q_t) p(\theta_t) \,\mathrm{d}(s_{t+1}, s_t, q_t, \theta_t), \end{aligned} \tag{4.9}$$

for a smooth density $\tilde{q}$, where the negative *Entropy* $-\mathscr{H}$ is defined as

$$-\mathscr{H}(\tilde{q}) := \int \tilde{q}(s_t) \log(\tilde{q}(s_t)) \,\mathrm{d}\, s_t.$$

It is a straight-forward proof (see [Barber, 2012]) that the Gaussian with the same moments $q = \mathcal{N}(\mathbb{E}[s_t], \mathbb{V}[s_t])$ is the optimizer of (4.9).

This so-called *maximum-entropy principle* is common in machine learning [Bishop and Nasrabadi, 2006] and can be summed up to "always fit the available information with the broadest density". A motivation from principles of statistical physics can be found in [Jaynes, 2003]. In the context of predicting the state distribution, this leads to a cautious approximation, that captures the broad shape and the tails rather than the modes of the true density. For robust control this behavior is also desirable, as the solution should be good for a broad range of possible states.
Figure 4.1 gives a example of MM of a SEARD-GP, given some observation data. As the GP-function candidates are nonlinear, the posterior density can be non-Gaussian even for Gaussian input densities. The MM rather takes care of the distributions tails than of the two modes, leading to a conservative approximation.

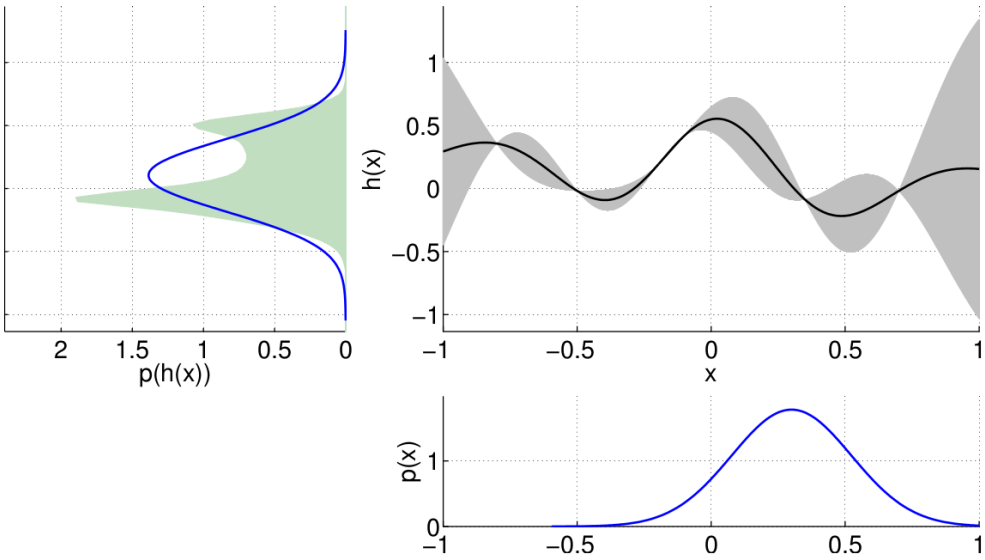

Figure 4.1: [Deisenroth and Rasmussen, 2011] Given a Gaussian input density $p(x)$ and possible candidate functions $h(x) \sim \mathcal{GP}$, the output density $p(h(x))$ is approximated by a Gaussian with same central moments.

Although it is also possible to use a single non-linear controller and apply additional MM (as in [Deisenroth and Rasmussen, 2011], [Englert et al., 2013]), a time-varying linear-affine control is a better choice. First, because using only $\mathbb{E}$ and $\mathbb{V}$, a fully parametrized linear affine controller suffices to represent every input/output-pair. Second, the calculation of the output moments is exact for *every* distribution and also very simple. Finally, the LQR solution, for the linearization of the model fixed to the mean, offers a good initialization for an iterative approach to NMPC.

Alternatives to MM can be any approach for nonlinear filtering, e.g., extended or unscented Kalman-filters [Julier and Uhlmann, 1997] that also allow to compute the derivatives of the propagated quantities for the optimization.

4.3.1 Moment-Matching for Bayesian Linear Systems

The basic technique to compute the moments in Bayesian models is the use of the *Law of Total Expectation* or the *Law of Total Variance*:

$$\begin{aligned}\mathbb{E}[x] &= \mathbb{E}_y[\mathbb{E}_x[x|y]],\\ \mathbb{V}[x] &= \mathbb{V}_y[\mathbb{E}_x[x|y]] + \mathbb{E}_y[\mathbb{V}_x[x|y]].\end{aligned}$$

Assuming a Bayesian model of the form

$$y = \Theta x, \quad \mu_x := \mathbb{E}[x], \Sigma_x := \mathbb{V}[x], \quad \mu_\Theta := \mathbb{E}[\mathrm{vec}(\Theta)], \Sigma_\Theta := \mathbb{V}[\mathrm{vec}(\Theta)]$$

the expectation of y is

$$\mathbb{E}[y] = \mathbb{E}_\Theta[\mathbb{E}_x[\Theta x|\Theta]] = \mathbb{E}_\Theta[\Theta\mu_x] = \mathrm{vec}^{-1}(\mu_\Theta)\mu_x,$$

whereas the covariance is

$$\begin{aligned}\mathbb{V}[y] &= \mathbb{V}_\Theta[\mathbb{E}_x[\Theta x|\Theta]] + \mathbb{E}_\Theta[\mathbb{V}_x[\Theta x|x]] = (\mu_x^T \otimes \mathfrak{I}^{n_x})\Sigma_\Theta(\mu_x \otimes \mathfrak{I}^{n_x}) + \mathbb{E}_w[\Theta\Sigma_x\Theta^T]\\ &= (\mu_x^T \otimes \mathfrak{I}^{n_x})\Sigma_\Theta(\mu_x \otimes \mathfrak{I}^{n_x}) + \mathrm{vec}^{-1}(\mu_\Theta)\Sigma_x\,\mathrm{vec}^{-1}(\mu_\Theta)^T + \underbrace{M(\Sigma_x)}_{\text{see (4.4)}}.\end{aligned}$$

The derivatives are also straight-forward to compute:

$$\begin{aligned}\frac{\partial\,\mathbb{E}[y]}{\partial\,\mu_x} &= \mathrm{vec}^{-1}(\mu_\Theta), \quad \frac{\partial\,\mathbb{E}[y]}{\partial\,\Sigma_x} = 0,\\ \frac{\partial\,\mathbb{V}[y]}{\partial\,\mu_x}(\mu) &= (\mu^T \otimes \mathfrak{I})\Sigma_\Theta(\partial\mu_x \otimes \mathfrak{I}) + (\partial\mu_x^T \otimes \mathfrak{I})\Sigma_\Theta(\mu \otimes \mathfrak{I}),\\ \frac{\partial\,\mathbb{V}[y]}{\partial\,\Sigma_x} &= \mathrm{vec}^{-1}(\mu_\Theta)\partial\Sigma_x\,\mathrm{vec}^{-1}(\mu_\Theta)^T + M(\partial\Sigma_x).\end{aligned}$$

It should be mentioned, that the distributions (compare 4.2) of both x and $\mathrm{vec}(\Theta)$ do *not* matter. As a consequence the propagation of the moments of the state variables $\mathbb{E}[x_t], \mathbb{V}[x_t]$ is exact.

4.3.2 Moment-Matching for the Squared Exponential Kernel Gaussian Processes

For Gaussian processes, in principle the same laws can be applied, but unlike the linear case, the assumption of Gaussian input distributions is crucial. In addition to that, the derivation of the moments is more complicated than in Bayesian linear systems and only given in analytic form for certain kernel functions, e.g., SEARD or polynomial kernels. First the propagation of uncertainty through GP models by MM was proposed [Quinonero-Candela et al., 2003] for multiple-step prediction in time-series forecasting. [Deisenroth and Rasmussen, 2011] later used MM prediction to

optimize a controller in model-based reinforcement learning.
The formulae for the SEARD-GP $y \sim \mathcal{GP}(x)$ will be presented without proof for sake of brevity.

