Technical University of Munich

The Role of Control Frequency for the Stability and Closed-Loop Performance of Uncertain Systems

A Thesis by

Ralf Römer

In Partial Fulfillment of the Requirements for the Degree

Master of Science
in Electrical Engineering and Information Technology

Supervisors: L. Brunke, Dr. S. Zhou, Prof. A. P. Schoellig

October 24, 2023

To my family.

Advisors: L. Brunke, Dr. S. Zhou, Prof. A. P. Schoellig *Ralf Römer*

The Role of Control Frequency for the Stability and Closed-Loop Performance of Uncertain Systems

ABSTRACT

Learning models or control policies from data have become powerful tools to improve the performance of uncertain systems. While a strong focus has been placed on increasing the amount and quality of data to improve performance, data can never entirely eliminate uncertainty, making feedback necessary to ensure stability and performance. In this thesis, we show that the control frequency at which the input is recalculated is a crucial design parameter, yet it has hardly been considered before. We address this gap by combining probabilistic model learning and sampled-data control. We use Bayesian linear regression and Gaussian processes to learn a continuous-time dynamics model and compute a corresponding discrete-time controller. The result is an uncertain sampled-data control system, for which we derive robust stability conditions. We formulate optimization problems to compute the minimum control frequency required for stability and to optimize performance. Based on these, we can robustly control an uncertain system at different control frequencies without adapting or relearning the model. We leverage this to study the effect of both control frequency and data on stability and closed-loop performance. We show through numerical simulations and real-world experiments with a quadrotor that feedback and learning must each be employed to at least some extent to stabilize the system. Control performance can be improved by increasing either the amount of data or the control frequency, and we can trade off one for the other. For example, by increasing the control frequency by 33% in the simulation, we can reduce the number of data points by half while still achieving similar performance. However, throughout all our experiments, a greater performance improvement can be achieved by increasing the control frequency than by collecting more data. Therefore, our results motivate considering fast feedback as a powerful alternative to the often computationally expensive learning-based control approaches.

Contents

List of Figures

List of Tables

Acronyms

BLR Bayesian Linear Regression

GEVP Generalized Eigenvalue Problem

GP Gaussian Process

i.i.d Independent and Identically Distributed

LKF Lyapunov-Krasovskii Functional

LMI Linear Matrix Inequality

LQR Linear Quadratic Regulator

MCF Minimum Control Frequency

MPC Model Predictive Control

RMSE Root-Mean-Square Error

ZOH Zero-Order-Hold

Chapter 1

Introduction

1.1 Motivation

Real-world systems such as robots can exhibit complex dynamics, making deriving accurate models from first principles difficult. Therefore, many studies in recent years have addressed learning unknown dynamics from measured data using machine learning methods and designing a controller based on the learned model [1, 2, 3, 4], as illustrated in Fig. 1.1. Much attention has been paid to the role of data [5] and increasing its amount and quality [6]. However, no derived or learned model can perfectly capture the dynamic behavior of a real-world system [7]. Therefore, feedback [8] is required to guarantee stability and performance despite uncertainties.

The control frequency at which system measurements are fed back to recalculate the control input is often set without taking the dynamics and uncertainty into account [1, 3], neglecting that it represents a degree of freedom in the controller design. However, considering the control frequency as a design parameter can be advantageous, especially for resource-constrained systems such as robot platforms (e.g., drones). For example, knowledge of the minimum control frequency (MCF) required for guaranteed stability of an uncertain system can help improve energy efficiency by reducing unnecessary compu-

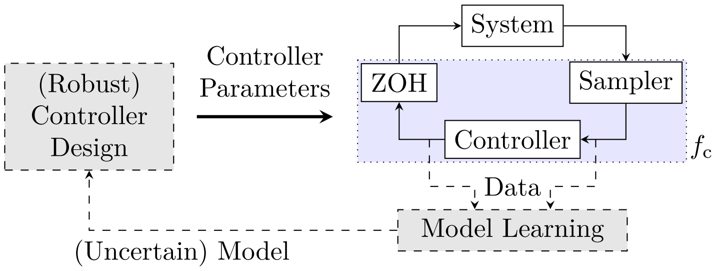

Figure 1.1: In digital control systems (blue shaded box), the sampler, controller, and zero-order-hold (ZOH) operate at a certain control frequency f_c. Many studies in recent years have revolved around using machine learning methods to learn a dynamics model from data and then designing a model-based controller from the learned model, however, without taking the control frequency into account as a design parameter.

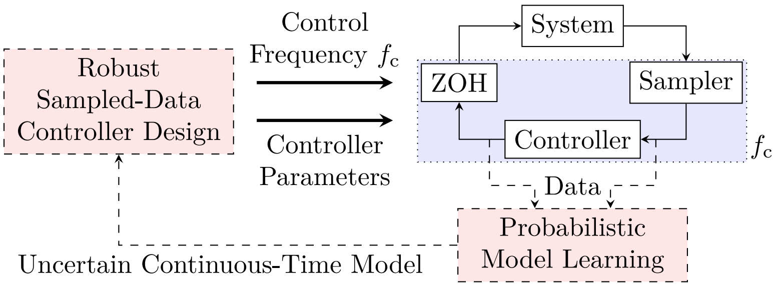

Figure 1.2: We propose a framework (dashed-line boxes) to design a robust controller operating at a desired frequency using an uncertain continuous-time dynamics model learned from data using either Bayesian linear regression or Gaussian processes.

tational demand and data transmission. In this work, we study the effect of both control frequency and data on closed-loop performance.

Learning a discrete-time model of the unknown dynamics yields a model specific to a particular sampling time. Analyzing the system's behavior and stability for a different controller sampling time based on this learned model is generally very difficult. Thus, we consider learning a continuous-time model in this work. Designing a discrete-time controller for a continuous-time system falls within the domain of sampled-data control [9]. We propose a framework combining probabilistic model learning with robust sampled-data control, which is illustrated in Fig. 1.2. In contrast to the standard approach [1, 3], we can flexibly choose the control frequency without changing the model or sacrificing theoretical guarantees with our framework, allowing us to investigate the effect of varying only the control frequency.

Before discussing the related work, we present a simple toy example illustrating the role of control frequency for uncertain systems. Therein, we consider two cases: The controller design is based on a perfect or an inaccurate dynamics model.

Example 1.1. *Consider a dynamical system with state $\boldsymbol{x}(t) \in \mathbb{R}^2$ and input $u(t) \in \mathbb{R}$ at time $t \in \mathbb{R}_{\geq 0}$ that evolves according to the linear continuous-time dynamics*

$$\dot{\boldsymbol{x}}(t) = \underbrace{\begin{bmatrix} 0 & 1 \\ 2 & -1 \end{bmatrix}}_{\boldsymbol{A}} \boldsymbol{x}(t) + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{\boldsymbol{b}} u(t), \qquad \boldsymbol{x}(0) = \boldsymbol{x}_0. \tag{1.1}$$

The eigenvalues of $\boldsymbol{A}$ are $\lambda_1 = -1$, $\lambda_2 = 2$, i.e., the open-loop system with $\boldsymbol{u}(t) = 0$ is unstable. We consider a discrete-time linear quadratic regulator (LQR) for stabilizing (1.1) to the origin. For this, (1.1) is discretized with a sampling time T_s assuming a zero-order hold input, i.e., $u(t)$ is kept constant during each time interval $t \in [t_k, t_{k+1})$, $k \in \mathbb{N}_0$, where $t_k = kT_\mathrm{s}$. Denoting the discrete-time state and input by $\boldsymbol{x}_k = \boldsymbol{x}(kT_\mathrm{s})$ and

$u_k = u(kT_\mathrm{s})$, respectively, we obtain the discrete-time dynamics

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_\mathrm{d}\boldsymbol{x}_k + \boldsymbol{b}_\mathrm{d}u_k, \tag{1.2}$$

where $\boldsymbol{A}_\mathrm{d} = \mathrm{e}^{\boldsymbol{A}T_\mathrm{s}}$ and $\boldsymbol{b}_\mathrm{d} = \boldsymbol{A}^{-1}(\boldsymbol{A}_\mathrm{d} - \boldsymbol{I})\boldsymbol{b}$. Discrete-time LQR uses the linear state feedback law $u_k = -\boldsymbol{k}^\top \boldsymbol{x}_k$, where the control gain $\boldsymbol{k} \in \mathbb{R}^2$ solves the discrete-time algebraic Riccati equation [10], minimizing the infinite-horizon cost

$$J_\infty = \frac{1}{2}\sum_{t=0}^{\infty} \boldsymbol{x}_k^\top \boldsymbol{Q}\boldsymbol{x}_k + Ru_k(t)^2. \tag{1.3}$$

We set the weighting matrices to $\boldsymbol{Q} = \boldsymbol{I}_2$ and $R = 0.01$. With the obtained discrete-time controller, the continuous-time system evolution is calculated by numerical integration of (1.1) with $u(t) = u_k$ for $t \in [t_k, t_{k+1})$, $k \in \mathbb{N}_0$. We evaluate the impact of the control frequency $f_\mathrm{c} = \frac{1}{T_\mathrm{s}}$ on the closed-loop dynamics, considering the initial state $\boldsymbol{x}_0 = [5,\ 1]^\top$ and different sampling times $T_\mathrm{s} \in \{0.05, 0.06, \ldots, 1\}$. For each, we simulate the system for a duration $t \in [0, T]$, where $T = 10$, and calculate the continuous-time finite-horizon cost.

$$J = \frac{1}{2}\int_{t=0}^{T} \boldsymbol{x}(t)^\top \boldsymbol{Q}\boldsymbol{x}(t) + Ru(t)^2 \,\mathrm{d}t. \tag{1.4}$$

Here, a finite horizon is used to ensure boundedness of the cost. We also consider the case of model mismatch, i.e., the controller is designed using an inaccurate discrete-time dynamics model that is based on

$$\hat{\boldsymbol{A}} = \begin{bmatrix} 0 & 1 \\ 1 & -1 \end{bmatrix}, \quad \hat{\boldsymbol{b}} = \begin{bmatrix} 0 \\ 1.5 \end{bmatrix}, \tag{1.5}$$

instead of $\boldsymbol{A}$ and $\boldsymbol{b}$. The achieved costs for both cases are shown on the left in Fig. 1.3, and the state trajectories for $T_\mathrm{s} = 0.4$ are shown on the right in Fig. 1.3. It can be seen that when the model is perfect, the control performance improves with increasing control frequency but very slightly for $f_\mathrm{c} > 5$. In the case of model mismatch, the control performance degrades significantly for small control frequencies. The system even becomes unstable for $f_\mathrm{c} < 2.13$. In contrast, when using a high control frequency, the achieved cost becomes very close to the controller designed with a perfect system model. We also observe that if the control frequency is high enough, the performance in the case of model mismatch can even exceed the performance achieved with a perfect model at lower control frequencies.

This example shows that the control frequency strongly influences the stability and performance of a control system, especially when the system dynamics are not precisely known. Investigating the role of control frequency is, thus, especially important in learning-based control, where there is typically a mismatch between the true and the

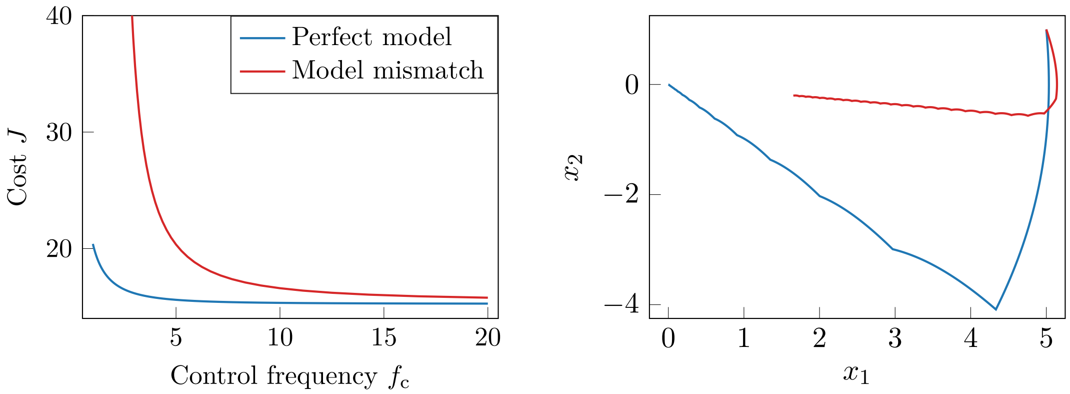

Figure 1.3: Left: Finite-horizon cost (1.4) for controlling system (1.1) with a discrete-time linear-quadratic regulator at different control frequencies f_c with and without model mismatch. Right: Corresponding state trajectories for $f_c = 2.5$.

learned dynamics model.

1.2 Related Work

For quantifying the uncertainty associated with a learned dynamics model, probabilistic methods such as Bayesian linear regression (BLR) [11, 12] or Gaussian processes (GPs) [13] have become popular [4]. Compared to BLR, which assumes linearity in a set of parameters, GPs offer the advantage of being a non-parametric method. In [1, 2, 3], GP model learning is combined with robust control methods. While these works demonstrate that the model uncertainty is related to the control performance, the connection between model learning and the control frequency is not investigated.

From an optimal control perspective [14], the control frequency or the sampling time affects how fine the input trajectory can be discretized, i.e., the size of the solution space for solving the optimal control problem. For example, the classic work [15] shows through a sensitivity analysis that for discrete-time LQR with a small sampling time, the cost increases approximately quadratically with the sampling time. In [16], the impact of the sampling time on the stability and performance of unconstrained nonlinear model predictive control (MPC) is analyzed. The authors show that the achieved cost in the discrete-time case decreases monotonically with the sampling time and that the continuous-time performance can be approximated arbitrarily well for a sufficiently small sampling time.

Sampled-data systems offer the advantage that the controller's discrete-time nature can already be considered in the controller design without discretizing the model using a certain sampling time. Numerous approaches have been proposed to analyze the stability of sampled-data systems; see the recent surveys [17, 9]. Most of the works revolve around the time-delay approach, which is explained in detail in [18]. Based on this approach, stability conditions are derived, for example, in [19, 20, 21]. These conditions can also guarantee robust stability for polytopic-type uncertainty. However, this quickly becomes computationally intractable for increasing state and input dimensions for the uncertainty set associated with a dynamics model learned from data.

Few studies have considered the control frequency in the context of model uncertainty. In [22], the maximum sampling interval for stabilizing an unknown linear system is computed from measured data following a direct data-driven paradigm. However, the approach assumes bounded noise and involves a computationally expensive iterative optimization scheme for controller design. GP-based feedback linearization with a data-dependent delay for updating the control input is considered in [23]. The authors show empirically that in terms of tracking accuracy, there can be a tradeoff between the accuracy of the GP model and the computational delay. Taking a different perspective, reducing the control frequency is discussed in [24] in the context of reinforcement learning as a way to reduce the sample complexity and thus improve performance.

The extent to which high-frequency recalculation of the control input can compensate for a lack of data has not yet been investigated.

1.3 Contributions

We propose a framework to design the control frequency based on the uncertainty associated with a dynamics model learned from data as illustrated in Fig. 1.2 and study the role of the control frequency compared to the amount of data. Taking a sampled-data control approach, we robustly stabilize a partially unknown nonlinear continuous-time system with a discrete-time controller. Our main contributions are:

- We propose a framework combining stochastic model learning using BLR and GPs with sampled-data control to independently study the effect of the control frequency and training data on the closed-loop performance.

- We derive robust stability conditions as matrix inequalities for a sampled-data control system with learned uncertain dynamics. Based on these, we formulate optimization problems for the computation of the MCF and performance optimization. We show that the number of decision variables in the first optimization problem can be reduced by one, reducing computational complexity. In addition, we derive a lower bound on the number of training samples required to robustly stabilize a system at a given control frequency. Our framework enables us to robustly control an uncertain system at different frequencies based on the same continuous-time model.

- We evaluate our framework in numerical simulations of a 2D quadrotor and hardware experiments with a real quadrotor. Thereby, we show and analyze the tradeoff between model uncertainty, affected by the amount of data collected, and control frequency with respect to stability and performance.

We conclude the introductory chapter by outlining the structure of the thesis and explaining the notation.

1.4 Outline of the Thesis

The remainder of this thesis is organized as follows: Chapter 2 reviews some mathematical concepts heavily used in this thesis and introduces probabilistic model learning with BLR and GPs as well as sampled-data systems. The following two chapters are devoted to the different components of our proposed framework and their theoretical analysis. In Chapter 3, we discuss probabilistic learning of linearized dynamics for sampled-data control and a necessary reparameterization of the model uncertainty. The design of a robust sampled-data controller for the learned uncertain model is treated in Chapter 4. Here, we start with deriving robust stability conditions and use these to formulate two optimization problems regarding stability and performance, respectively. We then employ our framework in Chapter 5, first in numerical simulation of a 2D quadrotor and then in experiments with a real quadrotor. Finally, we draw our main conclusions and wrap up the thesis with some ideas for future work in Chapter 6.

1.5 Notation

We employ the following notation in this thesis:

- The sets of positive and non-negative integers up to n are denoted by $\mathbb{I}_n = \{1, \dots, n\}$ and $\mathbb{I}_n^0 = \{0, 1, \dots, n\}$, respectively.
- The identity matrix is denoted by $\boldsymbol{I}_n \in \mathbb{R}^{n \times n}$, the one-vector by $\mathbf{1}_n \in \mathbb{R}^n$ and the one-matrix by $\mathbf{1}_{n\times m} \in \mathbb{R}^{n\times m}$. The zero-vector and zero-matrix are similarly denoted by $\mathbf{0}_n \in \mathbb{R}^n$ and $\mathbf{0}_{n\times m} \in \mathbb{R}^{n\times m}$, respectively. The dimensions are omitted if they can be inferred from context.
- The i-th element of a vector $\boldsymbol{a}$ is denoted by a_i. The element in the i-th row and j-th column of a matrix $\boldsymbol{A}$ is denoted by A_{ij}.
- The vector containing the diagonal elements of a square matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is defined as $\operatorname{diag}(\boldsymbol{A}) \in \mathbb{R}^n$. Given a vector $\boldsymbol{b} \in \mathbb{R}^n$, $\boldsymbol{B} = \operatorname{Diag}(\boldsymbol{b}) \in \mathbb{R}^{n\times n}$ is defined as the diagonal matrix with $B_{ii} = b_i$, $\forall i \in \mathbb{I}_n$.
- In symmetric matrices, $*$ denotes transpose elements that can be inferred from symmetry.
- Stacking two vectors $\boldsymbol{a} \in \mathbb{R}^n$ and $\boldsymbol{b} \in \mathbb{R}^m$ is denoted by $(\boldsymbol{a}, \boldsymbol{b}) = \left[\boldsymbol{a}^\top, \boldsymbol{b}^\top\right]^\top \in \mathbb{R}^{n+m}$.