Given an input-distribution $x \sim \mathcal{N}(\mu, \Sigma)$, $x \in \mathbb{R}^n$ and data $D = (Y, X), Y \in \mathbb{R}^{m \times d}, X \in \mathbb{R}^{n \times d}$ the moments of the output $y \in \mathbb{R}^m$ are:

$$\begin{aligned} \forall_{i=1}^{m} \quad \mathbb{E}[y]_i &= \boldsymbol{w}_i^T \boldsymbol{\theta}_i, \qquad \boldsymbol{\theta}_i = (\mathfrak{K}_i + \sigma_i^2 \mathfrak{I})^{-1} \boldsymbol{y}_i, \quad \boldsymbol{y}_i = (Y_{i,1}, \ldots, Y_{i,d})^T \\ \boldsymbol{w}_i &= (w_{i,1}, \ldots, w_{i,d})^T, \\ w_{i,k} &= \alpha_i^2 \det(\Sigma L_i^{-1} + \mathfrak{I})^{-\frac{1}{2}} \exp\big(-\frac{1}{2}(x_k - \mu)^T (\Sigma + \boldsymbol{L}_i)^{-1} (x_k - \mu)\big), \end{aligned} \tag{4.10}$$

$$\begin{aligned} \forall_{i,j=1}^{m} \quad \mathbb{V}[y]_{i,j} &= \begin{cases} \boldsymbol{\theta}_i^T \boldsymbol{W}_{i,j} \boldsymbol{\theta}_j - \mathbb{E}[y]_i \mathbb{E}[y]_j & \text{if} \quad i \neq j \\ \boldsymbol{\theta}_i^T \boldsymbol{W}_{i,j} \boldsymbol{\theta}_i - \mathbb{E}[y]_i^2 + \alpha_i^2 - \operatorname{tr}\left((\mathfrak{K}_i + \sigma_i^2 \mathfrak{I})^{-1} \boldsymbol{W}_{i,j}\right) & \text{else} \end{cases}, \\ \forall_{k,l}^{d} \, W_{i,j}^{k,l} &= \frac{\exp(\eta_{i,j}^{k,l})}{\sqrt{\det(\boldsymbol{R}_{i,j})}}, \\ \eta_{i,j}^{k,l} &= 2(\log(\alpha_i) + \log(\alpha_j)) - \frac{\upsilon_k^T \boldsymbol{L}_i^{-1} \upsilon_k + \upsilon_l^T \boldsymbol{L}_j^{-1} \upsilon_l - [\boldsymbol{z}_{i,j}^{k,l}]^T \boldsymbol{R}_{i,j} \Sigma \boldsymbol{z}_{i,j}^{k,l}}{2}, \\ \boldsymbol{z}_{i,j}^{k,l} &= \boldsymbol{L}_i^{-1} \upsilon_k + \boldsymbol{L}_j^{-1} \upsilon_l, \qquad \upsilon_k = x_k - \mu, \qquad \boldsymbol{R}_{i,j} = \Sigma(\boldsymbol{L}_i + \boldsymbol{L}_j)^{-1} + \mathfrak{I}. \end{aligned} \tag{4.11}$$

Therefore, together with the derivatives of the MM, which can be found in [Deisenroth and Rasmussen, 2011], approximate Bayesian optimal control can also be done for GPs.

4.4 Approximated Nonlinear Model Predictive Control with Gaussian Processes

The results of the previous chapters can now be combined to a robust NMPC algorithm to control a system, which is only known up to sample data.
So far bounds on the control input have not been considered. However, in practice in almost every control scenario torque limits are given and have to be respected in NMPC in both planning and application of the control. A common way to include bounds in stochastic optimization problems are so-called *chance-constraints*. Here the deterministic bound

$$u_{\min} \leq q \leq u_{\max}$$

is relaxed to be only fulfilled with a certain probability ϑ.
Combined with the linear affine controller, this leads to the nonlinear constraints

$$\begin{aligned} -u_{\max} + K\mu_s + k + \upsilon(\vartheta)\sqrt{\operatorname{diag}(K\Sigma_s K^T)} &\leq 0, \\ u_{\min} - \big(K\mu_s + k - \upsilon(\vartheta)\sqrt{\operatorname{diag}(K\Sigma_s K^T)}\big) &\leq 0, \end{aligned} \tag{4.12}$$

where $\upsilon(\vartheta)$ is the ϑ-quantile of the Gaussian. The constraints (4.12) can be seen as a safety margin from the original bounds depending on the uncertainty in the state.
From an optimization perspective, however, these constraints are problematic as they are non-convex.

The NMPC algorithm for Gaussian processes can be summed up in the optimization problems

$$\text{GP-NMPC}(x_t) = \begin{cases} \min_{\mu,\Sigma,K,k} & \|\mu_{T+1} - x^\star_{T+t}\|^2_{Q_{t+T}} + \operatorname{tr}(Q_{t+T}\Sigma_{T+1}) \\ & + \sum_{i=1}^{T} \Big(\|x^\star_{i-1+t} - \mu_i\|^2_{Q_{t-1+i}} + \operatorname{tr}(Q_{t-1+i}\Sigma_i) \\ & \qquad + \|u^\star_{i-1+t} - K_i\mu_i - k_i\|^2_{R_{t-1+i}} + \operatorname{tr}(R_{t-1+i}K_i\Sigma_i K_i^T) \Big) \\ \text{s.t.} & \mu_1 - x_t = 0, \\ & \Sigma_1 = \mathbf{0}, \\ & \\ & i \in \{1,\dots,T\}, \\ & GP_\mu(\mu_t, \Sigma_t; K_t, k_t) = \mu_t, \\ & GP_\Sigma(\mu_t, \Sigma_t; K_t, k_t) = \Sigma_t, \\ & -u_{\max} + K_t\mu_t + k_t + \upsilon(\vartheta)\sqrt{\operatorname{diag}(K_t\Sigma_t K_t^T)} \le \text{tol}, \\ & u_{\min} - \left(K_t\mu_t + k_t - \upsilon(\vartheta)\sqrt{\operatorname{diag}(K_t\Sigma_t K_t^T)}\right) \le \text{tol}, \end{cases} \tag{4.13}$$

where we use MM GP_μ, GP_Σ to propagate the uncertainty. The initial state is *deterministic* $\mu_1 - x_t = 0, \quad \Sigma_1 = \mathbf{0}$, as we assume exact measurement of the system state. The additional tolerance tol is introduced to prevent infeasibility problems and set to the order of the system noise.

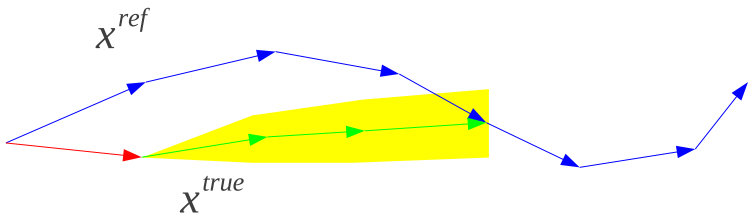

Figure 4.2: GP-NMPC on a tracking problem. To minimize the *expected* deviation, with respect to model errors and disturbances, from the reference [blue], a state-tube [yellow] from a disturbed state [red] is planned.

Figure 4.2 shows how the original planning in NMPC changes to robust planning by means of a state-tube, which includes the possible uncertainties.
As a linear-affine controller is directly optimized, one can think of the method as a explicit approach. However, the first controller K_1, k_1 can be reduced to a control input only, because the the first state is deterministic. Using Diehl's techniques from section 2.2 and replanning in every time-step, the presented method can be described as *Semi-Implicit*: The direct parametrization of the controller in the planning controls the state-tube and is sub-optimal w.r.t. the control-bounds (see [Bemporad et al., 2002]), but the correction at every time step keeps the control-signal feasible.

5 Evaluation

To test the proposed algorithms and investigate whether the theoretical appealing properties lead to good performance, we implemented all in MATLAB. The QPs appearing in NMPC were solved with the presented homotopy method using the MATLAB interface of qpOASES[1]. All code necessary for simulation, the NMPC solution and training of the GP is available upon request.
As a benchmark problem we decided on the *double-pendulum* and the *cart-pole* in the implementation of [Deisenroth, 2008]. In both scenarios, the dynamics of the system are highly nonlinear, have control constraints and are simulated with a external noise-influence.

5.1 Implemented Methods

We implemented and compared different methods using the weights $Q_t = 2\,\mathfrak{I}, \quad R_t = \mathfrak{I}$:

1. The ***(Full) Gaussian Process Nonlinear Model Predictive Control (GP-NMPC)*** method (4.13) introduced in chapter 4 with a BFGS-update rule (section 2.2.2) within Diehl's framework (section 2.2.1) on a time horizon of 10 steps on the double-pendulum and a horizon of 25 steps on the cart-pole,

2. Deterministic NMPC with the GP-mean function
   a) with the standard quadratic objective

$$\|x_t^\star - x_t\|_{Q_t}^2 + \|u_t^\star - u_t\|_{R_t}^2$$

   (***Deterministic Gaussian Process Nonlinear Model Predictive Control (D-GP-NMPC)***),

   b) with the quadratic objective *augmented* with the the predictive variance $\Sigma_t^{\mathcal{GP}}$ at input (x_{t-1}, u_{t-1})

$$\|x_t^\star - x_t\|_{Q_t}^2 + \mathrm{tr}(Q_t \Sigma_t^{\mathcal{GP}}) + \|u_t^\star - u_t\|_{R_t}^2,$$

   (***Augmented Gaussian Process Nonlinear Model Predictive Control (AUG-GP-NMPC)***),

   within Diehl's framework using the Gauss-Newton approximation (see section 2.2.2) on a time horizon of 10/25 steps,

3. Linear MPC (section 2.1.2) using the linearization of the GP along the reference (***Gaussian Process (Linear) Model Predictive Control (GP-MPC)***) on a horizon of 10/25 steps.

4. Linear MPC (section 2.1.2) using the exact linearization of the dynamics (**exact MPC**) on a horizon of 10/25 steps.

[1] latest version 3.0beta downloaded from `www.qpoases.org`

5. Full horizon LQR using the exact linearization of the dynamics, control signals truncated to torque-limits in execution (**exact LQR**),

6. Full horizon LQR using the linearization of the trained GP, control signals truncated to torque-limits in execution (**GP-LQR**),

All deterministic iterative methods were initialized with the reference, while the full GP-approach was initialized with a *feasible* controller, which was computed offline. The first five methods hereby serve to evaluated the GP model and the use of the predictive uncertainty in NMPC, while the last two ones are benchmark heuristics, which show, what performance is possible if the full horizon is considered.

5.2 Double-Pendulum Scenario

The double-pendulum is the concatenation of two ordinary pendulums and is used in nonlinear control theory because of its simplicity and yet rich variety of dynamics including even chaotic behavior.

5.2.1 Description of Scenario

Using Lagrangian mechanics (compare [Deisenroth, 2008], Appendix C *pendubot*, which differs only in actuation) the equations of motion of the joint-angles (see figure 5.1) ω_1, ω_2, when torques u_1, u_2 are applied in the joints, of the double-pendulum result in the *Differential-Algebraic Equation*

$$\begin{pmatrix} \ddot{\omega}_1 \\ \ddot{\omega}_2 \end{pmatrix} = \begin{pmatrix} l_2^2(\frac{1}{4}m_1 + m_2) + I_1 & \frac{1}{2}m_2 l_1 l_2 \cos(\omega_1 - \omega_2) \\ \frac{1}{2}m_2 l_1 l_2 \cos(\omega_1 - \omega_2) & \frac{1}{4}m_2 l_2^2 + I_2 \end{pmatrix}^{-1} v,$$

$$v = \begin{pmatrix} g l_1 \sin(\omega_1)(\frac{1}{2}m_1 + m_2) - \frac{1}{2}m_2 l_1 l_2 \dot{\omega}_2^2 \sin(\omega_1 - \omega_2) + u_1 \\ \frac{1}{2}m_2 l_2 (l_1 \dot{\omega}_1^2 \sin(\omega_1 - \omega_2) + g \sin(\omega_2)) + u_2 \end{pmatrix},$$

Table 5.1: Double-Pendulum: Parameters

m_1	0.5 kg	mass of first link
m_2	0.5 kg	mass of second link
l_1	0.5 m	length of first pendulum
l_2	0.5 m	length of second pendulum
g	0.5 $\frac{\mathrm{m}}{\mathrm{s}^2}$	acceleration of gravity
I_1	$m_1 l_1^2$	moment of inertia around midpoint of first pendulum
I_2	$m_2 l_2^2$	moment of inertia around midpoint of second pendulum
u_i	$\lvert u_i \rvert \leq 2$ N m	torque limits for $i \in \{1, 2\}$

where the links of the pendulum are modeled as straight rods and the joints are assumed to be

friction-free.
The given reference trajectory is an indirect swing-up (see figure 5.1) into the unstable full upright pose, which has to be held for a certain amount of time. With the given parameters (see tabular 5.1), especially the torque limits, it is impossible to perform a direct swing-up. Hence, potential energy has to be built up and then turned into kinetic energy first to perform the swing-up.

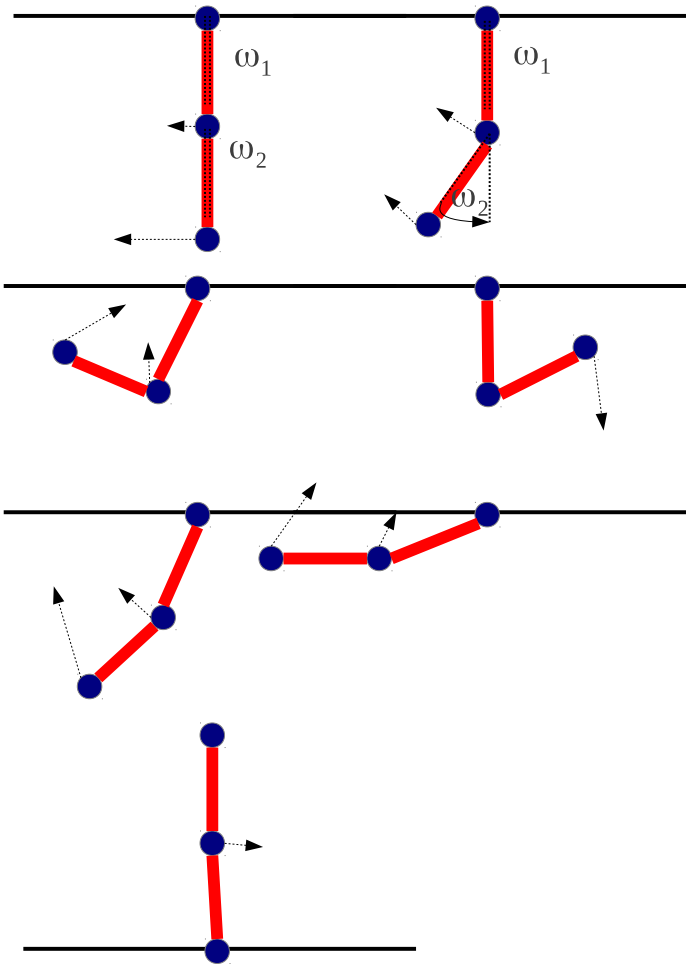

Figure 5.1: The indirect swing-up into the full upright pose

If the time-dependent system linearization should be inferred by Bayesian regression using a prior, problems arise because of the differing stability properties of the system at start and end point. As the double-pendulum starts in a stable rest-position and reaches an unstable terminal state, the first and last Jacobians differ. Hence, a assumption of global similarity is not valid, whereas a prior on neighboring system matrices as in Kalman-regression is reasonable.

The sample data (see figure 5.2) was generated using PILCO[2] [Deisenroth and Rasmussen, 2011] to learn a nonlinear controller and the noise free roll-out of the final controller, starting in the rest position, defines the reference.
The inputs are both angular velocities $\dot{\omega}_1, \dot{\omega}_2$, both angles ω_1, ω_2 and the two applied torques u_1, u_2, whereas the output consists of the differences in velocities and angles $\Delta(\dot{\omega}_1, \dot{\omega}_2, \omega_1, \omega_2)$.

[2] Implementation downloadable at `http://mlg.eng.cam.ac.uk/pilco/pilcoV0.9.zip`

Therefore, the final model is the sum of the identity map of the states and the GP-model on both states and controls. To deal with the periodic nature of the angle inputs, the angle data was centered to the point with the least distance in periodic space to all samples, the so-called *Fréchet-Mean*. Additionally, the data was periodically continued at the boundaries (see figure 5.2). However, the learned GP-model is not perfect yet, which causes the reference to *not* exactly match the GP-mean prediction dynamics.
The simulation of the double-pendulum was done at a sampling time of 0.02 s using MATLAB's ode45 solver and Gaussian noise with variance $\Sigma = \mathrm{diag}(0.001, 0.001, 0.01, 0.01)$.