- Operators (e.g., the square root) act element-wise if applied to vectors or matrices.
- Positive (negative) definiteness of a square matrix $\boldsymbol{A}$ is denoted by $\boldsymbol{A} \succ \boldsymbol{0}$ ($\boldsymbol{A} \prec \boldsymbol{0}$) and positive (negative) semi-definiteness of $\boldsymbol{A}$ by $\boldsymbol{A} \succeq \boldsymbol{0}$ ($\boldsymbol{A} \preceq \boldsymbol{0}$).
- The Hadamard (element-wise) product of two matrices matrices $\boldsymbol{A} \in \mathbb{R}^{n \times m}$ and $\boldsymbol{B} \in \mathbb{R}^{n \times m}$ is denoted by $\boldsymbol{C} = \boldsymbol{A} \circ \boldsymbol{B}$, where $C_{ij} = A_{ij} B_{ij}$.
- The spectral norm of a real matrix $\boldsymbol{A}$ is denoted by $\|\boldsymbol{A}\|_2 = \sqrt{\lambda_{\max}(\boldsymbol{A}^\top \boldsymbol{A})}$, where $\lambda_{\max}(\cdot)$ denotes the largest eigenvalue.

Chapter 2

Preliminaries

This chapter summarizes the technical background for this thesis. After briefly treating some relevant mathematical concepts, we discuss two popular approaches to probabilistic model learning before finally introducing sampled-data systems and their stability analysis.

2.1 Mathematical Background

This section introduces some important mathematical concepts and results that are heavily used in Chapters 3 and 4, respectively.

2.1.1 Confidence Regions for Multivariate Gaussian Distributions

A robust treatment of randomly distributed model uncertainties or disturbances usually requires the computation of bounded regions containing the random variables with at least a given probability. In the following, we provide two formulations of such confidence regions for the special case of Gaussian distributions.

Lemma 2.1 ([25], Lemma 2)**.** *Let* $\boldsymbol{d} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ *be a random variable in* $\mathbb{R}^n$*. Then, for any* $p \in [0, 1)$*,*

$$\Pr\left(\boldsymbol{d} \in \boldsymbol{\mu} + \mathcal{E}\right) = p, \tag{2.1}$$

where $\mathcal{E} \subset \mathbb{R}^n$ *is an ellipsoidal set defined by*

$$\mathcal{E} = \left\{ \boldsymbol{d} \in \mathbb{R}^n \;\middle|\; \boldsymbol{d}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{d} \leq \chi^2_n(p) \right\}, \tag{2.2}$$

and χ^2_n *is the quantile function of the chi-squared distribution with* n *degrees of freedom.*

Computing the axis-aligned bounding box of the ellipsoid (2.2) yields an alternative representation of a confidence region.

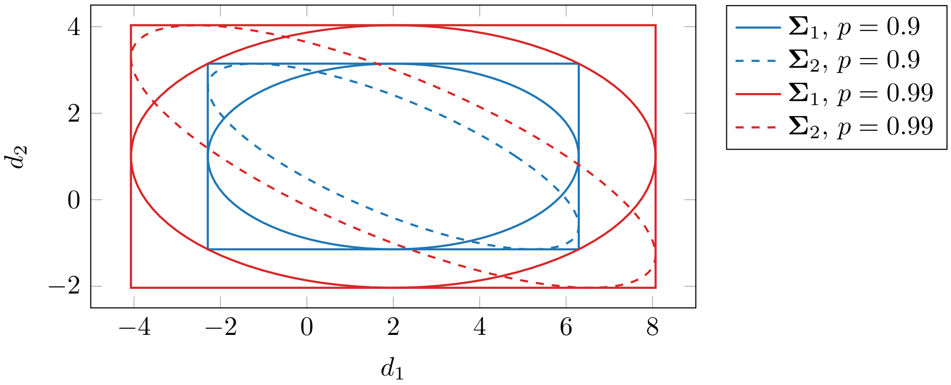

Figure 2.1: Illustration of the ellipsoidal and box-shaped confidence regions for two Gaussian distributions $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}_1)$ and $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}_2)$ with mean $\boldsymbol{\mu} = [2,\ 1]^\top$ and covariance matrices $\boldsymbol{\Sigma}_1 = \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix}$ and $\boldsymbol{\Sigma}_2 = \begin{bmatrix} 4 & -1.5 \\ -1.5 & 1 \end{bmatrix}$ for two different confidence thresholds p.

Lemma 2.2 (adopted from [25], Lemma 1)**.** *Let $\boldsymbol{d} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be a random variable in $\mathbb{R}^n$. Then, for any $p \in [0, 1)$,*

$$\Pr\left(\boldsymbol{d} \in \boldsymbol{\mu} + \mathcal{B}\right) \geq p, \tag{2.3}$$

where $\mathcal{B} \subset \mathbb{R}^{n_d}$ is a hyperrectangle or box defined by

$$\mathcal{B} = \left[-\sqrt{\chi^2_n(p)\Sigma_{11}}, \sqrt{\chi^2_n(p)\Sigma_{11}}\right] \times \cdots \times \left[-\sqrt{\chi^2_n(p)\Sigma_{nn}}, \sqrt{\chi^2_n(p)\Sigma_{nn}}\right]. \tag{2.4}$$

While the bounding box (2.4) is more conservative than the ellipsoid (2.2), it can sometimes be more computationally efficient to use as it provides one-dimensional confidence intervals for each dimension of $\boldsymbol{d}$. The two types of confidence regions are illustrated in the following example.

Example 2.1. *We consider two normal distributions $\mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $\mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_2)$ in $\mathbb{R}^2$ with the same mean $\boldsymbol{\mu} = [2,\ 1]^\top$ and different covariance matrices*

$$\boldsymbol{\Sigma}_1 = \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix} \quad \textit{and} \quad \boldsymbol{\Sigma}_2 = \begin{bmatrix} 4 & -1.5 \\ -1.5 & 1 \end{bmatrix}.$$

The corresponding ellipsoidal and box-shaped confidence regions for the confidence thresholds $p = 0.9$ and $p = 0.99$, respectively, are depicted in Fig. 2.1. We observe the confidence regions to grow when p is increased. Moreover, the off-diagonal elements of the covariance matrix do not affect the bounding box, which is also evident from (2.4).

As demonstrated in the example, larger off-diagonal entries of the covariance matrix increase the degree of over-approximation of the ellipsoidal confidence region by the bounding box.

2.1.2 Linear Matrix Inequalities and Generalized Eigenvalue Problems

Consider an optimization problem whose decision or optimization variables are stacked into a vector $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_m]^\top \in \mathbb{R}^m$. Constraints of the form

$$\boldsymbol{A}(\boldsymbol{\theta}) = \boldsymbol{A}_0 + \theta_1 \boldsymbol{A}_1 + \cdots + \theta_m \boldsymbol{A}_m \succ \boldsymbol{0}, \tag{2.5}$$

where $\boldsymbol{A}_0, \ldots, \boldsymbol{A}_n$ are given symmetric matrices, are called linear matrix inequality (LMI) constraints [26]. The LMI (2.5) is convex in $\boldsymbol{\theta}$, as $\boldsymbol{A}(\alpha\boldsymbol{\theta} + (1-\alpha)\boldsymbol{\theta}') \succ \boldsymbol{0}$ for all $\alpha \in [0,1]$ if $\boldsymbol{A}(\boldsymbol{\theta}) \succ \boldsymbol{0}$ and $\boldsymbol{A}(\boldsymbol{\theta}') \succ \boldsymbol{0}$. Hence, the feasible set of an LMI-constrained optimization problem is a convex set, facilitating a numerical solution [27].

Many problems arising in control theory can be cast into LMI-constrained optimization problems. An often encountered special type of such optimization problem is

$$\begin{aligned} \min_{\boldsymbol{\theta},\lambda} \quad & \lambda \\ \text{s.t.} \quad & \lambda \boldsymbol{B}(\boldsymbol{\theta}) - \boldsymbol{A}(\boldsymbol{\theta}) \succ \boldsymbol{0}, \quad \boldsymbol{B}(\boldsymbol{\theta}) \succ \boldsymbol{0}, \quad \boldsymbol{C}(\boldsymbol{\theta}) \succ \boldsymbol{0}, \end{aligned} \tag{2.6}$$

with optimization variables $\boldsymbol{\theta} \in \mathbb{R}^m$ and $\lambda \in \mathbb{R}$, and symmetric matrices $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ that are affine functions of $\boldsymbol{\theta}$. The problem (2.6) is also referred to as generalized eigenvalue problem (GEVP) because it can be expressed as

$$\begin{aligned} \min_{\boldsymbol{\theta},\lambda} \quad & \lambda_{\max}(\boldsymbol{A}(\boldsymbol{\theta}), \boldsymbol{B}(\boldsymbol{\theta})) \\ \text{s.t.} \quad & \boldsymbol{B}(\boldsymbol{\theta}) \succ \boldsymbol{0}, \quad \boldsymbol{C}(\boldsymbol{\theta}) \succ \boldsymbol{0}, \end{aligned} \tag{2.7}$$

where $\lambda_{\max}(\boldsymbol{A}, \boldsymbol{B})$ denotes the largest generalized eigenvalue of $\boldsymbol{A}$ and $\boldsymbol{B}$, i.e., the largest value λ such that $\det(\boldsymbol{A} - \lambda\boldsymbol{B}) = 0$. The objective function $\lambda_{\max}(\boldsymbol{A}(\boldsymbol{\theta}), \boldsymbol{B}(\boldsymbol{\theta}))$ is quasiconvex in $\boldsymbol{\theta}$. Therefore, GEVPs are quasiconvex optimization problems that can be solved efficiently in polynomial time [26], for example, using the bisection method or interior-point algorithms [27].

In special cases, nonlinear matrix inequality constraints can be transformed into LMIs, making the optimization problem convex and thus easier to solve by utilizing the following well-known result, which is stated, for instance, in [26].

Lemma 2.3 (Schur Complement)**.** *Given matrices $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$, the following equivalence holds:*

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{B}^\top & \boldsymbol{C} \end{bmatrix} \succ \boldsymbol{0} \quad \Leftrightarrow \quad \boldsymbol{C} \succ \boldsymbol{0}, \quad \boldsymbol{A} - \boldsymbol{B}\boldsymbol{C}^{-1}\boldsymbol{B}^\top \succ \boldsymbol{0}. \tag{2.8}$$

Here, $\boldsymbol{A} - \boldsymbol{B}\boldsymbol{C}^{-1}\boldsymbol{B}^\top$ is called the Schur complement of the block $\boldsymbol{C}$ of $\boldsymbol{M}$.

For a comprehensive treatment of LMIs and their use in control theory, refer to [26].

2.2 Probabilistic Machine Learning Methods

We introduce two probabilistic machine learning methods for approximating unknown functions from input/output data: Bayesian linear regression (BLR) and Gaussian processes (GPs). Both methods can incorporate prior information about the unknown function but make different assumptions about its structure. To simplify notation, we consider learning a scalar function $g : \mathbb{R}^{n_z} \to \mathbb{R}$. We assume the availability of noisy training data collected through multiple evaluations of g, as formalized in the following.

Assumption 2.1. *A training data set of noise-free training inputs and noisy training targets*

$$\mathcal{D} = \left\{ \boldsymbol{z}^{(i)},\, y^{(i)} = g(\boldsymbol{z}^{(i)}) + w^{(i)} \right\}_{i=1}^{N} \tag{2.9}$$

is available, where $w^{(i)} \sim \mathcal{N}(0,\, \sigma_{\mathrm{n}}^2)$, $\forall i \in \mathbb{I}_N$, is independent and identically distributed (i.i.d) Gaussian noise.

2.2.1 Bayesian Linear Regression

BLR [11] is based on the assumption that the unknown function g is linear with respect to a parameter vector $\boldsymbol{\theta} \in \mathbb{R}^m$, i.e.,

$$g(\boldsymbol{z}) = \boldsymbol{\theta}^\top \boldsymbol{\phi}(\boldsymbol{z}), \tag{2.10}$$

where the feature vector $\boldsymbol{\phi}(\boldsymbol{z}) = \left[\phi_1(\boldsymbol{z}), \ldots, \phi_{n_\theta}(\boldsymbol{z})\right]^\top$ consists of n_θ scalar feature functions $\phi_i : \mathbb{R}^{n_z} \to \mathbb{R}$, $i = 1, \ldots, n_\theta$. Under this assumption, the observed data (2.9) can be written compactly in matrix-vector form as

$$\boldsymbol{y} = \boldsymbol{\Phi}^\top \boldsymbol{\theta} + \boldsymbol{w}, \tag{2.11}$$

where the vector $\boldsymbol{y} = \left[y^{(1)}, \ldots, y^{(N)}\right]^\top \in \mathbb{R}^N$ contains the observed targets, the matrix $\boldsymbol{\Phi} = \left[\boldsymbol{\phi}(\boldsymbol{z}^{(1)}), \ldots, \boldsymbol{\phi}(\boldsymbol{z}^{(N)})\right] \in \mathbb{R}^{n_\theta \times N}$ contains the feature vectors of the training inputs, and $\boldsymbol{w} \sim \mathcal{N}\big(\boldsymbol{0},\, \sigma_{\mathrm{n}}^2 \boldsymbol{I}\big)$. In standard linear regression, which also makes the linearity assumption (2.10), the parameter vector $\boldsymbol{\theta}$ minimizing $\|\boldsymbol{y} - \boldsymbol{\Phi}^\top \boldsymbol{\theta}\|_2$ is obtained via computation of the pseudo-inverse of $\boldsymbol{\Phi}^\top$. In contrast, BLR additionally assumes a Gaussian prior $\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}},\, \boldsymbol{\Sigma}_{\boldsymbol{\theta}})$ on the parameter vector. As a result, $\boldsymbol{w}$ and $\boldsymbol{\theta}$ follow a joint Gaussian distribution

$$\begin{bmatrix} \boldsymbol{w} \\ \boldsymbol{\theta} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{\mu}_{\boldsymbol{\theta}} \end{bmatrix}, \begin{bmatrix} \sigma_{\mathrm{n}}^2 \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \end{bmatrix} \right). \tag{2.12}$$

It follows from the linear transformation

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{\theta} \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{\Phi}^\top \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{w} \\ \boldsymbol{\theta} \end{bmatrix}, \tag{2.13}$$

that $\boldsymbol{y}$ and $\boldsymbol{\theta}$ are also jointly Gaussian distributed as

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{\theta} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\Phi}^\top \boldsymbol{\mu}_{\boldsymbol{\theta}} \\ \boldsymbol{\mu}_{\boldsymbol{\theta}} \end{bmatrix}, \begin{bmatrix} \sigma_{\mathrm{n}}^2 \boldsymbol{I} + \boldsymbol{\Phi}^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \boldsymbol{\Phi} & \boldsymbol{\Phi}^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \\ * & \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \end{bmatrix} \right). \tag{2.14}$$

The posterior parameter distribution can be obtained by Gaussian conditioning [28] of $\boldsymbol{\theta}$ on the observed data $\boldsymbol{y}$, yielding

$$\boldsymbol{\theta} \,|\, \mathcal{D} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}|\mathcal{D}}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}|\mathcal{D}}), \tag{2.15}$$

where the posterior parameter mean and variance are given by

$$\boldsymbol{\mu}_{\boldsymbol{\theta}|\mathcal{D}} = \boldsymbol{\mu}_{\boldsymbol{\theta}} + \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \boldsymbol{\Phi} \left(\sigma_{\mathrm{n}}^2 \boldsymbol{I} + \boldsymbol{\Phi}^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \boldsymbol{\Phi} \right)^{-1} (\boldsymbol{y} - \boldsymbol{\Phi}^\top \boldsymbol{\mu}_{\boldsymbol{\theta}}) \tag{2.16}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{\theta}|\mathcal{D}} = \boldsymbol{\Sigma}_{\boldsymbol{\theta}} - \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \boldsymbol{\Phi} \left(\sigma_{\mathrm{n}}^2 \boldsymbol{I} + \boldsymbol{\Phi}^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi}^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}}. \tag{2.17}$$

Remark 2.1. *If no prior information on the parameter vector $\boldsymbol{\theta}$ is available, corresponding to $\boldsymbol{\Sigma}_{\boldsymbol{\theta}} = \gamma \boldsymbol{I}$ with $\gamma \to \infty$, then (2.16) and (2.17) simplify to*

$$\boldsymbol{\mu}_{\boldsymbol{\theta}|\mathcal{D}} = \boldsymbol{\Phi}(\boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1} \boldsymbol{y} \tag{2.18}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{\theta}|\mathcal{D}} = \boldsymbol{0}, \tag{2.19}$$

This corresponds to the well-known result for standard linear regression.