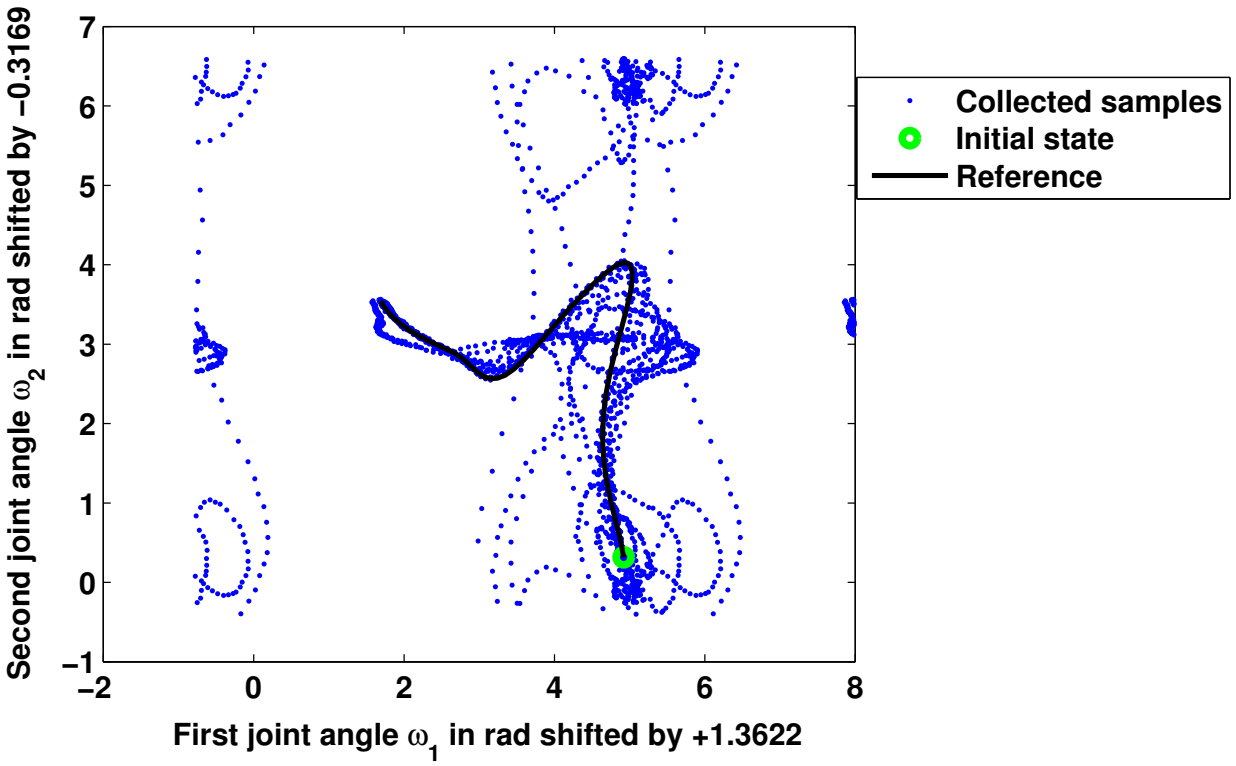

Figure 5.2: Collected trajectory data and reference. To manage the periodic nature of angles, the data was centered to the Fréchet-Mean of the periodic manifold and samples next to boundary where copied

5.2.2 Results on the Double-Pendulum

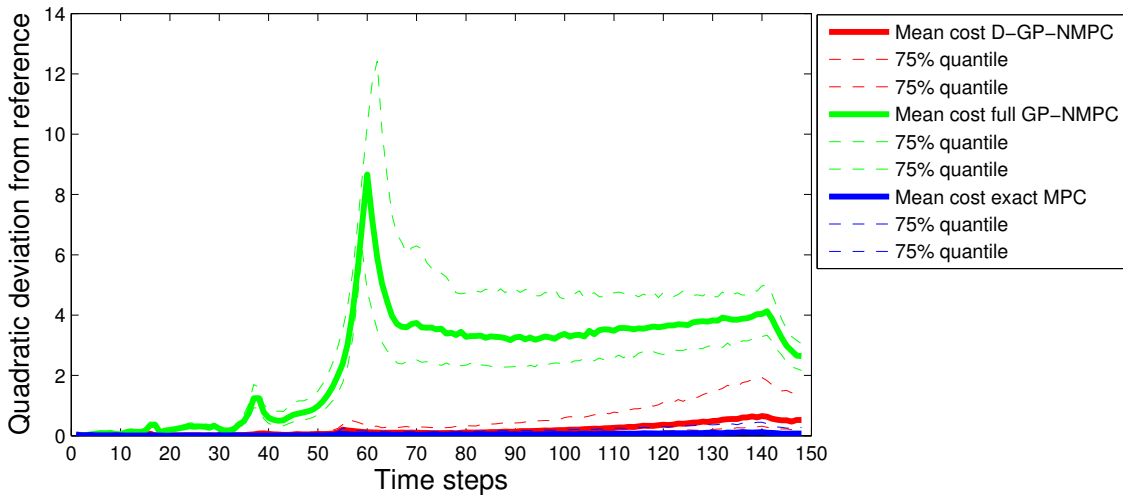

Figure 5.3: Comparison of full GP-NMPC, D-GP-NMPC and exact MPC on 200 roll-outs

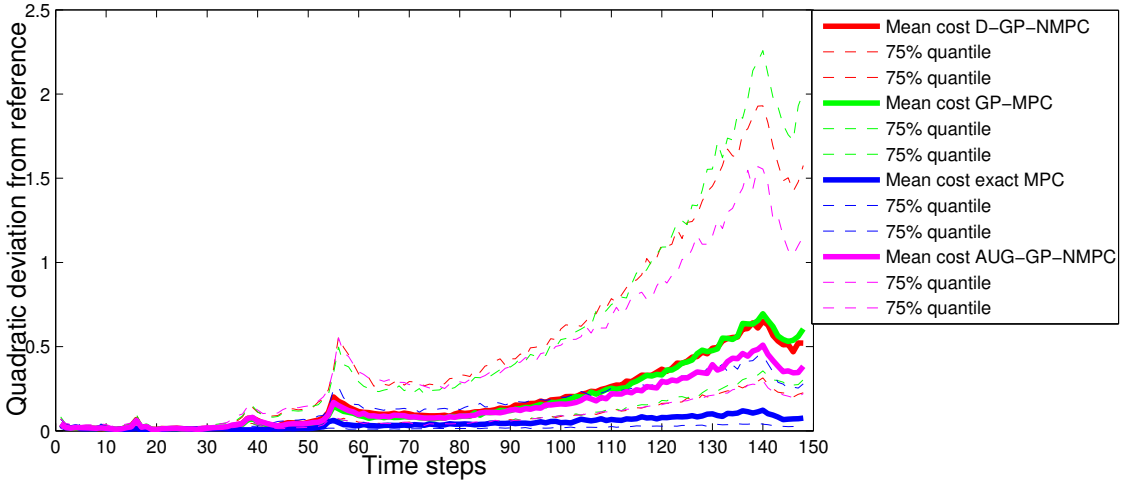

Figure 5.4: Comparison of full GP-MPC, D-GP-NMPC, AUG-GP-NMPC and exact MPC on 200 roll-outs

The first figure 5.3 shows the direct comparison of the NMPC approach with full propagation of the uncertainty to deterministic NMPC using the mean function and the 'optimal'[3] constrained

[3] The NMPC version was not implemented as the necessary sensitivity computation of the differential algebraic equation would have been out of scope of this thesis

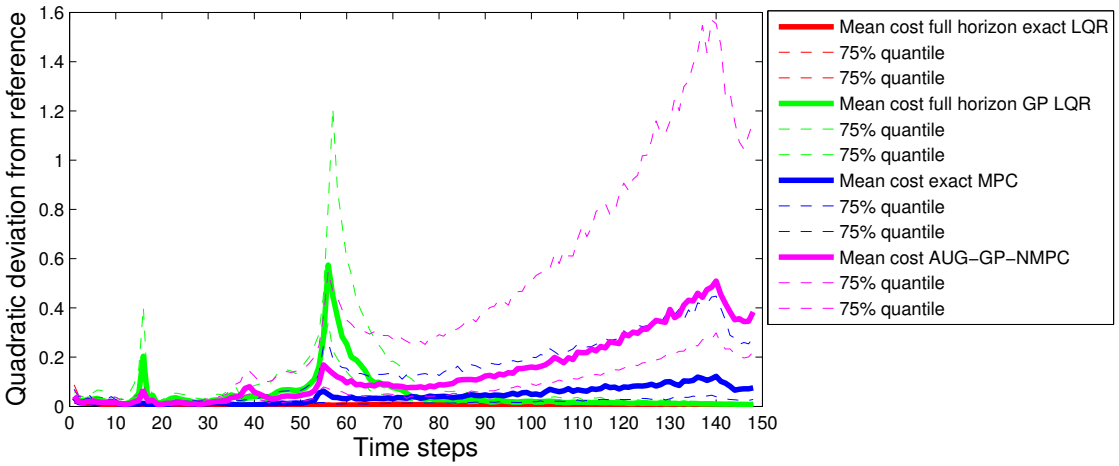

Figure 5.5: Comparison of AUG-GP-NMPC, exact MPC, GP LQR and exact LQR on 200 roll-outs

MPC on the exact system linearization on the time horizon of 10 steps. Unfortunately all the effort put into MM and the chance constraints, did not pay off, as the full NMPC approach was the worst. Aspects form both theoretical and numerical side that are likely to have caused this bad performance are discussed in chapter 6.
The second figure 5.4 shows two aspects: First, despite the failure of the full GP-NMPC approach the predictive variance is still useful, as it leads to significantly better performance compared to NMPC without its use. Second, the slight improvement of GP-NMPC to GP-MPC shows that the main contribution to the deviation is not caused by the system noise but from the imperfect model. Therefore, the linearization of the GP around the reference has almost the same quality than the iterative linearization in NMPC.
Finally, the last figure 5.5 shows that AUG-GP-NMPC is competitive to MPC controllers that use the full time horizon, the true linearization or even both. Hereby, the good performance of the truncated LQR approaches indicates that especially in this scenario the horizon length has a strong influence on the performance of MPC-methods.

Table 5.2 gives a summation of the results for all tested methods.

Table 5.2: Double-Pendulum: Total Squared Deviations

7.5382×10^2	GP-NMPC
7.1918×10^1	D-GP-NMPC
5.5732×10^1	AUG-GP-NMPC
8.1623×10^1	GP-MPC
2.0537×10^1	exact MPC
1.8854×10^1	GP-LQR
1.5161×10^0	exact LQR

5.3 Cart-Pole Scenario

In addition to the double-pendulum we also evaluated the methods on the cart-pole for comparison and validation of our hypotheses. Therefore, we will only briefly introduce the scenario and present the results. Similar to the double-pendulum, the cart-pole is a nonlinear system with stable and unstable rest-positions. It consists of a pendulum mounted on a cart, which can only move horizontal.

5.3.1 Description of Scenario

The derivation dynamic equation can again be found in [Deisenroth, 2008]. If an external force u is applied to the cart-pole, the horiontal position x and the joint-angle of the cart-mounted pendulum ω (defined in figure 5.6) evolve according to the *Ordinary Differential Equation*

$$\begin{pmatrix} \ddot{x} \\ \ddot{\omega} \end{pmatrix} = \begin{pmatrix} \dfrac{2ml\dot{\omega}^2 \sin(\omega) + 3mg\sin(\omega)\cos(\omega) + 4u - 4b\dot{x}}{4(M+m) - 3m\cos(\omega)^2}, \\ \dfrac{-3ml\dot{\omega}^2 \sin(\omega)\cos(\omega) - 6(M+m)g\sin(\omega) - 6(u - b\dot{x})\cos(\omega)}{4l(m+M) - 3ml\cos(\omega)^2} \end{pmatrix}$$

Table 5.3: Cart-Pole: Parameters

m	0.5 kg	mass of pendulum
M	0.5 kg	mass of cart
l	0.5 m	length of pendulum
g	0.5 $\frac{\mathrm{m}}{\mathrm{s}^2}$	acceleration of gravity
b	0.1 $\frac{\mathrm{N\,s}}{m}$	coefficient of friction between cart and ground
u	$\lvert u\rvert \leq 10\,\mathrm{N}$	force limits

The reference is a indirect up-swing to the unstable up-right position (see figure 5.6) and a direct up-swing is prevented by the force-limit in combination with the parameters of the system.

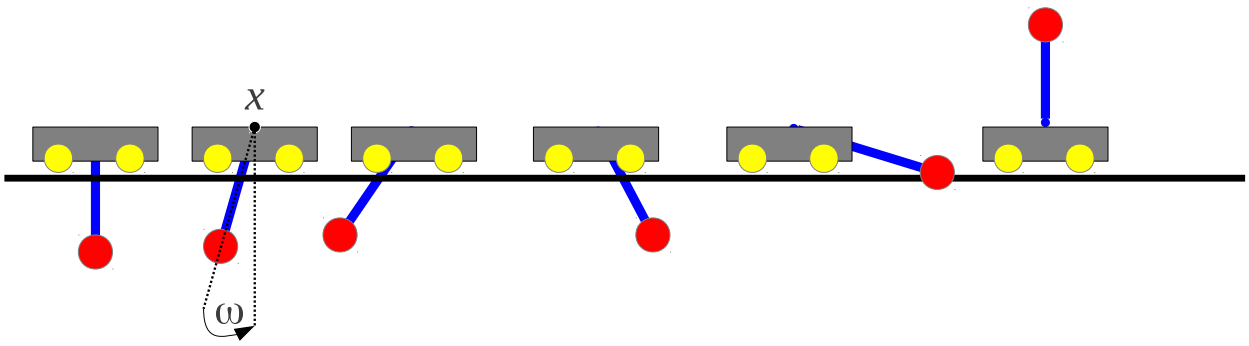

Figure 5.6: The indirect swing-up into the full upright pose

The sampling interval is 0.02 s, the noise covariance $\Sigma = \mathrm{diag}(0.001, 0.001, 0.01, 0.01)$ and the GP was learned on the state-differences of roll-outs of the PILCO-method as in the double-pendulum scenario.

5.3.2 Results on the Cart-Pole

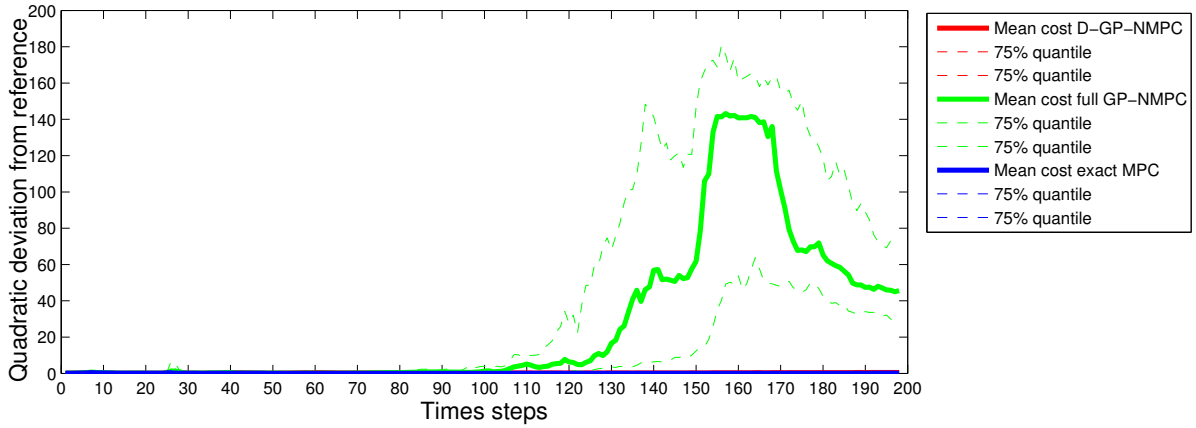

Figure 5.7: Comparison of full GP-NMPC, D-GP-NMPC and exact MPC on 200 roll-outs

Figure 5.7, is similiar to the results on the double-pendulum. Here the full approach is not even able to stabilize at all, as from time step 100 on the system behaves chaotic. This might be explained with the fact, that we spend more time on learning the hyper-parameters of the GP and especially designing a suitable penalty for extreme lenght-scales on the double-pendulum

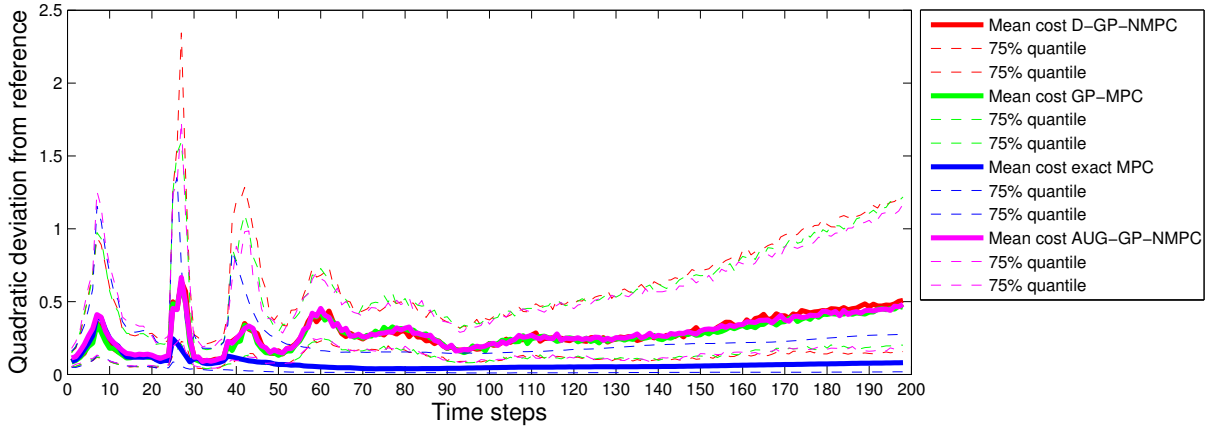

Figure 5.8: Comparison of full GP-MPC, D-GP-NMPC, AUG-GP-NMPC and exact MPC on 200 roll-outs

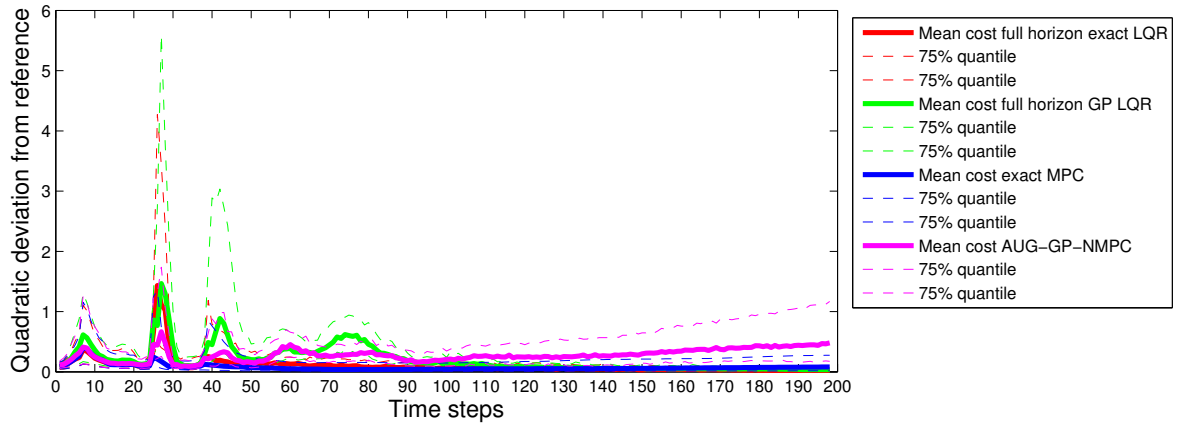

Figure 5.9: Comparison of AUG-GP-NMPC, exact MPC, GP LQR and exact LQR on 200 roll-outs

In figure 5.8 and 5.9, the same results as in the case of the double-pendulum can be observed: The simplified approaches built on the mean-function give good performance compared to approaches with the exact model.
The average total deviation of all tested methods can be found in table 5.4.