Consider the case that the data set (2.9) contains only a single observation, i.e., $\mathcal{D} = \{(\boldsymbol{z}, y)\}$. Then, the expressions (2.16) and (2.17) for the posterior mean and variance become

$$\boldsymbol{\mu}_{\boldsymbol{\theta}|\mathcal{D}} = \boldsymbol{\mu}_{\boldsymbol{\theta}} + \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \boldsymbol{\phi}(\boldsymbol{z})(\sigma_{\mathrm{n}}^2 + \boldsymbol{\phi}(\boldsymbol{z})^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \boldsymbol{\phi}(\boldsymbol{z}))^{-1} (y - \boldsymbol{\phi}(\boldsymbol{z})^\top \boldsymbol{\mu}_{\boldsymbol{\theta}}) \tag{2.20}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{\theta}|\mathcal{D}} = \boldsymbol{\Sigma}_{\boldsymbol{\theta}} - \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \boldsymbol{\phi}(\boldsymbol{z})(\sigma_{\mathrm{n}}^2 + \boldsymbol{\phi}(\boldsymbol{z})^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \boldsymbol{\phi}(\boldsymbol{z}))^{-1} \boldsymbol{\phi}(\boldsymbol{z})^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}}. \tag{2.21}$$

The Sherman-Morrison formula [29] can be employed to rewrite (2.20) and (2.21), yielding

$$\boldsymbol{\mu}_{\boldsymbol{\theta}|\mathcal{D}} = \boldsymbol{\Sigma}_{\boldsymbol{\theta}|\mathcal{D}} \left(\boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{-1} \boldsymbol{\mu}_{\boldsymbol{\theta}} + \frac{1}{\sigma_{\mathrm{n}}^2} \boldsymbol{\phi}(\boldsymbol{z}) y \right) \tag{2.22}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{\theta}|\mathcal{D}} = \left(\boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{-1} + \frac{1}{\sigma_{\mathrm{n}}^2} \boldsymbol{\phi}(\boldsymbol{z}) \boldsymbol{\phi}(\boldsymbol{z})^\top \right)^{-1}. \tag{2.23}$$

The expressions (2.22) and (2.23) can alternatively be derived by applying Bayes' rule

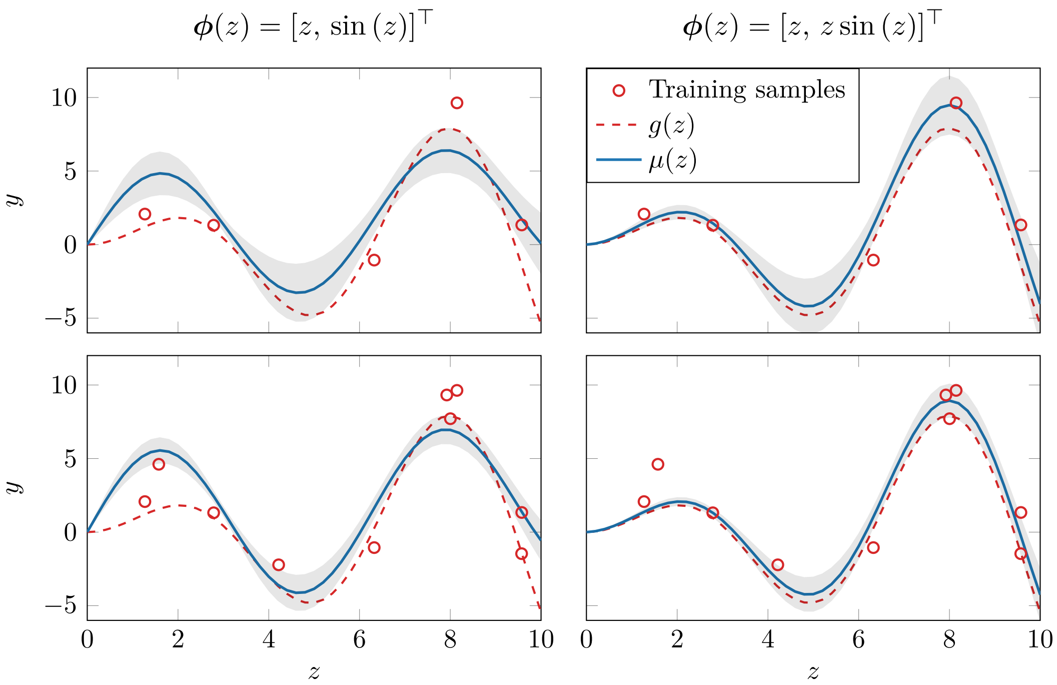

Figure 2.2: Bayesian linear regression for learning the unknown function $g(z) = z\sin(z)$ from $N = 5$ (top) and $N = 10$ (bottom) training samples for two different choices of features. The shaded regions correspond to ± two standard deviations.

and making use of the fact that the Gaussian distribution is the conjugate prior of itself; see [11]. In addition to the posterior parameter distribution, many applications require evaluating the predictive distribution of the function value at a query input $\boldsymbol{z}^* \in \mathbb{R}^{n_z}$. As the prediction model (2.10) is linear, the predictive distribution is also Gaussian; given by $g(\boldsymbol{z}^*) \,|\, \mathcal{D} \sim \mathcal{N}(\mu(\boldsymbol{z}^*),\, \sigma^2(\boldsymbol{z}^*))$ with mean and variance

$$\mu(\boldsymbol{z}^*) = \boldsymbol{\phi}(\boldsymbol{z}^*)^\top \boldsymbol{\mu}_{\boldsymbol{\theta}|\mathcal{D}} \tag{2.24}$$

$$\sigma^2(\boldsymbol{z}^*) = \boldsymbol{\phi}(\boldsymbol{z}^*)^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}|\mathcal{D}} \boldsymbol{\phi}(\boldsymbol{z}^*). \tag{2.25}$$

Next, we consider a simple example to highlight some key properties of BLR.

Example 2.2 (Bayesian linear regression)**.** *We aim to learn the unknown function $g(z) = z\sin(z)$ from N training inputs, which are drawn uniformly from the interval $[0, 10]$ with observation noise variance $\sigma_{\mathrm{n}}^2 = 1$. We consider two choices of feature vectors: $\boldsymbol{\phi}(z) = [z,\, \sin(z)]^\top$ and $\boldsymbol{\phi}(z) = [z,\, z\sin(z)]^\top$. The training data and the learned functions are shown in Figure 2.2 for $N = 5$ and $N = 10$ training samples, respectively. The second feature vector results in a more accurate approximation of g as its second component is equivalent to g. We observe that more data reduces the prediction uncertainty. The uncertainty can be made arbitrarily small if sufficient data is available. We can also see that the prediction uncertainty increases quadratically with the feature* *values, which is evident from* (2.25).

2.2.2 Gaussian Process Regression

GP regression [13] is based on the assumption that the unknown function g is drawn from a GP, denoted as $g \sim \mathcal{GP}(m(\cdot), k(\cdot, \cdot))$. A GP induces a distribution over functions such that any finite number of function evaluations is jointly Gaussian distributed. The prior mean function $m : \mathbb{R}^{n_z} \to \mathbb{R}$ can incorporate prior knowledge in the form of an approximate model, and the kernel $k : \mathbb{R}^{n_z} \times \mathbb{R}^{n_z} \to \mathbb{R}$ encodes information about the structure of the unknown function. The mean function can be set to zero without loss of generality. Under the GP assumption, the function value at a query point $\boldsymbol{z}^* \in \mathbb{R}^{n_z}$ and the observed targets, stacked into a vector $\boldsymbol{y} = \left[y^{(1)}, \ldots, y^{(N)}\right]^\top$, have the joint probability distribution

$$\begin{bmatrix} \boldsymbol{y} \\ g(\boldsymbol{z}^*) \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \bar{\boldsymbol{K}} & \boldsymbol{k}(\boldsymbol{z}^*)^\top \\ \boldsymbol{k}(\boldsymbol{z}^*) & k(\boldsymbol{z}^*, \boldsymbol{z}^*) \end{bmatrix}\right) \tag{2.26}$$

where the gram matrix $\bar{\boldsymbol{K}} \in \mathbb{R}^{N \times N}$ and vector $\boldsymbol{k}(\boldsymbol{z}^*) \in \mathbb{R}^N$ are defined as $\bar{\boldsymbol{K}} = \boldsymbol{K} + \sigma_\mathrm{n}^2 \boldsymbol{I}_n$, where $K_{ij} = k(\boldsymbol{z}^{(i)}, \boldsymbol{z}^{(j)})$, and $\boldsymbol{k}(\boldsymbol{z}^*) = \left[k(\boldsymbol{z}^{(1)}, \boldsymbol{z}^*), \ldots, k(\boldsymbol{z}^{(N)}, \boldsymbol{z}^*)\right]^\top$, respectively. By conditioning $g(\boldsymbol{z}^*)$ on the training data, we obtain the posterior predictive distribution $g(\boldsymbol{z}^*) \,|\, \mathcal{D} \sim \mathcal{N}(\mu(\boldsymbol{z}^*), \sigma^2(\boldsymbol{z}^*))$ with the predictive mean and variance

$$\mu(\boldsymbol{z}^*) = \boldsymbol{k}(\boldsymbol{z}^*)^\top \bar{\boldsymbol{K}}^{-1} \boldsymbol{y} \tag{2.27}$$

$$\sigma^2(\boldsymbol{z}^*) = k(\boldsymbol{z}^*, \boldsymbol{z}^*) - \boldsymbol{k}(\boldsymbol{z}^*)^\top \bar{\boldsymbol{K}}^{-1} \boldsymbol{k}(\boldsymbol{z}^*). \tag{2.28}$$

The computational complexity of evaluating (2.27) and (2.28) is $\mathcal{O}(N^3)$ due to the inversion of the $N \times N$ matrix $\bar{\boldsymbol{K}}$. However, in practice, the inverse $\bar{\boldsymbol{K}}^{-1}$ is often precomputed, reducing the complexity of the mean computation to $\mathcal{O}(N)$ and of the variance computation to $\mathcal{O}(N^2)$ [13].

The kernel function is usually parametrized by a vector of hyperparameters $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^{n_\theta}$, but we omit this dependence in the notation for brevity. A popular choice of the kernel is the squared-exponential (SE) function

$$k(\boldsymbol{z}, \boldsymbol{z}') = \sigma_f^2 \exp\left(-\frac{1}{2}\left(\boldsymbol{z} - \boldsymbol{z}'\right)^\top \boldsymbol{L}^{-2}\left(\boldsymbol{z} - \boldsymbol{z}'\right)\right), \tag{2.29}$$

where $\sigma_f^2 > 0$ is the output variance, and $\boldsymbol{L} = \mathrm{Diag}(\boldsymbol{l}) \in \mathbb{R}^{n_z \times n_z}$ contains the vector of length scales $\boldsymbol{l} = [l_1, \ldots, l_{n_z}]^\top$, which correspond to the rate of change or sensitivity of g w.r.t. its input dimensions. The choice (2.29) is not restrictive in practice since the corresponding space of sample functions of the GP contains all continuous functions [30]. Other commonly used kernel functions include the Matérn kernel or the rotational quadratic kernel [13]. A detailed discussion about the kernel selection and its implications can be found in [31].

The true hyperparameters of the kernel are typically unknown in practice. The most common approach for obtaining suitable values is through maximization of the marginal log-likelihood of the training data (2.9), which can be achieved by solving

$$\begin{aligned}\boldsymbol{\theta}^* &= \arg\max_{\boldsymbol{\theta}\in\Theta} \log p\left(y^{(1)},\dots,y^{(N)} \,\middle|\, \boldsymbol{z}^{(1)},\dots,\boldsymbol{z}^{(N)},\boldsymbol{\theta}\right)\\ &= \arg\max_{\boldsymbol{\theta}\in\Theta} \log \mathcal{N}\left(\mathbf{0},\bar{\boldsymbol{K}}\right)\\ &= \arg\max_{\boldsymbol{\theta}\in\Theta} -\frac{1}{2}\boldsymbol{y}^\top\bar{\boldsymbol{K}}^{-1}\boldsymbol{y} - \frac{1}{2}\log\det\bar{\boldsymbol{K}}. \end{aligned} \tag{2.30}$$

A (potentially suboptimal) solution to this nonlinear optimization problem can be computed using gradient ascent approaches, for instance, based on the Broyden–Fletcher–Goldfarb–Shanno (BFGS) method [32].

In addition to the function value, predicting the derivative of g at a given input $\boldsymbol{z}^*$ can be of interest. Since the derivative is a linear operator, the derivative of a GP is also a GP [13]. This property allows the inclusion of derivative observations into the training data but it can also be used to compute derivative predictions. From (2.26), we obtain

$$\begin{bmatrix}\boldsymbol{y}\\ \frac{\partial g(\boldsymbol{z})}{\partial \boldsymbol{z}}\Big|_{\boldsymbol{z}=\boldsymbol{z}^*}\end{bmatrix} \sim \mathcal{N}\left(\mathbf{0},\begin{bmatrix}\boldsymbol{K}+\sigma_{\mathrm{n}}^2\boldsymbol{I} & \frac{\partial \boldsymbol{k}(\boldsymbol{z})}{\partial \boldsymbol{z}}\Big|_{\boldsymbol{z}=\boldsymbol{z}^*}\\ \frac{\partial \boldsymbol{k}(\boldsymbol{z})^\top}{\partial \boldsymbol{z}}\Big|_{\boldsymbol{z}=\boldsymbol{z}^*} & \frac{\partial^2 k(\boldsymbol{z},\boldsymbol{z})}{\partial\boldsymbol{z}\partial\boldsymbol{z}}\Big|_{\boldsymbol{z}=\boldsymbol{z}^*}\end{bmatrix}\right). \tag{2.31}$$

Similar to the prediction of the function value at $\boldsymbol{z}^*$, predictions of the derivatives can be obtained by conditioning the derivative of g at $\boldsymbol{z}^*$ on the training data, which yields $\frac{\partial g(\boldsymbol{z})}{\partial \boldsymbol{z}}\Big|_{\boldsymbol{z}=\boldsymbol{z}^*} \,\Big|\, \mathcal{D} \sim \mathcal{N}(\boldsymbol{\mu}'(\boldsymbol{z}^*),\, \boldsymbol{\Sigma}'(\boldsymbol{z}^*))$ with the derivative mean and variance,

$$\boldsymbol{\mu}'(\boldsymbol{z}^*) = \frac{\partial \boldsymbol{k}(\boldsymbol{z})}{\partial \boldsymbol{z}}\bigg|_{\boldsymbol{z}=\boldsymbol{z}^*}\bar{\boldsymbol{K}}^{-1}\boldsymbol{y} \tag{2.32}$$

$$\boldsymbol{\Sigma}'(\boldsymbol{z}^*) = \frac{\partial^2 k(\boldsymbol{z}^*,\boldsymbol{z}^*)}{\partial\boldsymbol{z}\partial\boldsymbol{z}}\bigg|_{\boldsymbol{z}=\boldsymbol{z}^*} - \frac{\partial \boldsymbol{k}(\boldsymbol{z})}{\partial \boldsymbol{z}}\bigg|_{\boldsymbol{z}=\boldsymbol{z}^*}\bar{\boldsymbol{K}}^{-1}\left(\frac{\partial \boldsymbol{k}(\boldsymbol{z})}{\partial \boldsymbol{z}}\bigg|_{\boldsymbol{z}=\boldsymbol{z}^*}\right)^\top. \tag{2.33}$$

For the SE kernel (2.29), the partial derivative terms in (2.31) - (2.33) are straightforward to calculate and given by

$$\frac{\partial \boldsymbol{k}(\boldsymbol{z})}{\partial \boldsymbol{z}}\bigg|_{\boldsymbol{z}=\boldsymbol{z}^*} = \boldsymbol{L}^{-2}\left[\left(\boldsymbol{z}^{(1)}-\boldsymbol{z}^*\right)k\left(\boldsymbol{z}^*,\,\boldsymbol{z}^{(1)}\right) \;\dots\; \left(\boldsymbol{z}^{(N)}-\boldsymbol{z}^*\right)k\left(\boldsymbol{z}^*,\,\boldsymbol{z}^{(N)}\right)\right] \tag{2.34}$$

$$\frac{\partial^2 k(\boldsymbol{z},\boldsymbol{z})}{\partial\boldsymbol{z}\partial\boldsymbol{z}}\bigg|_{\boldsymbol{z}=\boldsymbol{z}^*} = \sigma_f^2\boldsymbol{L}^{-2}. \tag{2.35}$$

We conclude this section with a simple example illustrating the use of GPs for learning an unknown function and its derivative.

Example 2.3 (Gaussian process regression)**.** *We consider learning the same unknown*

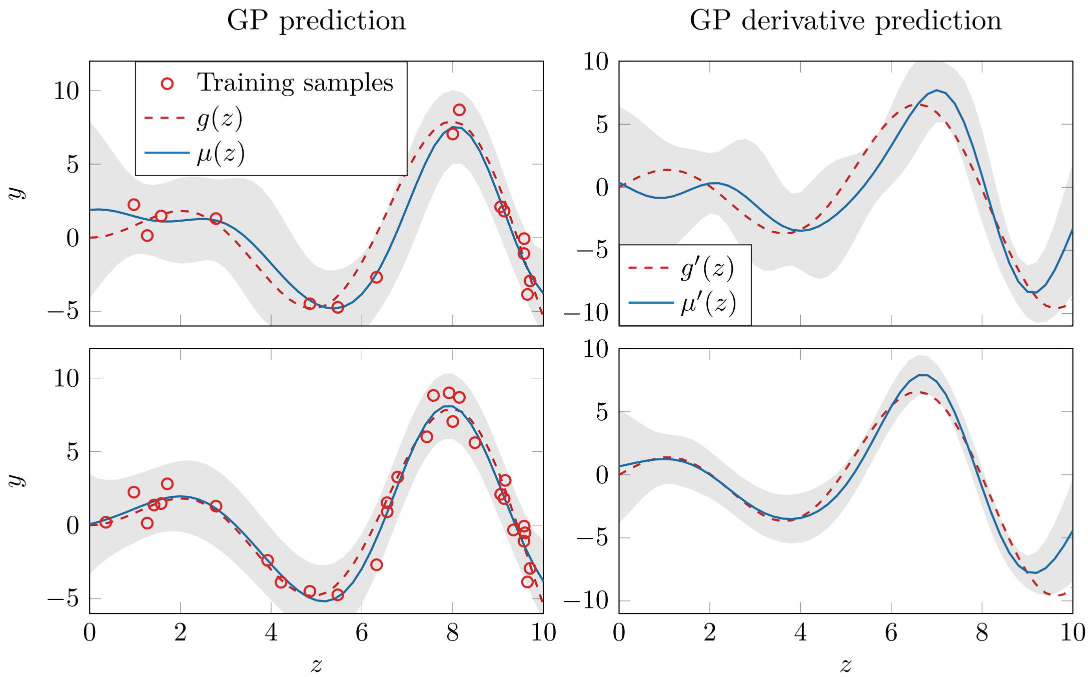

Figure 2.3: Gaussian process (GP) regression for learning the unknown function $g(z) = z\sin(z)$ and its derivative from $N = 15$ (top) and $N = 30$ (bottom) training samples. The shaded regions correspond to $\pm$ two standard deviations.

function $g(z) = z\sin(z)$ as in Example 2.2, whose derivative function is given by $g'(z) = \sin(z) + z\cos(z)$. We employ the SE kernel, set the variance of the observation noise to $\sigma_\mathrm{n}^2 = 0.5^2$, and draw N training inputs randomly from a uniform distribution over the set $[0, 10]$. Figure 2.3 shows the GP prediction (left) and the GP derivative prediction (right) for $N = 15$ and $N = 30$ training samples, respectively. It can be seen that increasing the amount of training data reduces the uncertainty in the predictions. Moreover, in contrast to BLR, the model uncertainty is highest for inputs with a large distance (as measured by the kernel function) from the training data. Compared to BLR in Example 2.2, which makes more use of prior knowledge, more data is required to obtain a good approximation of $g(\cdot)$ despite the observation noise standard deviation being only half as large.

2.3 Sampled-Data Systems

Any real-world system evolves continuously, but a digital controller updates the control input only at discrete points in time. This fact is typically neglected, and either fully continuous or discrete-time models and controllers are used. However, a subfield within control theory deals with combining a continuous-time process with a discrete-time controller, called sampled-data systems. In this section, we introduce sampled-data systems, focusing on their stability analysis.

2.3.1 Introduction

Consider a continuous-time linear time-invariant (LTI) system with state $\boldsymbol{x}(t) \in \mathbb{R}^n$ and input $\boldsymbol{u}(t) \in \mathbb{R}^m$ at time $t \in \mathbb{R}_{\geq 0}$. The system evolves according to the dynamics

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t), \tag{2.36}$$

where $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{B} \in \mathbb{R}^{n\times m}$ are referred to as dynamics matrix and input matrix, respectively. The input is only updated at discrete points in time $t_k \in \mathbb{R}_0^+$, $k \in \mathbb{N}_0$, called sampling instants, where $t_0 = 0$. We consider a linear control law

$$\boldsymbol{u}(t) = \boldsymbol{K}\boldsymbol{x}(t_k), \qquad \forall t \in [t_k, t_{k+1}), \tag{2.37}$$

where $\boldsymbol{K} \in \mathbb{R}^{m\times n}$ is the state feedback matrix. We make the following assumption about the time interval between the sampling instants.

Assumption 2.2. *The time between two consecutive samples satisfies $t_{k+1} - t_k \leq T_\mathrm{s}$, $\forall k \in \mathbb{N}_0$, where $T_\mathrm{s} > 0$ is an upper bound on the sampling interval.*

Clearly, Assumption 2.2 captures common periodic sampling, i.e., $t_{k+1} - t_k = T_\mathrm{s}$, $\forall k \in \mathbb{N}_0$, as well as aperiodic sampling. Inserting the controller (2.37) into the dynamics (2.36) gives the continuous-time closed-loop system

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{K}\boldsymbol{x}(t_k), \qquad \forall t \in [t_k, t_{k+1}), \quad k \in \mathbb{N}_0. \tag{2.38}$$

It is shown in [19] that if the continuous-time feedback $\boldsymbol{u}(t) = \boldsymbol{K}\boldsymbol{x}(t)$ stabilizes the system (2.36), then the discrete-time feedback (2.37) stabilizes (2.36) for all small enough T_s. The most important question in designing a sampled-data controller is about the conditions the control gain $\boldsymbol{K}$ and the upper bound on the sampling interval T_s have to satisfy such that the system (2.38) is guaranteed to be stable.

The stability of system (2.38) can be analyzed from different perspectives [17, 9], for example, via a time-delay modeling of the sampled-data system or by considering it a hybrid system [33]. In this thesis, we focus on the time-delay approach [18] as it allows us to eventually derive robust stability conditions for the uncertainty in a learned model; see Section 4.2. The underlying idea of the time-delay approach is to rewrite (2.38) as an LTI system with a time-varying delay

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{K}\boldsymbol{x}(t - \tau(t)), \tag{2.39}$$

where the delay function

$$\tau(t) = t - t_k, \qquad \forall t \in [t_k, t_{k+1}), \quad k \in \mathbb{N}_0, \tag{2.40}$$

is piecewise-continuous and satisfies $\dot{\tau}(t) = 1$ almost everywhere. As a result of Assump-

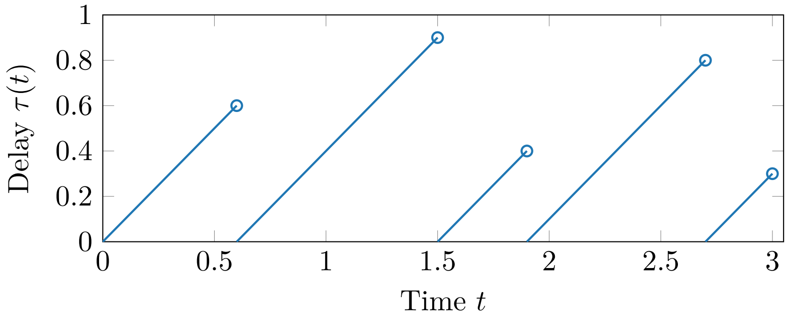

Figure 2.4: Piecewise-continuous time delay introduced by employing a discrete-time control law for a continuous-time system.

tion 2.2, the delay is bounded by $\tau(t) \in [0, T_\mathrm{s}]$, $\forall t \in \mathbb{R}_{\geq 0}$.

Example 2.4. *Consider the sampling sequence* $\{t_k\}_{k=1}^{5} = \{0.6, 1.5, 1.9, 2.7, 3\}$. *The corresponding evolution of the delay* $\tau(t)$ *is shown in Fig. 2.4 and has a sawtooth shape.*

2.3.2 Stability Analysis

It is well known that delays in the feedback loop can have a destabilizing effect. Hence, $\boldsymbol{A}+\boldsymbol{BK}$ being Hurwitz is generally insufficient to guarantee stability of the system (2.39); see [18]. However, deriving sufficient conditions for the stability of (2.39) is non-trivial. In [19], the authors address this problem utilizing a transformation method, which has been proposed in [34] in the context of delay differential-algebraic equations. It can be used to write the system (2.39) in the equivalent descriptor form

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{y}(t) \tag{2.41a}$$

$$\boldsymbol{y}(t) = \begin{cases} \boldsymbol{Ax}(t) + \boldsymbol{BKx}(t-\tau(t)), & \text{if } t \in [0,\, T_\mathrm{s}) \\ (\boldsymbol{A}+\boldsymbol{BK})\boldsymbol{x}(t) - \boldsymbol{BK}\int_{t-\tau(t)}^{t} \boldsymbol{y}(s)\mathrm{d}s, & \text{if } t \geq T_\mathrm{s}. \end{cases} \tag{2.41b}$$

Given an initial condition $\boldsymbol{x}(t) = \boldsymbol{\phi}(t)$, $t \in [-T_\mathrm{s}, 0]$, where $\boldsymbol{\phi}$ is a continuous function, $\boldsymbol{x}(t)$ satisfies the original dynamics (2.39) for $t \in \mathbb{R}_{\geq 0}$ if and only if it satisfies (2.41). The first equivalence in (2.41b) for $t \in [0, T_\mathrm{s})$ directly follows from (2.39), and the second equivalence can be shown by rewriting (2.39) as

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{Ax}(t) + \boldsymbol{BKx}(t-\tau(t)) + \boldsymbol{BKx}(t) - \boldsymbol{BKx}(t) \\ &= (\boldsymbol{A} + \boldsymbol{BKx}(t)) - \boldsymbol{BK}(\boldsymbol{x}(t) - \boldsymbol{x}(t-\tau(t))) \\ &= (\boldsymbol{A} + \boldsymbol{BKx}(t)) - \boldsymbol{BK}\int_{t-\tau(t)}^{t} \dot{\boldsymbol{x}}(s)\mathrm{d}s. \end{aligned}$$

The stability of the descriptor system (2.41) can be analyzed using the Lyapunov-

Krasovskii functional (LKF) [19]

$$V(t) = V_1(t) + V_2(t), \tag{2.42}$$

where

$$V_1(t) = \boldsymbol{x}(t)^\top \boldsymbol{P}_1 \boldsymbol{x}(t) \tag{2.43a}$$
$$V_2(t) = \int_{-T_\mathrm{s}}^{0} \int_{t+\theta}^{t} \boldsymbol{y}(s)^\top \boldsymbol{R} \boldsymbol{y}(s) \mathrm{d}s \mathrm{d}\theta, \tag{2.43b}$$

where $\boldsymbol{P}_1 \succ \boldsymbol{0}$ and $\boldsymbol{R} \succ \boldsymbol{0}$. While the term $V_1(t)$ is common in Lyapunov-based stability analysis [35], the choice of the second term $V_2(t)$ may not be as obvious and will be justified later. We define the augmented state $\bar{\boldsymbol{x}}(t) = (\boldsymbol{x}(t), \boldsymbol{y}(t)) \in \mathbb{R}^{2n}$ of the descriptor system (2.41). It can be shown that the LKF (2.42) satisfies

$$a\|\boldsymbol{x}(t)\|_2^2 \leq V(t) \leq b \sup_{s\in[-T_\mathrm{s},0]} \|\bar{\boldsymbol{x}}(t+s)\|_2^2 \tag{2.44}$$

for some $a > 0$, $b > 0$, and is therefore a valid Lyapunov functional candidate [19]. To prove asymptotic stability, it must be shown that $\dot{V}(t) \leq 0$ with equality holding only for $\bar{\boldsymbol{x}}(t) = \boldsymbol{0}$. Computing the derivative of the first LKF term (2.43a) and inserting (2.41) yields

$$\begin{aligned}
\dot{V}_1(t) &= \dot{\boldsymbol{x}}(t)^\top \boldsymbol{P}_1 \boldsymbol{x}(t) + \boldsymbol{x}(t)^\top \boldsymbol{P}_1 \dot{\boldsymbol{x}}(t) \\
&= \boldsymbol{x}(t)^\top ((\boldsymbol{A}+\boldsymbol{B}\boldsymbol{K})^\top \boldsymbol{P}_1 + \boldsymbol{P}_1(\boldsymbol{A}+\boldsymbol{B}\boldsymbol{K}))\boldsymbol{x}(t) - 2\boldsymbol{x}(t)^\top \boldsymbol{P}_1 \boldsymbol{B}\boldsymbol{K} \int_{t-\tau(t)}^{t} \boldsymbol{y}(s)\mathrm{d}s \\
&= \bar{\boldsymbol{x}}(t)^\top \left(\boldsymbol{P}^\top \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I} \\ \boldsymbol{A}+\boldsymbol{B}\boldsymbol{K} & -\boldsymbol{I} \end{bmatrix} + \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I} \\ \boldsymbol{A}+\boldsymbol{B}\boldsymbol{K} & -\boldsymbol{I} \end{bmatrix}^\top \boldsymbol{P} \right) \bar{\boldsymbol{x}}(t) \\
&\qquad - 2\bar{\boldsymbol{x}}(t)^\top \boldsymbol{P}^\top \begin{bmatrix} \boldsymbol{B}\boldsymbol{K} \\ \boldsymbol{0} \end{bmatrix} \int_{t-\tau(t)}^{t} \boldsymbol{y}(s)\mathrm{d}s,
\end{aligned} \tag{2.45}$$

where

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{P}_1 & \boldsymbol{0} \\ \boldsymbol{P}_2 & \boldsymbol{P}_3 \end{bmatrix}, \qquad \boldsymbol{P}_2 \in \mathbb{R}^{n\times n}, \quad \boldsymbol{P}_3 \in \mathbb{R}^{n\times n}. \tag{2.46}$$

The second term in (2.45) satisfies

$$-2\bar{\boldsymbol{x}}(t)^\top \boldsymbol{P}^\top \begin{bmatrix} \boldsymbol{B}\boldsymbol{K} \\ \boldsymbol{0} \end{bmatrix} \int_{t-\tau(t)}^{t} \boldsymbol{y}(s)\mathrm{d}s \leq \bar{\boldsymbol{x}}(t)^\top T_\mathrm{s} \boldsymbol{Z} \bar{\boldsymbol{x}}(t) + \int_{t-\tau(t)}^{t} \boldsymbol{y}(s)^\top \boldsymbol{R} \boldsymbol{y}(s)\mathrm{d}s, \tag{2.47}$$

where

$$\boldsymbol{Z} = \begin{bmatrix} \boldsymbol{Z}_1 & \boldsymbol{0} \\ \boldsymbol{Z}_2 & \boldsymbol{Z}_3 \end{bmatrix}, \qquad \boldsymbol{Z}_1 = \boldsymbol{Z}_1^\top \in \mathbb{R}^{n \times n}, \quad \boldsymbol{Z}_2 \in \mathbb{R}^{n \times n}, \quad \boldsymbol{Z}_3 = \boldsymbol{Z}_3^\top \in \mathbb{R}^{n \times n}, \tag{2.48}$$

if the matrix inequality

$$\begin{bmatrix} \boldsymbol{R} & \begin{bmatrix} \boldsymbol{K}^\top \boldsymbol{B}^\top & \boldsymbol{0} \end{bmatrix} \boldsymbol{P} \\ * & \boldsymbol{Z} \end{bmatrix} \succeq \boldsymbol{0} \tag{2.49}$$

holds [19]. To see this, note that

$$-2\bar{\boldsymbol{x}}(t)^\top \boldsymbol{P}^\top \begin{bmatrix} \boldsymbol{BK} \\ \boldsymbol{0} \end{bmatrix} \boldsymbol{y}(s) = \begin{bmatrix} \boldsymbol{y}(s) \\ \bar{\boldsymbol{x}}(t) \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{0} & -\begin{bmatrix} \boldsymbol{K}^\top \boldsymbol{B}^\top & \boldsymbol{0} \end{bmatrix} \boldsymbol{P} \\ * & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{y}(s) \\ \bar{\boldsymbol{x}}(t) \end{bmatrix}$$

and, as a result of (2.49), we have

$$\begin{aligned}
&\begin{bmatrix} \boldsymbol{y}(s) \\ \bar{\boldsymbol{x}}(t) \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{0} & -\begin{bmatrix} \boldsymbol{K}^\top \boldsymbol{B}^\top & \boldsymbol{0} \end{bmatrix} \boldsymbol{P} \\ * & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{y}(s) \\ \bar{\boldsymbol{x}}(t) \end{bmatrix} \\
&\le \begin{bmatrix} \boldsymbol{y}(s) \\ \bar{\boldsymbol{x}}(t) \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{0} & -\begin{bmatrix} \boldsymbol{K}^\top \boldsymbol{B}^\top & \boldsymbol{0} \end{bmatrix} \boldsymbol{P} \\ * & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{y}(s) \\ \bar{\boldsymbol{x}}(t) \end{bmatrix} + \underbrace{\begin{bmatrix} \boldsymbol{y}(s) \\ \bar{\boldsymbol{x}}(t) \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{R} & \begin{bmatrix} \boldsymbol{K}^\top \boldsymbol{B}^\top & \boldsymbol{0} \end{bmatrix} \boldsymbol{P} \\ * & \boldsymbol{Z} \end{bmatrix} \begin{bmatrix} \boldsymbol{y}(s) \\ \bar{\boldsymbol{x}}(t) \end{bmatrix}}_{\ge 0} \\
&= \begin{bmatrix} \boldsymbol{y}(s) \\ \bar{\boldsymbol{x}}(t) \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{R} & \boldsymbol{0} \\ * & \boldsymbol{Z} \end{bmatrix} \begin{bmatrix} \boldsymbol{y}(s) \\ \bar{\boldsymbol{x}}(t) \end{bmatrix} \\
&= \boldsymbol{y}(s)^\top \boldsymbol{R} \boldsymbol{y}(s) + \bar{\boldsymbol{x}}(t)^\top \boldsymbol{Z} \bar{\boldsymbol{x}}(t).
\end{aligned}$$

Then, (2.47) directly follows from the fact that the delay is bounded by $\tau(t) \in [0, T_\mathrm{s}]$ due to Assumption 2.2. Computing the derivative of the second LKF term (2.43b) gives

$$\begin{aligned}
\dot{V}_2(t) &= T_\mathrm{s} \boldsymbol{y}(t)^\top \boldsymbol{R} \boldsymbol{y}(t) - \int_{t-T_\mathrm{s}}^{t} \boldsymbol{y}(s)^\top \boldsymbol{R} \boldsymbol{y}(s) \mathrm{d}s \qquad (2.50) \\
&= \bar{\boldsymbol{x}}(t)^\top \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & T_\mathrm{s} \boldsymbol{R} \end{bmatrix} \bar{\boldsymbol{x}}(t) - \int_{t-T_\mathrm{s}}^{t} \boldsymbol{y}(s)^\top \boldsymbol{R} \boldsymbol{y}(s) \mathrm{d}s \qquad (2.51) \\
&\le \bar{\boldsymbol{x}}(t)^\top \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & T_\mathrm{s} \boldsymbol{R} \end{bmatrix} \bar{\boldsymbol{x}}(t) - \int_{t-\tau(t)}^{t} \boldsymbol{y}(s)^\top \boldsymbol{R} \boldsymbol{y}(s) \mathrm{d}s, \qquad (2.52)
\end{aligned}$$

where the inequality holds due to $\tau(t) \in [0, T_\mathrm{s}]$. Adding (2.45) and (2.52) and making use of the inequality (2.47), we obtain

$$\dot{V}(t) \le \bar{\boldsymbol{x}}(t)^\top \boldsymbol{\Psi} \bar{\boldsymbol{x}}(t), \tag{2.53}$$

where

$$\boldsymbol{\Psi} = \boldsymbol{P}^\top \begin{bmatrix} \mathbf{0} & \boldsymbol{I} \\ \boldsymbol{A}+\boldsymbol{B}\boldsymbol{K} & -\boldsymbol{I} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \boldsymbol{I} \\ \boldsymbol{A}+\boldsymbol{B}\boldsymbol{K} & -\boldsymbol{I} \end{bmatrix}^\top \boldsymbol{P} + T_\mathrm{s}\boldsymbol{Z} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ * & T_\mathrm{s}\boldsymbol{R} \end{bmatrix}, \tag{2.54}$$

provided (2.49) holds. The above derivation illustrates that $V_2(t)$ is chosen in order to compensate for the delay-dependent term in $\dot{V}_1(t)$. Clearly, $\dot{V}(t) < 0$ for all $\bar{\boldsymbol{x}}(t) \neq \mathbf{0}$ if

$$\boldsymbol{\Psi} \prec \mathbf{0}, \tag{2.55}$$

which leads to the following sufficient conditions for asymptotic stability.