Table 5.4: Cart-Pole: Total Squared Deviations

7.0514×10^3	GP-NMPC
1.0744×10^2	D-GP-NMPC
1.0462×10^2	AUG-GP-NMPC
1.2508×10^2	GP-MPC
5.3040×10^1	exact MPC
7.3590×10^1	GP-LQR
4.5827×10^1	exact LQR

6 Discussion

The experimental results show worse performance of the NMPC approach building on the propagation of the uncertainty according to noise influence and parametric uncertainty. As this is opposing to the success of similar methods in reinforcement learning [Deisenroth and Rasmussen, 2011] and imitation learning [Englert et al., 2013], in the following possible reasons for the degeneracy are investigated and discussed. Hereby, the first section focuses on problems that arise from the used optimization model and algorithm, while the second discusses numeric effects. The hypotheses are backed up by numerical values obtained from the double-pendulum.

6.1 Theoretical Issues

A first possible reason for the bad performance can be found in the combination of the propagation of uncertainty with Diehl's suboptimal real-time scheme (see section 2.2.1). The prediction error $\epsilon = x_0 - x_1$ between the estimated and true state played a major role in all bounds on suboptimality. On the first glance, this seems to be quite unimportant in the case of GP-models, as even the uncertainty of the prediction can be quantified, but it is actually problematic.
In deterministic NMPC the recent state x_t corresponds to a series of planned optimal states and controls $(\boldsymbol{s},\boldsymbol{q})_t$, where the prediction $\tilde{x}_{t+1}$ is the next planned state s_1. If the difference $\tilde{x}_{t+1} - x_{t+1}$ of prediction and true next state x_{t+1} is small, then also the optimal next planned states and controls $(\boldsymbol{s},\boldsymbol{q})_{t+1}$ (except the last control and state) differ little from the previous. In GP-NMPC this might not be the case, as the noise and the model uncertainty have additional influence.

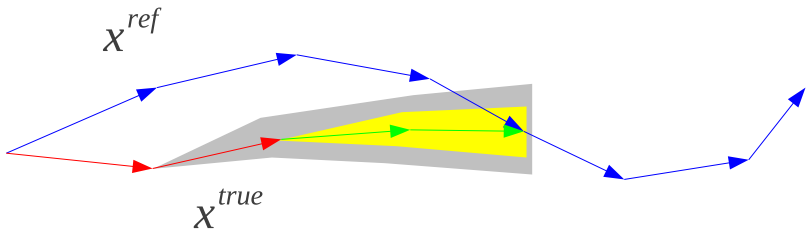

Figure 6.1: Two consecutive state-tubes, first [gray] and second [yellow], in GP-NMPC in case of perfect prediction.

As in GP-NMPC the planned states $(\boldsymbol{s})_t$ also contain the planned covariances $\boldsymbol{\Sigma}_s$, there is always

a difference to the successive step because the uncertainty decreases. Figure 6.1 illustrates these differences in the covariances by the difference of the state tubes. Although this behaviour is desirable , it causes problems in the optimization algorithm underlying Diehl's scheme. In fact it is questionable, if the conditions for the bounds on sub-optimality (see section 2.2.1) can be full-filled. As the effect is inherent to any model with propagation of uncertainty in combination with Diehl's scheme, the possible options are:

- Increasing the number of iteration steps of the optimizer per time step, which is likely to produce a significant delay in measurement of the state and execution of the control.
- No use of the propagation of the uncertainty. Here, the AUG-GP-NMPC (see figure 6.2)

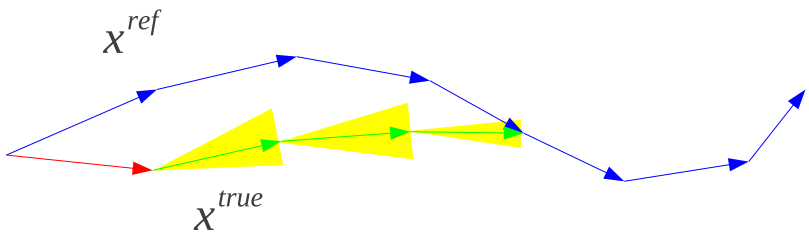

Figure 6.2: The predictive uncertainty yellow cones is part of the optimization objective but not incorporated in the dynamics

approach might serve as an example, as the model uncertainty is included in the objective but not in the dynamics. The good experimental results compared to GP-NMPC and NMPC using the mean-function indicate that this approach is a good trade-off between taking the model quality into account and efficient optimization.

Second, the non-convex (underlined) objective

$$\|x^d - \mu_s\|_Q^2 + \mathrm{tr}(Q\Sigma_s) + \|u^d - \underline{K\mu_s} - k\|_R^2 + \mathrm{tr}(\underline{RK\Sigma_s K^T}) \tag{6.1}$$

and constraints

$$u_{\max} - \underline{K\mu_s} - k + \underline{\vartheta\sqrt{\mathrm{diag}(K\Sigma_s K^T)}} \leq \mathrm{tol} \tag{6.2}$$

cause additional difficulties for the optimizer. Although every NMPC instance is potentially non-convex because of the nonlinear dynamics, offline optimization of full GP-NMPC and NMPC on the mean function show drastic differences in necessary iterations to convergence. Especially the complicated mixed state-control constraints (6.2) lead to SQP-iterations with infeasible QPs (see

QP (2.14)) and are generally hard to fulfil. Additionally, it turned out that additional regularization in form of step-size limitation is needed to stabilize the algorithm in online application.
As a possible solution-approach we suggest fixing the controller gain to an offline optimized matrix and only optimizing the offset $\boldsymbol{k}$ in the online algorithm. This results in a convex objective (6.1) and also parts of the constraints become convex. To simplify the constraints further we tried to use linear regression to approximate the square-root on covariance-matrices collected in offline-optimization. However, the approach turned out to cause even more problems with infeasibility, therefore, the results are not presented. In further work other convex approximation techniques as presented in [Nemirovski and Shapiro, 2006] should be considered.

6.2 Numerical Issues

Besides the theoretical problems, there were also issues with the implementation. Necessary for numerical optimization is the exact computation of the gradients and Jacobians. To compute the derivatives of the MM we used the routines included in the implementation of PILCO[1] [Deisenroth and Rasmussen, 2011]. Although the thorough implementation takes use of several matrix-identities for maximal numerical stability, still visible numerical errors in the evaluations occur. The GP learned on the given data at first produced average Jacobian $\frac{\partial \Sigma_{\text{out}}}{\partial \Sigma_{\text{in}}}$ errors of 10^{-4} compared to central differences with the standard disturbance of 10^{-6}, even at the optimal solution. The reason for this behavior can be explained by the learned hyper-parameters $\boldsymbol{L}, \boldsymbol{\sigma}, \boldsymbol{\alpha}$ (see equation (3.7)). The table 6.1 shows the final hyper-parameter learned with a penalty on large length-

Table 6.1: Double-Pendulum: Learned Length-Scales

$10^5\times$	0.0022	0.0011	0.0107	0.0548	l_1^2
	0.0012	0.0021	1.0727	0.0053	l_2^2
	0.0000	0.0000	0.0003	0.0000	l_3^2
	0.0000	0.0000	0.0006	0.0003	l_4^2
	0.0017	0.0028	0.0448	0.0051	l_5^2
	0.0009	0.0007	0.0027	0.0088	l_6^2
	0.0000	0.0001	0.0000	0.0000	σ^2
	0.0000	0.0000	0.0000	0.0000	α^2

scales yielding Jacobian errors of 10^{-6}. On the trajectory data these values yield kernel matrices $\boldsymbol{\mathfrak{K}}$ with *Condition Number* κ

$$\begin{aligned}\kappa(\mathfrak{K}_1) &= 9.9887 \times 10^{18}\\ \kappa(\mathfrak{K}_2) &= 4.4273 \times 10^{18}\\ \kappa(\mathfrak{K}_3) &= 1.0532 \times 10^{21}\\ \kappa(\mathfrak{K}_4) &= 1.6954 \times 10^{21},\end{aligned} \tag{6.3}$$

[1] Implementation downloadable at `http://mlg.eng.cam.ac.uk/pilco/pilcoV0.9.zip`

$\kappa(A) := \frac{\lambda_{\max}}{\lambda_{\min}}$, where $\lambda_{\min}, \lambda_{\max}$ are the largest and smallest eigenvalue of a matrix A.