Lemma 2.4 ([19], Lemma 2.1)**.** *Given a gain matrix $\boldsymbol{K}$, the sampled-data system (2.39) is asymptotically stable for all samplings satisfying Assumption 2.2 if there exist matrices $\boldsymbol{P}_1 = \boldsymbol{P}_1^\top \succ \mathbf{0}$, $\boldsymbol{P}_2$, $\boldsymbol{P}_3$, $\boldsymbol{Z}_1$, $\boldsymbol{Z}_2$, $\boldsymbol{Z}_3$ and $\boldsymbol{R} = \boldsymbol{R}^\top \succ \mathbf{0}$, all in $\mathbb{R}^{n\times n}$, that satisfy the matrix inequalities (2.55) and (2.49), where $\boldsymbol{P}$ and $\boldsymbol{Z}$ are defined in (2.46) and (2.48), respectively.*

Lemma 2.4 can be used to analyze the stability of (2.39) for a *given* control gain $\boldsymbol{K}$ and a *given* sampling time T_s. However, since the matrix inequalities (2.55) and (2.49) are bilinear in $\boldsymbol{P}$ and $\boldsymbol{K}$, Lemma 2.4 is unsuitable for designing a stabilizing controller. This problem can be addressed by reformulating (2.55) and (2.49). For this purpose, we define

$$\boldsymbol{P}^{-1} = \boldsymbol{Q} = \begin{bmatrix} \boldsymbol{Q}_1 & \mathbf{0} \\ \boldsymbol{Q}_2 & \boldsymbol{Q}_3 \end{bmatrix}, \qquad \boldsymbol{\Delta} = \begin{bmatrix} \boldsymbol{Q} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{I} \end{bmatrix},$$

$\bar{\boldsymbol{R}} = \boldsymbol{R}^{-1}$, $\bar{\boldsymbol{Z}} = \boldsymbol{Q}^\top\boldsymbol{Z}\boldsymbol{Q}$ and $\boldsymbol{Y} = \boldsymbol{K}\boldsymbol{Q}_1$. Applying the Schur complement stated in Lemma 2.3 to the term $T_\mathrm{s}\boldsymbol{R}$ in Lemma 2.3 yields

$$\begin{aligned} &\boldsymbol{P}^\top \begin{bmatrix} \mathbf{0} & \boldsymbol{I} \\ \boldsymbol{A}+\boldsymbol{B}\boldsymbol{K} & -\boldsymbol{I} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \boldsymbol{I} \\ \boldsymbol{A}+\boldsymbol{B}\boldsymbol{K} & -\boldsymbol{I} \end{bmatrix}^\top \boldsymbol{P} + T_\mathrm{s}\boldsymbol{Z} + \begin{bmatrix} \mathbf{0} & \boldsymbol{I} \end{bmatrix} T_\mathrm{s}\boldsymbol{R} \begin{bmatrix} \mathbf{0} \\ \boldsymbol{I} \end{bmatrix} \prec \mathbf{0} \\ \Leftrightarrow &\begin{bmatrix} \boldsymbol{P}^\top \begin{bmatrix} \mathbf{0} & \boldsymbol{I} \\ \boldsymbol{A}+\boldsymbol{B}\boldsymbol{K} & -\boldsymbol{I} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \boldsymbol{I} \\ \boldsymbol{A}+\boldsymbol{B}\boldsymbol{K} & -\boldsymbol{I} \end{bmatrix}^\top \boldsymbol{P} + T_\mathrm{s}\boldsymbol{Z} & \begin{matrix} \mathbf{0} \\ T_\mathrm{s}\boldsymbol{I} \end{matrix} \\ * & -T_\mathrm{s}\bar{\boldsymbol{R}} \end{bmatrix} \prec \mathbf{0}. \end{aligned} \tag{2.56}$$

Finally, multiplying (2.56) and (2.49) by $\boldsymbol{\Delta}^\top$ and $\boldsymbol{\Delta}$ on the left and the right, respectively, yields the following constructive stability conditions.

Lemma 2.5 ([19], Lemma 2.3)**.** *The control law (2.37) asymptotically stabilizes the system (2.36) for all samplings satisfying Assumption 2.2 if there exist matrices $\boldsymbol{Q}_1 = \boldsymbol{Q}_1^\top \succ \mathbf{0}$, $\boldsymbol{Q}_2$, $\boldsymbol{Q}_3$, $\boldsymbol{Z}_1$, $\boldsymbol{Z}_2$, $\boldsymbol{Z}_3$, $\boldsymbol{R} = \boldsymbol{R}^\top \succ \mathbf{0}$, all in $\mathbb{R}^{n\times n}$, and $\boldsymbol{Y} \in \mathbb{R}^{m\times n}$, that* *satisfy the matrix inequalities*

$$\begin{bmatrix} \boldsymbol{Q}_2 + \boldsymbol{Q}_2^\top + T_\mathrm{s}\bar{\boldsymbol{Z}}_1 & \boldsymbol{Q}_3 - \boldsymbol{Q}_2^\top + \boldsymbol{Q}_1\boldsymbol{A}^\top + T_\mathrm{s}\bar{\boldsymbol{Z}}_2 + \boldsymbol{Y}^\top\boldsymbol{B}^\top & T_\mathrm{s}\boldsymbol{Q}_2^\top \\ * & -\boldsymbol{Q}_3 - \boldsymbol{Q}_3^\top + T_\mathrm{s}\bar{\boldsymbol{Z}}_3 & T_\mathrm{s}\boldsymbol{Q}_3^\top \\ * & * & -T_\mathrm{s}\bar{\boldsymbol{R}} \end{bmatrix} \prec \boldsymbol{0} \tag{2.57}$$

$$\begin{bmatrix} \boldsymbol{Q}_1\bar{\boldsymbol{R}}^{-1}\boldsymbol{Q}_1 & \boldsymbol{0} & \boldsymbol{Y}^\top\boldsymbol{B}^\top \\ * & \bar{\boldsymbol{Z}}_1 & \bar{\boldsymbol{Z}}_2 \\ * & * & \bar{\boldsymbol{Z}}_3 \end{bmatrix} \succeq \boldsymbol{0}. \tag{2.58}$$

The stabilizing state-feedback gain is then given by $\boldsymbol{K} = \boldsymbol{Y}\boldsymbol{Q}_1^{-1}$.

Remark 2.2 (Conservativeness)**.** *In [20, 21], less conservative stability conditions than those in Lemma 2.5 are derived using a time-dependent LKF. However, these conditions are computationally more expensive to evaluate and do not allow for an extension to norm-bounded uncertainty, as performed for the conditions in Lemma 2.5 in Chapter 4.*

Remark 2.3 (Polytopic uncertainty)**.** *Let* $\boldsymbol{A}$ *and* $\boldsymbol{B}$ *be unknown but contained in a polytopic set with* M *vertices. This can be expressed as*

$$[\boldsymbol{A},\, \boldsymbol{B}] \in \mathcal{C} := \operatorname{conv}\left(\left\{\left[\boldsymbol{A}^{(1)}\ \boldsymbol{B}^{(1)}\right], \dots, \left[\boldsymbol{A}^{(M)}\ \boldsymbol{B}^{(M)}\right]\right\}\right), \tag{2.59}$$

where $\operatorname{conv}(\mathcal{X}) \subset \mathbb{R}^n$ *denotes the convex hull of the set* $\mathcal{X} \subset \mathbb{R}^n$*. Then, robust stability of* (2.39) *is guaranteed if* (2.57) *and* (2.58) *hold for all vertices of the polytope* $\mathcal{C}$*.*

We provide a simple example illustrating the relationship between the maximum admissible sampling time and polytopic-type model uncertainty.

Example 2.5 (Sampled-data system with polytopic-type uncertainty)**.** *Consider the LTI system*

$$\dot{\boldsymbol{x}}(t) = \begin{bmatrix} 1 & 0.5 \\ \theta_1 & -1 \end{bmatrix} \boldsymbol{x}(t) + \begin{bmatrix} 1+\theta_2 \\ -1 \end{bmatrix} \boldsymbol{u}(t), \tag{2.60}$$

where θ_1 *and* θ_2 *are uncertain, bounded parameters. In this case, the uncertainty polytope in* (2.59) *has* $M = 2^2 = 4$ *vertices. If the parameters are bounded by* $|\theta_1| \le 0.2$, $|\theta_2| \le 0.15$*, the LMIs* (2.57) *and* (2.58) *are feasible on all 4 vertices for all* $T_\mathrm{s} \le T_\mathrm{s,max} = 0.257$*. If the magnitude of the uncertainty increases to* $|\theta_1| \le 0.3$, $|\theta_2| \le 0.2$*, we obtain* $T_\mathrm{s,max} = 0.201$*. This shows that if the uncertainty about the system's dynamic behavior is larger, the sampling time must be reduced to guarantee stability, i.e., feedback has to be applied more frequently.*

This simple example illustrates a key benefit of the sampled-data system description: It allows us to relate the uncertainty in a continuous-time dynamics model to the minimum frequency at which a discrete-time controller must recalculate the input to guarantee robust stability.

Chapter 3

Probabilistic Model Learning for Sampled-Data Control

In this chapter, we discuss learning the linearized dynamics of an unknown system from noisy observations. We employ the probabilistic learning methods introduced in Section 2.2 and derive confidence intervals for the uncertainty associated with the estimated model.

3.1 Problem Statement

Consider a dynamical system whose state and input at time $t \in \mathbb{R}_{\geq 0}$ are given by $\boldsymbol{x}(t) \in \mathbb{R}^n$ and $\boldsymbol{u}(t) \in \mathbb{R}^m$, respectively. The system evolves according to the continuous-time dynamics

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \tag{3.1}$$

where $\boldsymbol{h} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is the dynamics function, which we assume to be unknown. For many physical systems, such as robots, an approximate dynamics model is available, for example, derived from first principles. To take such prior knowledge into account, we make the following assumption.

Assumption 3.1. *The unknown function $\boldsymbol{h}$ in (3.1) is the sum of a known function $\boldsymbol{f}(\cdot) = [f_1(\cdot), \ldots, f_n(\cdot)]^\top$ and an unknown function $\boldsymbol{g}(\cdot) = [g_1(\cdot), \ldots, g_n(\cdot)]^\top$, i.e., $\boldsymbol{h} = \boldsymbol{f} + \boldsymbol{g}$, where $\boldsymbol{f}$ and $\boldsymbol{g}$ are continuously differentiable.*

The unknown component $\boldsymbol{g}$ of the dynamics function accounts for unmodeled dynamic effects, such as air drag or friction, which are often difficult to model accurately [7]. As a consequence of Assumption 3.1, we can decompose the dynamics (3.1) into

$$\dot{\boldsymbol{x}}(t) = \underbrace{\boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t))}_{\text{known}} + \underbrace{\boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t))}_{\text{unknown}}. \tag{3.2}$$

In the following, we stack the state and input compactly into the vector $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{u}) \in \mathbb{R}^{n_z}$, where $n_z = n + m$.

We aim to learn the residual dynamics $\boldsymbol{g}$ from data. For this, as formalized in the following, we assume the availability of measured data collected from system (3.2).

Assumption 3.2. *A data set of noise-free training inputs and noisy targets*

$$\mathcal{D} = \left\{ \boldsymbol{z}^{(i)}, \, \boldsymbol{y}^{(i)} = \dot{\boldsymbol{x}}^{(i)} - \boldsymbol{f}(\boldsymbol{z}^{(i)}) + \boldsymbol{w}^{(i)} \right\}_{i=1}^{N} \tag{3.3}$$

collected from system (3.2) *is available, where the observation noise is i.i.d. as* $\boldsymbol{w}^{(i)} \sim \mathcal{N}(\boldsymbol{0}, \operatorname{diag}(\sigma_{\mathrm{n},1}^2, \dots, \sigma_{\mathrm{n},n}^2))$*,* $\forall i \in \mathbb{I}_N$*.*

We assume knowledge about a steady-state operating point $\boldsymbol{z}_{\mathrm{e}} = (\boldsymbol{x}_{\mathrm{e}}, \boldsymbol{u}_{\mathrm{e}})$ of system (3.2) satisfying

$$\boldsymbol{h}(\boldsymbol{z}_{\mathrm{e}}) = \boldsymbol{0}. \tag{3.4}$$

The assumption that $\boldsymbol{z}_{\mathrm{e}}$ is known does not represent a significant restriction since any unknown operating point $\boldsymbol{z}_{\mathrm{e}}$ satisfying (3.4) can easily be estimated by solving a nonlinear optimization problem involving $\boldsymbol{f}$ and an approximation of $\boldsymbol{g}$, cf. [1]. As we eventually want to design a sampled-data controller for system (3.2), and the theoretical results for sampled-data systems are formulated for linear systems, we are interested in the linearized dynamics of (3.2) around $\boldsymbol{z}_{\mathrm{e}}$.

Problem 3.1 (Probabilistic Learning of Linear Dynamics)**.** *Given the data set* (3.3)*, an equilibrium* $\boldsymbol{z}_{\mathrm{e}}$ *satisfying* (3.4) *and a desired confidence level* $\tilde{p} \in [0, 1)$*, we aim to compute a set* $\mathcal{C} = \mathcal{A} \times \mathcal{B} \subset \mathbb{R}^{n \times n_z}$*, where* $\mathcal{A} \subset \mathbb{R}^{n \times n}$ *and* $\mathcal{B} \subset \mathbb{R}^{n \times m}$*, such that the linearization of the partially unknown dynamics* (3.2) *at* $\boldsymbol{z}_{\mathrm{e}}$ *satisfies*

$$\Pr\left(\left. \frac{\partial \boldsymbol{h}(\boldsymbol{z})}{\partial \boldsymbol{z}} \right|_{\boldsymbol{z}_{\mathrm{e}}} \in \mathcal{C} \, \middle| \, \mathcal{D} \right) \geq \tilde{p}. \tag{3.5}$$

Solving Problem 3.1 requires making assumptions about the unknown residual dynamics $\boldsymbol{g}$. We consider two approaches in this chapter: In Section 3.2, we assume that $\boldsymbol{g}$ is linear in $\boldsymbol{z}$ and learn a probabilistic estimate of the dynamics using BLR. In Section 3.3, we assume that each component of $\boldsymbol{g}$ is drawn from a GP.

3.2 Bayesian Linear Regression for Linear Dynamics

We make the following assumption on the residual dynamics.

Assumption 3.3. *The residual dynamics* $\boldsymbol{g}$ *in* (3.2) *are linear in* $\boldsymbol{z}$*, i.e.,*

$$\boldsymbol{g}(t) = \begin{bmatrix} \boldsymbol{A}_g & \boldsymbol{B}_g \end{bmatrix} \boldsymbol{z}(t) = \boldsymbol{A}_g \boldsymbol{x}(t) + \boldsymbol{B}_g \boldsymbol{u}(t), \tag{3.6}$$

for some unknown matrices $\boldsymbol{A}_g \in \mathbb{R}^{n \times n}$ *and* $\boldsymbol{B}_g \in \mathbb{R}^{n \times m}$*.*

We use BLR as described in Section 2.2.1 to obtain a probabilistic estimate for each component g_i of $\boldsymbol{g}$. For this, we define parameter vectors $\boldsymbol{\theta}_i \in \mathbb{R}^{n_z}$, $i = 1, \dots, n_z$, as

$$\begin{bmatrix} \boldsymbol{\theta}_1^\top \\ \vdots \\ \boldsymbol{\theta}_n^\top \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_g & \boldsymbol{B}_g \end{bmatrix}. \tag{3.7}$$

Then, we can write each component of $\boldsymbol{g}$ as

$$g_i(\boldsymbol{z}(t)) = \boldsymbol{\theta}_i^\top \boldsymbol{\phi}_i(\boldsymbol{z}(t)), \qquad \forall i \in \mathbb{I}_n, \tag{3.8}$$

where the feature functions are given by $\boldsymbol{\phi}_i(\boldsymbol{z}(t)) = \boldsymbol{z}(t)$, $\forall i \in \mathbb{I}_n$. This linear model corresponds to (2.10). We define prior Gaussian distributions on the unknown parameter vectors denoted by

$$\boldsymbol{\theta}_i \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}_i}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}_i}), \qquad \forall i \in \mathbb{I}_n. \tag{3.9}$$

If a prior dynamics model is available, it can be used to define the prior means $\boldsymbol{\mu}_{\boldsymbol{\theta}_i}$. Otherwise, we can simply set $\boldsymbol{\mu}_{\boldsymbol{\theta}_i} = \boldsymbol{0}$, $\forall i \in \mathbb{I}_n$. The prior covariance matrices $\boldsymbol{\Sigma}_{\boldsymbol{\theta}_i}$ reflect the uncertainty about the prior model, and their diagonal entries should be set to high values if no approximate model is available. As explained in Section 2.2.1, after observing the data (3.3), we can compute the posterior parameter distributions

$$\boldsymbol{\theta}_i | \mathcal{D} \sim \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{\theta}_i|\mathcal{D}}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}_i|\mathcal{D}}\right). \tag{3.10}$$

Here, the expressions for the posterior mean $\boldsymbol{\mu}_{\boldsymbol{\theta}_i|\mathcal{D}}$ and the posterior variance $\boldsymbol{\Sigma}_{\boldsymbol{\theta}_i|\mathcal{D}}$ follow from (2.16) and (2.17) as

$$\boldsymbol{\mu}_{\boldsymbol{\theta}_i|\mathcal{D}} = \boldsymbol{\mu}_{\boldsymbol{\theta}} + \boldsymbol{\Sigma}_{\boldsymbol{\theta}_i} \boldsymbol{\Phi} \left(\sigma_{\mathrm{n},i}^2 \boldsymbol{I} + \boldsymbol{\Phi}^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}_i} \boldsymbol{\Phi}\right)^{-1} (\boldsymbol{t}_i - \boldsymbol{\Phi}^\top \boldsymbol{\mu}_{\boldsymbol{\theta}_i}) \tag{3.11}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{\theta}_i|\mathcal{D}} = \boldsymbol{\Sigma}_{\boldsymbol{\theta}_i} - \boldsymbol{\Sigma}_{\boldsymbol{\theta}_i} \boldsymbol{\Phi} \left(\sigma_{\mathrm{n},i}^2 \boldsymbol{I} + \boldsymbol{\Phi}^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}_i} \boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^\top \boldsymbol{\Sigma}_{\boldsymbol{\theta}_i}, \tag{3.12}$$

where $\boldsymbol{t}_i = \left[y_i^{(1)}, \dots, y_i^{(N)}\right]^\top \in \mathbb{R}^N$ and $\boldsymbol{\Phi} = \left[\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(N)}\right] \in \mathbb{R}^{n_z \times N}$ containing the training targets and training inputs, respectively. Note that only the i-th components of the training targets in $\mathcal{D}$ are used to calculate $\boldsymbol{\mu}_{\boldsymbol{\theta}_i|\mathcal{D}}$ and that the posterior variance does not depend on the training targets. From the posterior distribution (3.10), we can compute a probabilistic estimate of the linearization of the dynamics (3.2).

Theorem 3.1. *Under Assumptions 3.1 and 3.3 and given the data set* (3.3)*, for any confidence level* $p \in [0, 1)$,

$$\Pr\left(\left.\frac{\partial \boldsymbol{h}(\boldsymbol{z})}{\partial \boldsymbol{z}}\right|_{\boldsymbol{z}=\boldsymbol{z}_{\mathrm{e}}} \in \mathcal{C} \,\middle|\, \mathcal{D}\right) \geq p, \tag{3.13}$$

where $\mathcal{C} = \left[\boldsymbol{C} - \hat{\boldsymbol{C}}, \boldsymbol{C} + \hat{\boldsymbol{C}}\right] \subset \mathbb{R}^{n \times n_z}$ *is defined by*

$$\boldsymbol{C} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \end{bmatrix} = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{z})}{\partial \boldsymbol{z}}\right|_{\boldsymbol{z}_{\mathrm{e}}} + \begin{bmatrix} \boldsymbol{\mu}_{\boldsymbol{\theta}_1|\mathcal{D}}^\top \\ \vdots \\ \boldsymbol{\mu}_{\boldsymbol{\theta}_n|\mathcal{D}}^\top \end{bmatrix} \tag{3.14}$$

$$\hat{\boldsymbol{C}} = \begin{bmatrix} \hat{\boldsymbol{A}} & \hat{\boldsymbol{B}} \end{bmatrix} = \sqrt{\chi^2_{n_z}\left(\sqrt[n]{p}\right)} \begin{bmatrix} \sqrt{\operatorname{diag}\left(\boldsymbol{\Sigma}_{\boldsymbol{\theta}_1|\mathcal{D}}\right)}^\top \\ \vdots \\ \sqrt{\operatorname{diag}\left(\boldsymbol{\Sigma}_{\boldsymbol{\theta}_n|\mathcal{D}}\right)}^\top \end{bmatrix}. \tag{3.15}$$

Here, $\chi^2_{n_z}$ *is the quantile function of the chi-squared distribution with* n_z *degrees of freedom introduced in Section 2.1.1.*

Proof. Assumptions 3.1 and 3.3 imply that

$$\frac{\partial h_i(\boldsymbol{z})}{\partial \boldsymbol{z}} = \frac{\partial f_i(\boldsymbol{z})}{\partial \boldsymbol{z}} + \frac{\partial g_i(\boldsymbol{z})}{\partial \boldsymbol{z}} = \frac{\partial f_i(\boldsymbol{z})}{\partial \boldsymbol{z}} + \boldsymbol{\theta}_i^\top, \qquad \forall i \in \mathbb{I}_n.$$

Therefore,

$$\left(\frac{\partial h_i(\boldsymbol{z})}{\partial \boldsymbol{z}} \,\middle|\, \mathcal{D}\right)^\top \sim \mathcal{N}\left(\boldsymbol{\mu}_{\partial h_i}(\boldsymbol{z}), \boldsymbol{\Sigma}_{\boldsymbol{\theta}_i|\mathcal{D}}\right), \tag{3.16}$$

where $\boldsymbol{\mu}_{\partial h_i}(\boldsymbol{z}) = \left(\frac{\partial f_i(\boldsymbol{z})}{\partial \boldsymbol{z}}\right)^\top + \boldsymbol{\theta}_i$. By applying Lemma 2.2 to (3.16), we obtain that for any $\tilde{p} \in [0, 1)$,

$$\Pr\left(\frac{\partial h_i(\boldsymbol{z})}{\partial \boldsymbol{z}} \in \boldsymbol{\mu}_{\partial h_i}(\boldsymbol{z}) + \mathcal{B}_i \,\middle|\, \mathcal{D}\right) \geq \tilde{p},$$

where $\mathcal{B}_i$ is a box-shaped confidence region defined by

$$\begin{aligned} \mathcal{B}_i &= [-b_{i,1}, b_{i,1}] \times \ \ldots \ \times [-b_{i,n_z}, b_{i,n_z}] \\ &= \left[-\begin{bmatrix} b_{i,1} \\ \vdots \\ b_{i,n_z} \end{bmatrix}, \begin{bmatrix} b_{i,1} \\ \vdots \\ b_{i,n_z} \end{bmatrix} \right] \\ &= \left[-\sqrt{\chi^2_{n_z}(\tilde{p})} \operatorname{diag}\left(\boldsymbol{\Sigma}_{\boldsymbol{\theta}_i|\mathcal{D}}\right), \chi^2_{n_z}(\tilde{p}) \operatorname{diag}\left(\boldsymbol{\Sigma}_{\boldsymbol{\theta}_i|\mathcal{D}}\right) \right] \end{aligned}$$

where $b_{i,j} = \sqrt{\chi^2_{n_z}(\tilde{p})} \sqrt{\left(\boldsymbol{\Sigma}_{\boldsymbol{\theta}_i|\mathcal{D}}\right)_{jj}}$, $\forall j \in \mathbb{I}_{n_z}$, $\forall i \in \mathbb{I}_n$. Under the assumption that the parameter vectors are conditionally independent, the components of $\boldsymbol{h}$ and thus their partial derivatives are also independent, and we have

$$\Pr\left(\forall i \in \mathbb{I}_n : \frac{\partial h_i(\boldsymbol{z})}{\partial \boldsymbol{z}} \in \boldsymbol{\mu}_{\partial h_i}(\boldsymbol{z}) + \mathcal{B}_i \,\middle|\, \mathcal{D}\right) = \Pr\left(\left.\frac{\partial \boldsymbol{h}(\boldsymbol{z})}{\partial \boldsymbol{z}}\right|_{\boldsymbol{z}=\boldsymbol{z}_\mathrm{e}} \in \mathcal{C} \,\middle|\, \mathcal{D}\right) \geq \tilde{p}^n.$$

The result follows by substituting $\tilde{p} = \sqrt[n]{p}$. □

From Theorem 3.1, defining an uncertain learned system capturing the true dynamics with high probability is straightforward.

Corollary 3.1. *Under Assumptions 3.1 and 3.3 and given the data set (3.3), for any confidence level $p \in [0, 1)$, the linearization of the true dynamics (3.2) at $\boldsymbol{z} = \boldsymbol{z}_\mathrm{e}$ is captured with probability of at least p by*

$$\tilde{\boldsymbol{x}}(t) = \left(\boldsymbol{A} + \hat{\boldsymbol{A}} \circ \boldsymbol{\Omega}\right) \tilde{\boldsymbol{x}}(t) + \left(\boldsymbol{B} + \hat{\boldsymbol{B}} \circ \boldsymbol{\Psi}\right) \tilde{\boldsymbol{u}}(t), \tag{3.17}$$

where $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x}_\mathrm{e}$ and $\tilde{\boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{u}_\mathrm{e}$ are deviations from the equilibrium, $\boldsymbol{A}$ and $\boldsymbol{B}$ are given by (3.14), $\hat{\boldsymbol{A}}$ and $\hat{\boldsymbol{B}}$ are given by (3.15), and $\boldsymbol{\Omega} \in [-1, 1]^{n\times n}$ and $\boldsymbol{\Psi} \in [-1, 1]^{n\times m}$ are uncertain matrices.

If the known part of the dynamics $\boldsymbol{f}$ is equal to zero or, more generally speaking, linear in $\boldsymbol{z}$, then its partial derivative is independent of $\boldsymbol{z}$, and Corollary 3.1 holds globally for any state-input pair $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{u})$.

3.2.1 Bounding the Posterior Variance

It is evident from (3.12) that any data point $(\boldsymbol{z}, \boldsymbol{y}) \in \mathcal{D}$ with $\boldsymbol{z} \neq \boldsymbol{0}$ reduces model uncertainty. To relate the amount of data to uncertainty, we aim to find a lower bound of the norm of the covariance matrix after observing $M \in \mathbb{I}_N$ training samples. For this, we denote the subset of $\mathcal{D}$ consisting of the first M samples by $\mathcal{D}_M = \{\boldsymbol{z}^{(i)}, \boldsymbol{y}^{(i)}\}_{i=1}^{M} \subseteq \mathcal{D}$ and consider the case that the data arrives sequentially. Then, it follows from (2.23) that the parameter distribution after M observations can be calculated iteratively as

$$\boldsymbol{\theta}_i | \mathcal{D}_M \sim \mathcal{N}(\boldsymbol{\mu}_{i,M}, \boldsymbol{\Sigma}_{i,M}), \tag{3.18}$$

$$\boldsymbol{\mu}_{i,j} = \boldsymbol{\Sigma}_{i,j} \left(\boldsymbol{\Sigma}_{i,j-1}^{-1} \boldsymbol{\mu}_{i,j-1} + \frac{1}{\sigma_\mathrm{n}^2} \boldsymbol{z}^{(j)} y_i^{(j)} \right) \tag{3.19}$$

$$\boldsymbol{\Sigma}_{i,j} = \left(\boldsymbol{\Sigma}_{i,j-1}^{-1} + \frac{1}{\sigma_\mathrm{n}^2} \boldsymbol{z}^{(j)} \boldsymbol{z}^{(j)\top} \right)^{-1}, \qquad \forall j \in \mathbb{I}_M, \tag{3.20}$$

$\forall i \in \mathbb{I}_n$, where $\boldsymbol{\mu}_{i,0} = \boldsymbol{\mu}_{\boldsymbol{\theta}_i}$ and $\boldsymbol{\Sigma}_{i,0} = \boldsymbol{\Sigma}_{\boldsymbol{\theta}_i}$. It is more convenient to work with the inverse of the covariance matrix after M observations, called the precision matrix, which can be written compactly as

$$\boldsymbol{\Sigma}_{i,M}^{-1} = \boldsymbol{\Sigma}_{i,0}^{-1} + \frac{1}{\sigma_\mathrm{n}^2} \sum_{j=1}^{M} \boldsymbol{z}^{(j)} \boldsymbol{z}^{(j)\top}. \tag{3.21}$$

Any bound on the uncertainty requires an assumption about the training data distribution. To keep the discussion general, we only make an assumption about the support of the distribution.

Assumption 3.4. *The training inputs in* (3.3) *are drawn from a distribution with compact support* $\mathcal{Z} \subset \mathbb{R}^{n_z}$.

Consequently, there is an upper bound on the Euclidean norm of the training inputs, which we denote by

$$r = \max_{z \in \mathcal{Z}} \|\boldsymbol{z}\|_2. \tag{3.22}$$

This allows us to derive a lower bound on the covariance matrix.

Lemma 3.1. *Under Assumption 3.4, the posterior variance after* $M \in \mathbb{I}_N$ *observations can be lower bounded as*

$$\left\|\boldsymbol{\Sigma}_{i,M}\right\|_2 \geq \frac{\sigma_{\mathrm{n}}^2}{\sigma_{\mathrm{n}}^2 \left\|\boldsymbol{\Sigma}_{i,0}^{-1}\right\|_2 + rM}, \qquad \forall i \in \mathbb{I}_n, \tag{3.23}$$

where r *is given by* (3.22).

Proof. Due to the subadditivity of matrix norms, (3.21) implies

$$\left\|\boldsymbol{\Sigma}_{i,M}^{-1}\right\|_2 \leq \left\|\boldsymbol{\Sigma}_{i,0}^{-1}\right\|_2 + \frac{1}{\sigma_{\mathrm{n}}^2} \sum_{j=1}^{M} \left\|\boldsymbol{z}^{(j)} \boldsymbol{z}^{(j)\top}\right\|_2. \tag{3.24}$$

It is a standard result that given a vector $\boldsymbol{d}$, the outer product of $\boldsymbol{d}$ with itself satisfies $\|\boldsymbol{d}\boldsymbol{d}^\top\|_2 = \|\boldsymbol{d}\|_2^2$. Hence, we can make use of Assumption 3.4 to write (3.24) as

$$\left\|\boldsymbol{\Sigma}_{i,M}^{-1}\right\|_2 \leq \left\|\boldsymbol{\Sigma}_{i,0}^{-1}\right\|_2 + \frac{Mr}{\sigma_{\mathrm{n}}^2}.$$

Finally, the result follows from the fact that any invertible square matrix $\boldsymbol{D}$ satisfies the inequality $1 = \|\boldsymbol{I}\|_2 = \|\boldsymbol{D}\boldsymbol{D}^{-1}\|_2 \leq \|\boldsymbol{D}\|_2 \|\boldsymbol{D}^{-1}\|_2$. □

We evaluate the tightness of the derived bound with a simple example.

Example 3.1. *We consider learning the second row of a simple double-integrator system* $\dot{\boldsymbol{x}}(t) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \boldsymbol{x}(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$, *i.e., the true parameter vector is given by* $\boldsymbol{\theta} = [0,\, 0,\, 1]^\top$. *We set the prior mean vector to* $\boldsymbol{\mu}_{\boldsymbol{\theta}} = [0,\, 0,\, 1]^\top$ *and the prior covariance matrix to* $\boldsymbol{\Sigma}_{\boldsymbol{\theta}} = \boldsymbol{I}_3$. *We draw* $N = 100$ *training inputs uniformly from the set* $\mathcal{Z} = \{\boldsymbol{z} \in \mathbb{R}^3 | \, \|\boldsymbol{z}\|_\infty \leq 1\}$ *and set the noise variance to* $\sigma_{\mathrm{n}}^2 = 1$. *Fig. 3.1 shows the evolution of the norm of the parameter covariance* $\boldsymbol{\Sigma}_M$ *obtained iteratively via* (3.20) *for* $M = 1, \ldots, N$. *We repeat the simulation with ten different data sets and provide the*

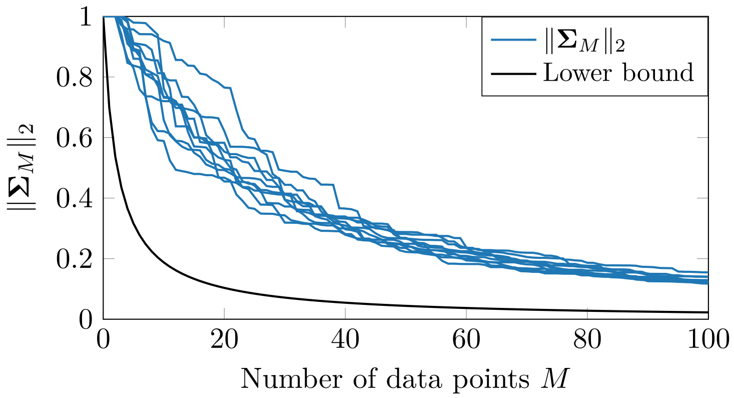

Figure 3.1: Evolution of the spectral norm of the parameter covariance matrix with an increasing amount of training samples compared to the theoretical lower bound (3.23).

lower bound calculated from Lemma 3.1 for comparison. On average, the value of the bound is 0.207 *times* $\|\mathbf{\Sigma}_M\|$.

3.3 Gaussian Process Regression and Linearization for Nonlinear Dynamics

Most real-world systems, particularly robots, do not exhibit linear dynamics but are affected by highly nonlinear effects such as friction, Coriolis, and centripetal forces. Therefore, we now take a more general approach to Problem 3.1 than in the previous section. In particular, we assume each component of the residual dynamics function $\boldsymbol{g}$ to be drawn from a GP with a known kernel function (see Section 2.2.2).

Assumption 3.5. *The components of the residual dynamics* $\boldsymbol{g}$ *in* (3.2) *are drawn from zero-mean GPs with SE kernel* k_i*, i.e.,* $g_i(\cdot) \sim \mathcal{GP}(0, k_i(\cdot,\cdot))$*,* $\forall i \in \mathbb{I}_n$*, where*

$$k_i(\boldsymbol{z}, \boldsymbol{z}') = \sigma_{\eta,i}^2 \exp\left(-\frac{1}{2}(\boldsymbol{z}-\boldsymbol{z}')^\top \boldsymbol{L}_i^{-2}(\boldsymbol{z}-\boldsymbol{z}')\right), \tag{3.25}$$

where $\sigma_{\eta,i}^2 > 0$*,* $\boldsymbol{L}_i = \mathrm{Diag}([l_{i,1},\ldots,l_{i,n_z}])$*,* $\forall i \in \mathbb{I}_n$*.*

This assumption is not restrictive in practice since the corresponding space of sample functions of the GP contains all continuous functions [30]. Other possible kernel functions include the Matérn kernel or the linear kernel [31].

Remark 3.1. *The hyperparameters of* (3.25) *can be interpreted physically [36]: The prior variance* $\sigma_{\eta,i}^2$ *indicates the confidence intervals* $[-\gamma\sigma_{\eta,i}, \gamma\sigma_{\eta,i}]$*,* $\gamma > 0$*, within which* g_i*; the residual derivative of the* i*-th state* x_i*, takes on values with high probability. The length scales* $l_{i,1},\ldots,l_{i,n_z}$ *correspond to the notion that a change of* $l_{i,j}$ *in the* j*-th dimension of the stacked state and input* $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{u})$ *can result in a large change of* g_i*.*

Consequently, a large value of $l_{i,j}$ implies that the j-th dimension of $\boldsymbol{z}$ has a small impact on g_i.

In practice, an initial guess is typically made for the hyperparameters, which should take available prior knowledge about the residual dynamics $\boldsymbol{g}$ into account. Starting from this initialization, suitable values for the hyperparameters can be obtained by maximizing the marginal log-likelihood of the training data, as explained in Section 2.2.2.