The condition number has a very important role in numerical mathematics, as it indicates how pone the solution of an equation $A\,b = c$ is to numerical errors. For a disturbance of the right side $c + \Delta c$, it holds for an invertible A and the solution $b + \Delta b$

$$A(b + \Delta b) = c + \Delta c \quad \Rightarrow \frac{\|\Delta b\|_2}{\|b\|_2} \leq \kappa(A) \frac{\|\Delta c\|_2}{\|c\|_2}$$

(see any textbook on numerical mathematics, e.g. [Stoer et al., 1993]). Therefore κ bounds the propagation of the relative errors in the solution of the linear equation system.

In Gaussian processes the condition-number of the kernel matrix strongly affects the possible precision in which the prediction equations (3.6) and the MM (4.10) (4.11) can be evaluated. Especially the equations for the variance (4.11) are strongly affected by the hyper-parameters, as also the matrices $\boldsymbol{W}_{i,j}$ are ill-condition because of their similarity to the kernel matrices $\mathfrak{K}$.
The connection of the large length-scales to the bad condition of $\mathfrak{K}$ arises from the give data set (see 5.2). Here the data does not spread uniformly over the entire space, but instead agglomerates along the reference. In combination with the large length-scales, the rows corresponding to these samples in the kernel matrix are very similar because of the kernel-function

$$\mathfrak{k}(x_1, x_2) = \exp\big(-\frac{1}{2}(x_1 - x_2)^T L^{-1}(x_1 - x_2)\big),$$

what then leads to the bad conditioning numbers.
However, a strong penalty on the length-scales, that yields reasonable derivative errors, produces a bad GP-model and increases the mismatch of model and reference further. The presented length-scales in table 6.1 are already a result of a compromise between both adversarial objectives.
As the problem is known in the Bayesian machine-learning community, and more general in *Radial Basis Function* interpolation with the squared exponential function, already several approaches have been proposed to compress the data, while reducing the condition number. [Snelson and Ghahramani, 2006] proposed the FITC algorithm that optimizes a small number of *Pseudo-Samples* to approximate the full GP best. Although this approach is also used in PILCO as it additionally reduces the computational effort in the MM, it can not solve the problem of ill-condition kernel-matrices, the average condition-number is still of order 10^{14} and the average Jacobian error at 10^{-6}.
The influence of the bad numerical quality of the Jacobians on the entire GP-NMPC algorithm can also be seen in the comparison to moment-propagation with a linear system.

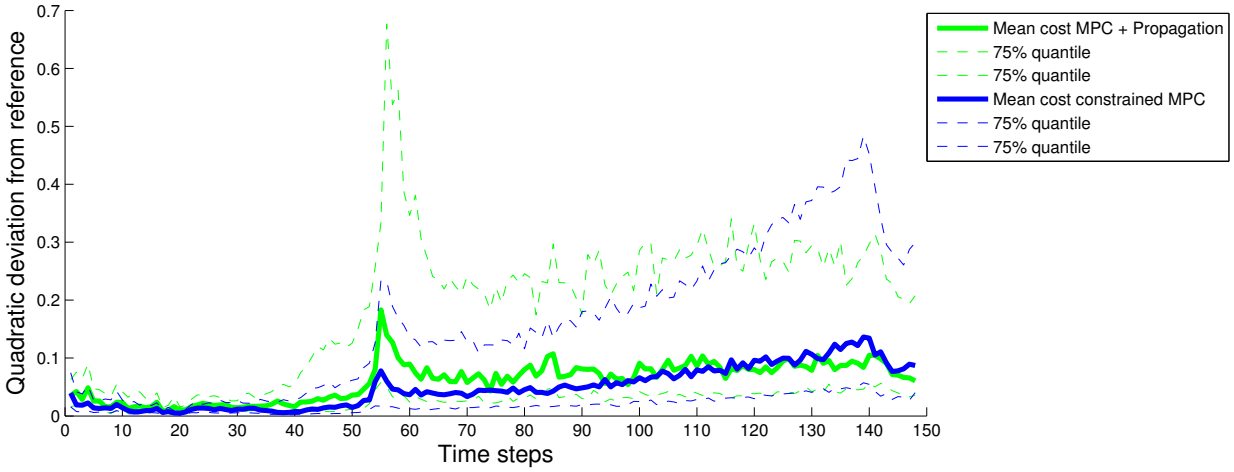

Figure 6.3: Comparison moment-propagation MPC and constrained MPC with the exact linearization on 200 roll-outs

Figure 6.3 shows, that the quality of the probabilistic approach can increase drastically if the MM is less complex and if the derivatives are more reliable. However, it should be mentioned that the initialization could also have been better, as finding a feasible controller for the moment-propagation in the linear model, starting from the same guess as in GP-NMPC, turned out to be easier.

As also the Jacobian of the deterministic GP-NMPC (see equation (3.6)) is effected by the extrem-length scales with a relative error of 10^{-8}, we suggest using a approximation using sparse local GP-models as proposed in [Snelson and Ghahramani, 2007] Especially for the trajectory data set, that has a varying sample-density, the approach seems to be advantageous and might result in less extreme length-scales while still giving a good prediction model.

7 Conclusion and Future Work

In this thesis we investigated the combination of model predictive control approaches with Bayesian system models. We introduced two recent optimization-techniques for online optimization and applied them to a moment-matching method for robust optimal control with Gaussian process models, leading to robust Gaussian process nonlinear model predictive control. Several different variants of the algorithm, as well as established approaches were tested on a simulated double-pendulum and cart-pole. Although the original proposed algorithm did not achieve good performance, simplified methods were competitive with control approaches using the exact dynamics. Furthermore, the degeneracy of the presented approach was discussed and potentially problematic theoretical and numerical aspects could be identified. Finally, we also presented suggestions to solve the occurring problems.

As the numerical issues with the algorithm were time demanding, we were not able to evaluate the method on hardware. Therefore, an important future issue is the implementation for hardware application and the evaluation on a real robot. If the real-time constraints can be fulfilled in the computation of the derivatives of the GP-model, we expect the presented AUG-GP-NMPC to also work in practise as Diehl's framework has already been successfully evaluated. Also an evaluation of the Kalman-approach to the inference of system linearizations and comparison to other Bayesian approaches, e.g. hierarchical models with a global prior, is a topic we want to investigate. Here it would also be of interest, if a suitable model of the series of linear models can out-perform the linearisation of a nonlinear global model, e.g. a GP as in the GP-LQR approach.
In addition to that, we could think of the combination of both online GP-learning and AUG-GP-NMPC as an extension of previous computed torque approaches [Nguyen-Tuong et al., 2008].
On the algorithmic side a thorough investigation of the effects of extreme length-scales and more important the development of GP-approaches that do not suffer from the numerical instabilities is an important issue.

List of Figures

List of Tables

Bibliography

[Annunziato and Borzi, 2013] Annunziato, M. and Borzi, A. (2013). A Fokker–Planck Control Framework for multidimensional Stochastic Processes. *Journal of Computational and Applied Mathematics*, 237(1):487 – 507.

[Archambeau et al., 2007] Archambeau, C., Cornford, D., Opper, M., and Shawe-Taylor, J. (2007). Gaussian Process Approximations of Stochastic Differential Equations. *Journal of machine learning research*, 1:1–16.

[Atkeson and Santamaria, 1997] Atkeson, C. G. and Santamaria, J. C. (1997). A Comparison of Direct and Model-based Reinforcement Learning. In *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, volume 4, pages 3557–3564. IEEE.

[Barber, 2012] Barber, D. (2012). *Bayesian Reasoning and Machine Learning*. Cambridge University Press.

[Bellman, 1954] Bellman, R. (1954). The Theory of Dynamic Programming. Technical report, DTIC Document.

[Bemporad et al., 2002] Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E. N. (2002). The Explicit Linear Quadratic Regulator for Constrained Systems. *Automatica*, 38(1):3–20.

[Bertsekas and Shreve, 1978] Bertsekas, D. P. and Shreve, S. E. (1978). *Stochastic Optimal Control: The Discrete time Case*, volume 139. Academic Press New York.

[Bertsekas and Tsitsiklis, 1996] Bertsekas, D. P. and Tsitsiklis, J. N. (1996). Neuro-Dynamic Programming (Optimization and Neural Computation Series, 3). *Athena Scientific*, 7:15–23.

[Bishop and Nasrabadi, 2006] Bishop, C. M. and Nasrabadi, N. M. (2006). *Pattern Recognition and Machine Learning*, volume 1. springer New York.

[Bock and Plitt, 1984] Bock, H. G. and Plitt, K.-J. (1984). A Multiple Shooting Algorithm for direct Solution of Optimal Control Problems. In Press, P., editor, *9th IFAC Word Congress Budapest*.

[Broyden, 1970] Broyden, C. G. (1970). The Convergence of a Class of Double-Rank Minimization Algorithms. *IMA Journal of Applied Mathematics*, 6:76–90.

[De Koning, 1982] De Koning, W. (1982). Infinite Horizon Optimal Control of Linear Discrete Time Systems with Stochastic Parameters. *Automatica*, 18(4):443–453.

[Deisenroth and Rasmussen, 2011] Deisenroth, M. and Rasmussen, C. E. (2011). Pilco: A Model-Based and Data-Efficient Approach to Policy Search. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 465–472.

[Deisenroth, 2008] Deisenroth, M. P. (2008). *Efficient Reinforcement Learning using Gaussian Processes*. PhD thesis, Karlsruhe Institute of Technology.

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38.

[Diehl, 2001] Diehl, M. (2001). *Real-Time Optimization for Large Scale Nonlinear Processes*. PhD thesis, Universität Heidelberg.

[Diehl et al., 2009] Diehl, M., Ferreau, H. J., and Haverbeke, N. (2009). Efficient Numerical Methods for nonlinear MPC and Moving Horizon Estimation. In *Nonlinear Model Predictive Control*, pages 391–417. Springer.

[Englert et al., 2013] Englert, P., Paraschos, A., Peters, J., and Deisenroth, M. P. (2013). Model-Based Imitation Learning by Probabilistic Trajectory Matching. In *Proceedings of the IEEE International Conference on Robotics and Automation*.

[Ferreau et al., 2008] Ferreau, H. J., Bock, H. G., and Diehl, M. (2008). An Online Active Set Strategy to Overcome the Limitations of Explicit MPC. *International Journal of Robust and Nonlinear Control*, 18(8):816–830.

[Fletcher, 1970] Fletcher, R. (1970). A New Approach to Variable Metric Algorithms. *The computer journal*, 13(3):317–322.

[Geiger and Kanzow, 2002] Geiger, C. and Kanzow, C. (2002). *Theorie und Numerik Restringierter Optimierungsaugaben*. Springer DE.

[Ghahramani and Hinton, 1996] Ghahramani, Z. and Hinton, G. E. (1996). Parameter Estimation for Linear Dynamical Systems. Technical report, Technical Report CRG-TR-96-2, University of Toronto, Dept. of Computer Science.

[Goldfarb, 1970] Goldfarb, D. (1970). A Family of Variable-Metric Methods derived by Variational Means. *Mathematics of computation*, 24(109):23–26.

[Gruene, 2007] Gruene, L. (2007). Stochastische Dynamische Optimierung, Vorlesungsskript Sommersemester 2007. Scriptum, Mathematisches Institut, Fakultät für Mathematik und Physik, Universität Bayreuth.

[Gruene and Pannek, 2011] Gruene, L. and Pannek, J. (2011). *Nonlinear Model Predictive Control*. Springer.

[Houska, 2011] Houska, B. (2011). *Robust Optimization of Dynamic Systems*. PhD thesis, KU Leuven.

[Houska et al., 2011] Houska, B., Ferreau, H. J., and Diehl, M. (2011). An Auto-Generated Real-Time Iteration Algorithm for Nonlinear MPC in the Microsecond Range. *Automatica*, 47(10):2279–2285.

[Jaynes, 2003] Jaynes, E. T. (2003). *Probability theory: the logic of science*. Cambridge university press.

[Julier and Uhlmann, 1997] Julier, S. J. and Uhlmann, J. K. (1997). New Extension of the Kalman Filter to Nonlinear Systems. In *AeroSense'97*, pages 182–193. International Society for Optics and Photonics.

[Kalman, 1960] Kalman, R. E. (1960). Contributions to the Theory of Optimal Control. *Bol. Soc. Mat. Mexicana*, 5(2):102–119.

[Kocijan et al., 2004] Kocijan, J., Murray-Smith, R., Rasmussen, C. E., and Girard, A. (2004). Gaussian Process Model Based Predictive Control. In *American Control Conference, 2004. Proceedings of the 2004*, volume 3, pages 2214–2219. IEEE.

[Kollar and Friedman, 2009] Kollar, D. and Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press.

[Kolmogorov, 1941] Kolmogorov, A. N. (1941). Interpolation und Extrapolation von Stationären Zufälligen Folgen. *Izv. Akad. Nauk SSSR,*, 5:3–14.

[Kosmol, 1993] Kosmol, P. (1993). *Methoden zur Numerischen Behandlung Nichtlinearer Gleichungen und Optimierungsaufgaben*. Teubner, 2nd edition.

[Krige, 1951] Krige, D. G. (1951). A Statistical Approach to some basic Mine Valuation Problems on the Witwatersrand. *Journal of the Chemical, Metallurgical and Mining Society of South Africa*, 52(6):119–139.

[Mardia and Marshall, 1984] Mardia, K. V. and Marshall, R. (1984). Maximum Likelihood Estimation of Models for Residual Covariance in Spatial Regression. *Biometrika*, 71(1):135–146.

[Marti and Qu, 1998] Marti, K. and Qu, S. (1998). Path Planning for Robots by Stochastic Optimization Methods. *Journal of Intelligent and Robotic Systems*, 22(2):117–127.

[Marti and Stein, 2012] Marti, K. and Stein, I. (2012). Optimal Structural Control under Stochastic Uncertainty: Robust Optimal Open-Loop Feedback Control. In *Mathematical Modelling*, volume 7, pages 270–275.

[Nemirovski and Shapiro, 2006] Nemirovski, A. and Shapiro, A. (2006). Convex Approximations of Chance Constrained Programs. *SIAM Journal on Optimization*, 17(4):969–996.

[Nguyen-Tuong et al., 2008] Nguyen-Tuong, D., Seeger, M., and Peters, J. (2008). Computed Torque Control with Nonparametric Regression Models. In *American Control Conference, 2008*, pages 212–217. IEEE.

[Petersen and Pedersen, 2006] Petersen, K. B. and Pedersen, M. S. (2006). The Matrix Cookbook.

[Platt et al., 2001] Platt, J. C., Burges, C. J., Swenson, S., Weare, C., and Zheng, A. (2001). Learning a Gaussian Process Prior for Automatically Generating Music Playlists. In *NIPS*, pages 1425–1432.

[Qin and Badgwell, 2003] Qin, S. J. and Badgwell, T. A. (2003). A Survey of Industrial Model Predictive Control Technology. *Control engineering practice*, 11(7):733–764.

[Quinonero-Candela et al., 2003] Quinonero-Candela, J., Girard, A., Larsen, J., and Rasmussen, C. E. (2003). Propagation of Uncertainty in Bayesian Kernel Models-Application to Multiple-Step Ahead Forecasting. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, volume 2, pages II–701. IEEE.

[Rasmussen and Williams, 2006] Rasmussen, C. E. and Williams, C. K. (2006). *Gaussian Process for Machine Learning. Adaptive Computation and Machine Learning*. MIT Press.

[Rauch et al., 1965] Rauch, H. E., Striebel, C., and Tung, F. (1965). Maximum Likelihood Estimates of Linear Dynamic Systems. *AIAA journal*, 3(8):1445–1450.

[Shanno, 1970] Shanno, D. F. (1970). Conditioning of Quasi-Newton Methods for Function Minimization. *Mathematics of computation*, 24(111):647–656.

[Sinz et al., 2004] Sinz, F. H., Candela, J. Q., Bakır, G. H., Rasmussen, C. E., and Franz, M. O. (2004). Learning Depth from Stereo. In *Pattern Recognition*, pages 245–252. Springer.

[Snelson and Ghahramani, 2006] Snelson, E. and Ghahramani, Z. (2006). Sparse Gaussian Processes using Pseudo-Inputs. In *Advances in Neural Information Processing Systems*, volume 18, pages 1257–1264.

[Snelson and Ghahramani, 2007] Snelson, E. and Ghahramani, Z. (2007). Local and Global Sparse Gaussian Process Approximations. In *International Conference on Artificial Intelligence and Statistics*, pages 524–531.

[Stoer et al., 1993] Stoer, J., Bulirsch, R., Bartels, R., Gautschi, W., and Witzgall, C. (1993). *Introduction to Numerical Analysis*, volume 2. Springer New York.

[Stoer et al., 2004] Stoer, J. et al. (2004). *Optimierung*. Springer DE.

[Van De Geer, 2000] Van De Geer, S. (2000). *Empirical Processes in M-Estimation*, volume 45. Cambridge University Press.

[Wiener, 1949] Wiener, N. (1949). Extrapolation, Interpolation and Smoothing of Stationary Time Series. *MIT Press, Cambridge*.

[Wu, 1983] Wu, C. (1983). On the Convergence Properties of the EM Algorithm. *The Annals of Statistics*, 11(1):95–103.

[Zantz, 2006] Zantz, T. (2006). *Einstufige Approximative Optimale Stochastische Planung neuer Robotertrajektorien in Echtzeit*. PhD thesis, Universität der Bundeswehr München, Universitätsbibliothek.