As we are interested in the linearized dynamics around the operating point $\boldsymbol{z}_\mathrm{e} = (\boldsymbol{x}_\mathrm{e}, \boldsymbol{u}_\mathrm{e})$, we compute the derivative of the i-th GP around $\boldsymbol{z}_\mathrm{e}$, denoted by

$$\left.\frac{\partial g_i(\boldsymbol{z})}{\partial \boldsymbol{z}}\right|_{\boldsymbol{z}_\mathrm{e}} \sim \mathcal{N}(\boldsymbol{\mu}'_i(\boldsymbol{z}_\mathrm{e}), \boldsymbol{\Sigma}'_i(\boldsymbol{z}_\mathrm{e})), \tag{3.26}$$

via (2.32) and (2.33). We obtain the derivatives of the GP mean and variance

$$\boldsymbol{\mu}'_i(\boldsymbol{z}_\mathrm{e}) = \left.\frac{\partial \boldsymbol{k}_i(\boldsymbol{z})}{\partial \boldsymbol{z}}\right|_{\boldsymbol{z}=\boldsymbol{z}_\mathrm{e}} \bar{\boldsymbol{K}}_i^{-1}\boldsymbol{y} \tag{3.27}$$

$$\boldsymbol{\Sigma}'_i(\boldsymbol{z}_\mathrm{e}) = \left.\frac{\partial^2 k_i(\boldsymbol{z},\boldsymbol{z})}{\partial \boldsymbol{z}\partial \boldsymbol{z}}\right|_{\boldsymbol{z}=\boldsymbol{z}_\mathrm{e}} - \left.\frac{\partial \boldsymbol{k}_i(\boldsymbol{z})}{\partial \boldsymbol{z}}\right|_{\boldsymbol{z}=\boldsymbol{z}_\mathrm{e}} \bar{\boldsymbol{K}}_i^{-1} \left(\left.\frac{\partial \boldsymbol{k}_i(\boldsymbol{z})}{\partial \boldsymbol{z}}\right|_{\boldsymbol{z}=\boldsymbol{z}_\mathrm{e}}\right)^\top, \tag{3.28}$$

where the gram vector is given by $\boldsymbol{k}_i(\boldsymbol{z}) = \left[k_i(\boldsymbol{z}^{(1)},\boldsymbol{z}), \ldots, k_i(\boldsymbol{z}^{(N)},\boldsymbol{z})\right]^\top \in \mathbb{R}^{n_z}$ and the gram matrix is given by $\bar{\boldsymbol{K}}_i = \boldsymbol{K}_i + \sigma_\mathrm{n}^2 \boldsymbol{I}_{n_z} \in \mathbb{R}^{n_z \times n_z}$, where $\left(\boldsymbol{K}_i\right)_{jl} = k_i(\boldsymbol{z}^{(j)}, \boldsymbol{z}^{(l)})$, $\forall i \in \mathbb{I}_n$. Similar to Section 3.2, we can use (3.26) to compute a probabilistic estimate of the linearized dynamics at the operating point.

Theorem 3.2. *Under Assumptions 3.1 and 3.5 and given the data set (3.3), for any confidence threshold $p \in [0, 1)$,*

$$\Pr\left(\left.\frac{\boldsymbol{h}(\boldsymbol{z})}{\partial \boldsymbol{z}}\right|_{\boldsymbol{z}=\boldsymbol{z}_\mathrm{e}} \in \mathcal{C} \,\middle|\, \mathcal{D}\right) \geq \tilde{p}, \tag{3.29}$$

where $\mathcal{C} = \left[\boldsymbol{C} - \hat{\boldsymbol{C}}, \boldsymbol{C} + \hat{\boldsymbol{C}}\right] \subset \mathbb{R}^{n \times n_z}$ is defined by

$$\boldsymbol{C} = \begin{bmatrix}\boldsymbol{A} & \boldsymbol{B}\end{bmatrix} = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{z})}{\partial \boldsymbol{z}}\right|_{\boldsymbol{z}_\mathrm{e}} + \begin{bmatrix}\boldsymbol{\mu}'_1(\boldsymbol{z}_\mathrm{e})^\top \\ \vdots \\ \boldsymbol{\mu}'_n(\boldsymbol{z}_\mathrm{e})^\top\end{bmatrix} \tag{3.30}$$

$$\hat{\boldsymbol{C}} = \begin{bmatrix}\hat{\boldsymbol{A}} & \hat{\boldsymbol{B}}\end{bmatrix} = \sqrt{\chi^2_{n_z}\left(\sqrt[n]{p}\right)} \begin{bmatrix}\sqrt{\mathrm{diag}(\boldsymbol{\Sigma}'_1(\boldsymbol{z}_\mathrm{e}))^\top} \\ \vdots \\ \sqrt{\mathrm{diag}(\boldsymbol{\Sigma}'_n(\boldsymbol{z}_\mathrm{e}))^\top}\end{bmatrix}. \tag{3.31}$$

Proof. The proof is very similar to that of Theorem 3.1 and therefore omitted. □

It follows from Theorem 3.2 that the state derivative at the equilibrium can be probabilistically bounded.

Corollary 3.2. *Under Assumptions 3.1 and 3.5 and given the data set* (3.3)*, for any* $p \in [0,1)$*, the linearization of the true dynamics* (3.2) *at* $\boldsymbol{z} = \boldsymbol{z}_\mathrm{e}$ *is captured with probability of at least* p *by*

$$\dot{\tilde{\boldsymbol{x}}}(t) = \left(\boldsymbol{A} + \hat{\boldsymbol{A}} \circ \boldsymbol{\Omega}\right) \tilde{\boldsymbol{x}}(t) + \left(\boldsymbol{B} + \hat{\boldsymbol{B}} \circ \boldsymbol{\Psi}\right) \tilde{\boldsymbol{u}}(t), \tag{3.32}$$

where $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x}_\mathrm{e}$*,* $\tilde{\boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{u}_\mathrm{e}$*,* $\boldsymbol{A}$ *and* $\boldsymbol{B}$ *are given by* (3.30)*,* $\hat{\boldsymbol{A}}$ *and* $\hat{\boldsymbol{B}}$ *are given by* (3.31)*, and* $\boldsymbol{\Omega} \in [-1,1]^{n\times n}$ *and* $\boldsymbol{\Psi} \in [-1,1]^{n\times m}$ *are uncertain matrices.*

Unlike Corollary 3.1, even when the known component of the dynamics $\boldsymbol{f}$ is linear, Corollary 3.2 can only be defined *at* the equilibrium, since the linearization of $\boldsymbol{g}$ via (3.30) and (3.31) depends on $\boldsymbol{z}_\mathrm{e}$. Despite the probabilistic bounds in Theorem 3.2, computing the probability that (3.32) captures the true dynamics at a given state-input pair $(\boldsymbol{x}, \boldsymbol{u}) \neq \boldsymbol{z}_\mathrm{e}$ is generally very difficult. Nevertheless, the uncertain linearized system (3.32) can be used for robust control design in practice, provided that the deviations from the equilibrium are kept small in the sense that they do not result in a big change of the derivative $\frac{\partial \boldsymbol{h}(\boldsymbol{z})}{\partial \boldsymbol{z}}$. For any controller designed for the linearized system, theoretical statements about the stability of the resulting nonlinear closed-loop system would require a study of its domain of attraction [37].

In this section, we first learn the nonlinear residual dynamics $\boldsymbol{g}$ and then linearize the system. As an alternative to this approach, linear methods such as BLR or the Kalman filter could, in principle, be used to directly estimate the linearized dynamics (3.32). An advantage of using GPs in this context is that their hyperparameters define a local region around the equilibrium in which the system behaves approximately linearly, resulting in a more accurate estimate of the local linearization of the dynamics [1].

3.4 Uncertainty Reparameterization

In Sections 3.2 and 3.3, we have obtained an uncertain linearized system of the form

$$\dot{\tilde{\boldsymbol{x}}}(t) = \left(\boldsymbol{A} + \hat{\boldsymbol{A}} \circ \boldsymbol{\Omega}\right) \tilde{\boldsymbol{x}}(t) + \left(\boldsymbol{B} + \hat{\boldsymbol{B}} \circ \boldsymbol{\Psi}\right) \tilde{\boldsymbol{u}}(t). \tag{3.33}$$

Ultimately, we aim to design a robust sampled-data controller for this system. However, it is difficult to directly take the two Hadamard product terms in (3.33) into account for robust stability analysis. To address this problem, we derive the following result.

Lemma 3.2. *Let* $\boldsymbol{X} = \left[\boldsymbol{x}_1 \, \dots \, \boldsymbol{x}_n\right]^\top \in \mathbb{R}^{n\times m}$ *and* $\boldsymbol{Y} = \left[\boldsymbol{y}_1 \, \dots \, \boldsymbol{y}_n\right]^\top \in \mathbb{R}^{n\times m}$*. Then, the*

Hadamard matrix product $\boldsymbol{X} \circ \boldsymbol{Y}$ *satisfies*

$$\boldsymbol{X} \circ \boldsymbol{Y} = (\boldsymbol{I}_n \otimes \mathbf{1}_{1\times n})\mathrm{Diag}\big(\mathrm{vec}(\boldsymbol{Y}^\top)\big)\begin{bmatrix}\mathrm{Diag}(\boldsymbol{x}_1)\\ \vdots \\ \mathrm{Diag}(\boldsymbol{x}_n)\end{bmatrix}, \tag{3.34}$$

where $\otimes$ *denotes the Kronecker product, and* $\mathrm{vec}(\boldsymbol{Y}^\top) = \left[\boldsymbol{y}_1^\top, \dots, \boldsymbol{y}_n^\top\right]^\top \in \mathbb{R}^{nm}$ *is the vectorization of* $\boldsymbol{Y}^\top$*.*

Proof. The i-th row of $\boldsymbol{X} \circ \boldsymbol{Y}$ is given by

$$\begin{aligned}\boldsymbol{x}_i^\top \circ \boldsymbol{y}_i^\top &= \begin{bmatrix} X_{i1}Y_{i1} & \dots & X_{im}Y_{im}\end{bmatrix}\\ &= \mathbf{1}_m^\top \begin{bmatrix} Y_{i1} & 0 & \dots & 0\\ 0 & Y_{i2} & \dots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \dots & Y_{im}\end{bmatrix}\begin{bmatrix} X_{i1} & 0 & \dots & 0\\ 0 & X_{i2} & \dots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \dots & X_{im}\end{bmatrix}\\ &= \mathbf{1}_m^\top \mathrm{Diag}(\boldsymbol{y}_i)\mathrm{Diag}(\boldsymbol{x}_i), \qquad \forall i \in \mathbb{I}_n.\end{aligned}$$

Hence, the Hadamard product of $\boldsymbol{X}$ and $\boldsymbol{Y}$ can be written as

$$\begin{aligned}\boldsymbol{X} \circ \boldsymbol{Y} &= \begin{bmatrix}\mathbf{1}_m^\top \mathrm{Diag}(\boldsymbol{y}_1)\mathrm{Diag}(\boldsymbol{x}_1)\\ \vdots \\ \mathbf{1}_m^\top \mathrm{Diag}(\boldsymbol{y}_n)\mathrm{Diag}(\boldsymbol{x}_n)\end{bmatrix}\\ &= \underbrace{\begin{bmatrix}\mathbf{1}_m^\top & \mathbf{0}_m^\top & \dots & \mathbf{0}_m^\top\\ \mathbf{0}_m^\top & \mathbf{1}_m^\top & \dots & \mathbf{0}_m^\top\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{0}_m^\top & \mathbf{0}_m^\top & \dots & \mathbf{1}_m^\top\end{bmatrix}}_{=\boldsymbol{I}_n \otimes \mathbf{1}_{1\times m}}\underbrace{\begin{bmatrix}\mathrm{Diag}(\boldsymbol{y}_1) & \mathbf{0}_{m\times m} & \dots & \mathbf{0}_{m\times m}\\ \mathbf{0}_{m\times m} & \mathrm{Diag}(\boldsymbol{y}_2) & \dots & \mathbf{0}_{m\times m}\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{0}_{m\times m} & \mathbf{0}_{m\times m} & \dots & \mathrm{Diag}(\boldsymbol{y}_n)\end{bmatrix}}_{=\mathrm{Diag}(\mathrm{vec}(\boldsymbol{Y}^\top))}\begin{bmatrix}\mathrm{Diag}(\boldsymbol{x}_1)\\ \mathrm{Diag}(\boldsymbol{x}_2)\\ \vdots \\ \mathrm{Diag}(\boldsymbol{x}_n)\end{bmatrix},\end{aligned}$$

which concludes the proof. □

We use Lemma 3.2 to reparameterize the Hadamard product terms in (3.33). Denoting the rows of $\hat{\boldsymbol{A}}$ and $\hat{\boldsymbol{B}}$ by $\hat{\boldsymbol{a}}_1^\top, \dots, \hat{\boldsymbol{a}}_n^\top$ and $\hat{\boldsymbol{b}}_1^\top, \dots, \hat{\boldsymbol{b}}_n^\top$, respectively, we can rewrite

$$\hat{\boldsymbol{A}} \circ \boldsymbol{\Omega} = \boldsymbol{H}_1\boldsymbol{\Delta}_1\boldsymbol{E} \tag{3.35}$$

$$\hat{\boldsymbol{B}} \circ \boldsymbol{\Psi} = \boldsymbol{H}_2\boldsymbol{\Delta}_2\boldsymbol{F}, \tag{3.36}$$

where

$$\boldsymbol{H}_1 = \boldsymbol{I}_n \otimes \boldsymbol{1}_{1\times n} \in \mathbb{R}^{n\times n^2} \tag{3.37}$$
$$\boldsymbol{H}_2 = \boldsymbol{I}_n \otimes \boldsymbol{1}_{1\times m} \in \mathbb{R}^{n\times nm} \tag{3.38}$$
$$\boldsymbol{\Delta}_1 = \mathrm{Diag}(\delta_{1,1},\dots,\delta_{1,n^2}) \in \mathbb{R}^{n^2\times n^2}, \qquad |\delta_{1,i}| \leq 1,\ \forall i \in \mathbb{I}_{n^2} \tag{3.39}$$
$$\boldsymbol{\Delta}_2 = \mathrm{Diag}(\delta_{2,1},\dots,\delta_{2,nm}) \in \mathbb{R}^{nm\times nm}, \qquad |\delta_{2,i}| \leq 1,\ \forall i \in \mathbb{I}_{nm} \tag{3.40}$$
$$\bar{\boldsymbol{E}} = \begin{bmatrix} \mathrm{Diag}(\hat{\boldsymbol{a}}_1) \\ \vdots \\ \mathrm{Diag}(\hat{\boldsymbol{a}}_n) \end{bmatrix} \in \mathbb{R}^{n^2\times n} \tag{3.41}$$
$$\bar{\boldsymbol{F}} = \begin{bmatrix} \mathrm{Diag}(\hat{\boldsymbol{b}}_1) \\ \vdots \\ \mathrm{Diag}(\hat{\boldsymbol{b}}_n) \end{bmatrix} \in \mathbb{R}^{nm\times n}. \tag{3.42}$$

Here, all matrices except for $\boldsymbol{\Delta}_1$ and $\boldsymbol{\Delta}_2$ are known. By stacking $\boldsymbol{\Delta}_1$ and $\boldsymbol{\Delta}_2$ diagonally into a single matrix, we obtain the following result to reparameterize the model uncertainty.

Lemma 3.3. *The Hadamard product terms in (3.33) can be reparameterized as*

$$\hat{\boldsymbol{A}} \circ \boldsymbol{\Omega} = \boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{E} \tag{3.43}$$
$$\hat{\boldsymbol{B}} \circ \boldsymbol{\Psi} = \boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{F}, \tag{3.44}$$

where

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{H}_1 & \boldsymbol{H}_2 \end{bmatrix} \in \mathbb{R}^{n\times(n^2+nm)} \tag{3.45}$$
$$\boldsymbol{\Delta} = \begin{bmatrix} \boldsymbol{\Delta}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Delta}_2 \end{bmatrix} \in \mathbb{R}^{(n^2+nm)\times(n^2+nm)} \tag{3.46}$$
$$\boldsymbol{E} = \begin{bmatrix} \bar{\boldsymbol{E}} \\ \boldsymbol{0}_{nm\times n} \end{bmatrix} \in \mathbb{R}^{(n^2+nm)\times n} \tag{3.47}$$
$$\boldsymbol{F} = \begin{bmatrix} \boldsymbol{0}_{n^2\times n} \\ \bar{\boldsymbol{F}} \end{bmatrix} \in \mathbb{R}^{(n^2+nm)\times n}, \tag{3.48}$$

where $\boldsymbol{H}_1$, $\boldsymbol{H}_2$, $\boldsymbol{\Delta}_1$, $\boldsymbol{\Delta}_2$, $\bar{\boldsymbol{E}}$, and $\bar{\boldsymbol{F}}$ are given by (3.37) - (3.42).

Although the matrix $\boldsymbol{\Delta}$ in (3.46) is unknown, we know from its definition via (3.39) and (3.40) that $\boldsymbol{\Delta}^\top\boldsymbol{\Delta} \preceq \boldsymbol{I}$. This property can be used for robust stability analysis, as shown in Section 4.2. Using Lemma 3.3, we can rewrite (3.33) as

$$\dot{\tilde{\boldsymbol{x}}}(t) = (\boldsymbol{A} + \boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{E})\tilde{\boldsymbol{x}}(t) + (\boldsymbol{B} + \boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{F})\tilde{\boldsymbol{u}}(t). \tag{3.49}$$

This represents an uncertain linear system with norm-bounded uncertainty capturing the true dynamics with high probability. In the next chapter, we consider robust sampled-data control of (3.49).

Chapter 4

Learning-Based Robust Sampled-Data Control

Having learned the linearized dynamics of a system with partially unknown dynamics in Chapter 3, we now discuss robust sampled-data control of the learned system. This perspective allows us to relate control frequency and data in different ways. In this chapter, we derive stability conditions depending on the control frequency and the model uncertainty and formulate an optimization problem for computing the minimum required control frequency. Under additional assumptions, we also derive a lower bound on the amount of data needed to control an uncertain system at a given frequency.

4.1 Problem Statement

In Chapter 3, we obtain for both linear and nonlinear residual dynamics an uncertain linearized continuous-time system of the form

$$\dot{\boldsymbol{x}}(t) = (\boldsymbol{A} + \boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{E})\boldsymbol{x}(t) + (\boldsymbol{B} + \boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{F})\boldsymbol{u}(t), \tag{4.1}$$

where $\boldsymbol{\Delta} \in \mathbb{R}^{p\times p}$ is an unknown matrix satisfying $\boldsymbol{\Delta}^\top\boldsymbol{\Delta} \preceq \boldsymbol{I}$, and $\boldsymbol{H} \in \mathbb{R}^{n\times p}$, $\boldsymbol{E} \in \mathbb{R}^{p\times n}$, and $\boldsymbol{F} \in \mathbb{R}^{p\times m}$ are known matrices, where $p = n^2 + nm$. In this chapter, we investigate the connection between the model uncertainty and the frequency at which the learned system (4.1) is controlled. To this end, we employ the sampled-data system description introduced in Section 2.3, as it allows us to control a system at different frequencies based on the same (uncertain) dynamics model. We consider a linear discrete-time control law

$$\boldsymbol{u}(t) = \boldsymbol{K}\boldsymbol{x}(t_k), \quad \forall t \in [t_k, t_{k+1}), \tag{4.2}$$

where $\boldsymbol{K} \in \mathbb{R}^{m\times n}$ is the state feedback matrix, and the sampling instants t_k, $k \in \mathbb{N}_0$, satisfy Assumption 2.2. Combining the continuous-time dynamics (4.1) with the discrete-time controller (4.2) results in the uncertain learned closed-loop sampled-data system

$$\dot{\boldsymbol{x}}(t) = (\boldsymbol{A} + \boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{E})\boldsymbol{x}(t) + (\boldsymbol{B} + \boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{F})\boldsymbol{K}\boldsymbol{x}(t_k), \qquad \forall t \in [t_k, t_{k+1}), \qquad k \in \mathbb{N}_0. \tag{4.3}$$

It is of fundamental interest at which control frequency feedback has to be applied in (4.3) such that stability is guaranteed.

Definition 4.1. *Let $T_{\mathrm{s,max}}$ be the largest value of T_{s} such that the system* (4.1) *with the control* (4.2) *satisfying Assumption 2.2 can be robustly stabilized. Then, the MCF is given by $f_{\mathrm{c,min}} = \frac{1}{T_{\mathrm{s,max}}}$.*

In other words, the MCF is the minimum rate at which the control input must be recalculated based on current state measurements to guarantee stability despite model uncertainty.

Problem 4.1. *Compute the MCF defined in Definition 4.1.*

Remark 4.1. *Computing the MCF as defined in Definition 4.1 exactly is generally intractable [9]. Therefore, the methods presented in this chapter provide only a lower bound on the MCF. For simplicity, we nevertheless refer to the solution as MCF.*

Without the uncertainty in (4.1) and under the assumption of periodic sampling, Problem 4.1 could easily be solved by discretizing (4.1) for increasing sampling times until the resulting discrete-time system becomes uncontrollable. An approximate discretized dynamics model could be obtained for polytopic uncertainty using the approach described in [38]. However, due to the norm-bounded uncertainty in (4.1), which does not allow for a discretization of the system, Problem 4.1 represents a challenging problem.

In this chapter, we address Problem 4.1 by deriving conditions for the robust stability of (4.3) and formulating an optimization problem based on these conditions to compute the MCF. Then, we discuss optimizing the performance of sampled-data controllers for different control frequencies.

4.2 Robust Stability for the Learned Uncertain System

Both Theorem 3.1 and Theorem 3.2 define a set $\mathcal{C}$ containing the dynamics matrix and input matrix of the true linearized dynamics with high probability. By definition, the set $\mathcal{C}$ represents a polytope with $M = 2^{nn_z}$ vertices. As pointed out in Remark 2.3, the stability conditions for sampled data systems derived in [19, 20, 21] can also guarantee stability in the case of polytopic-type uncertainty. However, the number of vertices to evaluate exponentially increases with the state and input dimensions. For example, for a relatively small system with $n = 3$ states and $m = 2$ inputs, the stability conditions have to be satisfied on all $M = 2^{15} = 32768$ vertices of $\mathcal{C}$ to guarantee robust stability. This shows that the polytopic uncertainty approach to stability analysis of (4.1) is computationally infeasible for an uncertain learned dynamics model except for very low-dimensional systems. To overcome this problem, we derive robust stability conditions that can take the norm-bounded uncertainty in the learned continuous-time dynamics (4.1) into account in a computationally more efficient way than the polytopic uncertainty approach. Before presenting the stability conditions, we provide a lemma that is used in their derivation.

Lemma 4.1. [39] *Let $\boldsymbol{\Theta}^\top\boldsymbol{\Theta} \preceq \boldsymbol{I}$. Then, for all constant matrices $\boldsymbol{U}$, $\boldsymbol{V}$ of appropriate dimension and all scalars $\epsilon > 0$, it holds that*

$$-\epsilon^{-1}\boldsymbol{U}\boldsymbol{U}^\top - \epsilon\boldsymbol{V}^\top\boldsymbol{V} \preceq \boldsymbol{U}\boldsymbol{\Theta}\boldsymbol{V} + \boldsymbol{V}^\top\boldsymbol{\Theta}^\top\boldsymbol{U}^\top \preceq \epsilon^{-1}\boldsymbol{U}\boldsymbol{U}^\top + \epsilon\boldsymbol{V}^\top\boldsymbol{V}. \tag{4.4}$$

Theorem 4.1. *The control law (4.2) stabilizes the uncertain system (4.1) for all samplings satisfying Assumption 2.2 if there exist matrices $\boldsymbol{Q}_1 \succ \boldsymbol{0}, \boldsymbol{Q}_2, \boldsymbol{Q}_3, \bar{\boldsymbol{Z}}_1, \bar{\boldsymbol{Z}}_2, \bar{\boldsymbol{Z}}_3, \bar{\boldsymbol{R}} \succ \boldsymbol{0}$, all in $\mathbb{R}^{n\times n}$, $\bar{\boldsymbol{Y}} \in \mathbb{R}^{m\times n}$ and scalars $\epsilon_1 > 0$, $\epsilon_2 > 0$, that satisfy the matrix inequalities*

$$\begin{bmatrix} \boldsymbol{Q}_2 + \boldsymbol{Q}_2^\top + T_\mathrm{s}\bar{\boldsymbol{Z}}_1 & \boldsymbol{\Phi}_{12} & T_\mathrm{s}\boldsymbol{Q}_2^\top & \boldsymbol{0} & \epsilon_1(\boldsymbol{Q}_1\boldsymbol{E}^\top + \bar{\boldsymbol{Y}}^\top\boldsymbol{F}^\top) \\ * & -\boldsymbol{Q}_3 - \boldsymbol{Q}_3^\top + T_\mathrm{s}\bar{\boldsymbol{Z}}_3 & T_\mathrm{s}\boldsymbol{Q}_3^\top & \boldsymbol{H} & \boldsymbol{0} \\ * & * & -T_\mathrm{s}\bar{\boldsymbol{R}} & \boldsymbol{0} & \boldsymbol{0} \\ * & * & * & -\epsilon_1\boldsymbol{I} & \boldsymbol{0} \\ * & * & * & * & -\epsilon_1\boldsymbol{I} \end{bmatrix} \prec \boldsymbol{0}, \tag{4.5}$$

$$\begin{bmatrix} 2\boldsymbol{Q}_1 - \bar{\boldsymbol{R}} & \boldsymbol{0} & \bar{\boldsymbol{Y}}^\top\boldsymbol{B}^\top & \boldsymbol{0} & \epsilon_2\boldsymbol{Y}^\top\boldsymbol{F}^\top \\ * & \bar{\boldsymbol{Z}}_1 & \bar{\boldsymbol{Z}}_2 & \boldsymbol{0} & \boldsymbol{0} \\ * & * & \bar{\boldsymbol{Z}}_3 & \boldsymbol{H} & \boldsymbol{0} \\ * & * & * & \epsilon_2\boldsymbol{I} & \boldsymbol{0} \\ * & * & * & * & \epsilon_2\boldsymbol{I} \end{bmatrix} \succeq \boldsymbol{0}, \tag{4.6}$$

where $\boldsymbol{\Phi}_{12} = \boldsymbol{Q}_3 - \boldsymbol{Q}_2^\top + \boldsymbol{Q}_1\boldsymbol{A}^\top + T_\mathrm{s}\bar{\boldsymbol{Z}}_2 + \boldsymbol{Y}^\top\boldsymbol{B}^\top$. The state-feedback gain is then given by $\boldsymbol{K} = \bar{\boldsymbol{Y}}\boldsymbol{Q}_1^{-1}$.

Proof. The idea is to extend Lemma 2.5 such that stability of (4.3) is robustly guaranteed despite the additional uncertain terms $\boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{E}$ and $\boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{F}$ compared to (2.38). First, we replace $\boldsymbol{A}$ and $\boldsymbol{B}$ in the first LMI (2.57) in Lemma 2.5 by $\boldsymbol{A} + \boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{E}$ and $\boldsymbol{B} + \boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{F}$, respectively, and obtain

$$\begin{aligned} &\begin{bmatrix} \boldsymbol{Q}_2 + \boldsymbol{Q}_2^\top + T_\mathrm{s}\bar{\boldsymbol{Z}}_1 & \boldsymbol{Q}_3 - \boldsymbol{Q}_2^\top + \boldsymbol{Q}_1(\boldsymbol{A} + \boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{E})^\top + T_\mathrm{s}\bar{\boldsymbol{Z}}_2 + \bar{\boldsymbol{Y}}^\top(\boldsymbol{B} + \boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{F})^\top & T_\mathrm{s}\boldsymbol{Q}_2^\top \\ * & -\boldsymbol{Q}_3 - \boldsymbol{Q}_3^\top + T_\mathrm{s}\bar{\boldsymbol{Z}}_3 & T_\mathrm{s}\boldsymbol{Q}_3^\top \\ * & * & -T_\mathrm{s}\bar{\boldsymbol{R}} \end{bmatrix} \\ &= \boldsymbol{W} + \begin{bmatrix} \boldsymbol{0} & \boldsymbol{Q}_1\boldsymbol{E}^\top\boldsymbol{\Delta}^\top\boldsymbol{H}^\top + \bar{\boldsymbol{Y}}^\top\boldsymbol{F}^\top\boldsymbol{\Delta}^\top\boldsymbol{H}^\top & \boldsymbol{0} \\ * & \boldsymbol{0} & \boldsymbol{0} \\ * & * & \boldsymbol{0} \end{bmatrix} \\ &= \boldsymbol{W} + \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{H} \\ \boldsymbol{0} \end{bmatrix} \boldsymbol{\Delta} \begin{bmatrix} \boldsymbol{E}\boldsymbol{Q}_1^\top + \boldsymbol{F}\bar{\boldsymbol{Y}} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} + \begin{bmatrix} \boldsymbol{Q}_1\boldsymbol{E}^\top + \bar{\boldsymbol{Y}}^\top\boldsymbol{F}^\top \\ \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix} \boldsymbol{\Delta}^\top \begin{bmatrix} \boldsymbol{0} & \boldsymbol{H}^\top & \boldsymbol{0} \end{bmatrix} \prec \boldsymbol{0}, \end{aligned} \tag{4.7}$$

where $\boldsymbol{W}$ denotes the left-hand-side of (2.57). Applying Lemma 4.1 to the second and third term of the right-hand-side of (4.7) leads to

$$
\begin{aligned}
&\begin{bmatrix} \mathbf{0} \\ \boldsymbol{H} \\ \mathbf{0} \end{bmatrix} \boldsymbol{\Delta} \begin{bmatrix} \boldsymbol{E}\boldsymbol{Q}_1^\top + \boldsymbol{F}\bar{\boldsymbol{Y}} & \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \boldsymbol{Q}_1\boldsymbol{E}^\top + \bar{\boldsymbol{Y}}^\top\boldsymbol{F}^\top \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \boldsymbol{\Delta}^\top \begin{bmatrix} \mathbf{0} & \boldsymbol{H}^\top & \mathbf{0} \end{bmatrix} \\
&\preceq \epsilon_1^{-1} \begin{bmatrix} \mathbf{0} \\ \boldsymbol{H} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & \boldsymbol{H}^\top & \mathbf{0} \end{bmatrix} + \epsilon_1 \begin{bmatrix} \boldsymbol{Q}_1\boldsymbol{E}^\top + \bar{\boldsymbol{Y}}^\top\boldsymbol{F}^\top \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{E}\boldsymbol{Q}_1^\top + \boldsymbol{F}\bar{\boldsymbol{Y}} & \mathbf{0} & \mathbf{0} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{0} & \epsilon_1(\boldsymbol{Q}_1\boldsymbol{E}^\top + \bar{\boldsymbol{Y}}^\top\boldsymbol{F}^\top) \\ \boldsymbol{H} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \epsilon_1^{-1}\boldsymbol{I} & \mathbf{0} \\ \mathbf{0} & \epsilon_1^{-1}\boldsymbol{I} \end{bmatrix} \begin{bmatrix} \mathbf{0} & \boldsymbol{H}^\top & \mathbf{0} \\ \epsilon_1(\boldsymbol{E}\boldsymbol{Q}_1^\top + \boldsymbol{F}\bar{\boldsymbol{Y}}) & \mathbf{0} & \mathbf{0} \end{bmatrix},
\end{aligned} \tag{4.8}
$$

for all scalars $\epsilon_1 > 0$. Finally, inserting (4.8) into (4.7) and applying the Schur complement gives (4.5). In a similar way, we replace $\boldsymbol{B}$ in the second LMI (2.58) in Lemma 2.5 by $\boldsymbol{B} + \boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{F}$, which results in

$$
\begin{aligned}
&\begin{bmatrix} \boldsymbol{Q}_1\bar{\boldsymbol{R}}^{-1}\boldsymbol{Q}_1 & \mathbf{0} & \bar{\boldsymbol{Y}}^\top(\boldsymbol{B} + \boldsymbol{H}\boldsymbol{\Delta}\boldsymbol{F})^\top \\ * & \bar{\boldsymbol{Z}}_1 & \bar{\boldsymbol{Z}}_2 \\ * & * & \bar{\boldsymbol{Z}}_3 \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{Q}_1\bar{\boldsymbol{R}}^{-1}\boldsymbol{Q}_1 & \mathbf{0} & \bar{\boldsymbol{Y}}^\top\boldsymbol{B}^\top \\ * & \bar{\boldsymbol{Z}}_1 & \bar{\boldsymbol{Z}}_2 \\ * & * & \bar{\boldsymbol{Z}}_3 \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} & \bar{\boldsymbol{Y}}^\top\boldsymbol{F}^\top\boldsymbol{\Delta}^\top\boldsymbol{H}^\top \\ * & \mathbf{0} & \mathbf{0} \\ * & * & \mathbf{0} \end{bmatrix} \succeq \mathbf{0}.
\end{aligned} \tag{4.9}
$$

The second term in (4.9) can be rewritten as

$$
\begin{aligned}
\begin{bmatrix} \mathbf{0} & \mathbf{0} & \bar{\boldsymbol{Y}}^\top\boldsymbol{F}^\top\boldsymbol{\Delta}^\top\boldsymbol{H}^\top \\ * & \mathbf{0} & \mathbf{0} \\ * & * & \mathbf{0} \end{bmatrix} &= \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \boldsymbol{H} \end{bmatrix} \boldsymbol{\Delta} \begin{bmatrix} \boldsymbol{F}\boldsymbol{Y} & \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \boldsymbol{Y}^\top\boldsymbol{F}^\top \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \boldsymbol{\Delta}^\top \begin{bmatrix} \mathbf{0} & \mathbf{0} & \boldsymbol{H}^\top \end{bmatrix} \\
&\succeq -\epsilon_2^{-1} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \boldsymbol{H} \end{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{0} & \boldsymbol{H}^\top \end{bmatrix} - \epsilon_2 \begin{bmatrix} \boldsymbol{Y}^\top\boldsymbol{F}^\top \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{F}\boldsymbol{Y} & \mathbf{0} & \mathbf{0} \end{bmatrix}
\end{aligned}
$$

for all $\epsilon_2 > 0$. Application of the Schur complement yields

$$
\begin{bmatrix} \boldsymbol{Q}_1\bar{\boldsymbol{R}}^{-1}\boldsymbol{Q}_1 & \mathbf{0} & \boldsymbol{Y}^\top\boldsymbol{B}^\top & \mathbf{0} & \epsilon_2\boldsymbol{Y}^\top\boldsymbol{F}^\top \\ * & \bar{\boldsymbol{Z}}_1 & \bar{\boldsymbol{Z}}_2 & \mathbf{0} & \mathbf{0} \\ * & * & \bar{\boldsymbol{Z}}_3 & \boldsymbol{H} & \mathbf{0} \\ * & * & * & \epsilon_2\boldsymbol{I} & \mathbf{0} \\ * & * & * & * & \epsilon_2\boldsymbol{I} \end{bmatrix} \succeq \mathbf{0}. \tag{4.10}
$$

To replace the nonlinear term $\boldsymbol{Q}_1\bar{\boldsymbol{R}}^{-1}\boldsymbol{Q}_1$, we make use of the fact that $\bar{\boldsymbol{R}} = \bar{\boldsymbol{R}}^\top \succ \mathbf{0}$

implies

$$\begin{aligned}(\boldsymbol{Q}_1 - \bar{\boldsymbol{R}})^\top \bar{\boldsymbol{R}}^{-1}(\boldsymbol{Q}_1 - \bar{\boldsymbol{R}}) = \boldsymbol{Q}_1^\top \bar{\boldsymbol{R}}^{-1}\boldsymbol{Q}_1 - 2\boldsymbol{Q}_1 + \bar{\boldsymbol{R}} \succ \boldsymbol{0}\\ \Rightarrow \boldsymbol{Q}_1^\top \bar{\boldsymbol{R}}^{-1}\boldsymbol{Q}_1 \succ 2\boldsymbol{Q}_1 - \bar{\boldsymbol{R}}.\end{aligned} \tag{4.11}$$

It follows from the application of the Schur complement that if (4.6) is satisfied, then (4.10) is also satisfied, which concludes the proof. □

Remark 4.2. *The application of Lemma 4.1 and (4.2) is done to convexify the problem for computational tractability, but it should be noted that these steps introduce some conservatism. Quantifying the degree of conservatism is generally difficult. One way to quantify at least the impact of (4.2) would be to first find a feasible solution for (4.6) and then gradually increase T_s until (4.10) is not satisfied anymore. However, the result would depend on the considered system. As an alternative to (4.2), we could also assume $\boldsymbol{Q}_1 = \alpha\bar{\boldsymbol{R}}$ for some $\alpha > 0$ similar to [19]. However, this would introduce another decision variable to the stability conditions, increasing the computational complexity of the optimization problems formulated in the following sections.*

Theorem 4.1 allows us to analyze the stability of the uncertain system (4.1) for a *given* value of T_s. In the following, we discuss the computation of the MCF based on Theorem 4.1.

4.3 Computing the Minimum Control Frequency

Recall that the MCF is defined as the maximum value of T_s such that there exists a controller (4.2) satisfying Assumption 2.2 that robustly stabilizes the system (4.1). With the sufficient stability conditions stated in (4.1), the computation of the MCF $f_{\mathrm{c,min}} = \frac{1}{T_{\mathrm{s,max}}}$ and the corresponding stabilizing control gain $\boldsymbol{K} = \boldsymbol{Y}\boldsymbol{Q}_1^{-1}$ can be formulated as the optimization problem

$$\begin{aligned}\min_{\substack{T_\mathrm{s},\boldsymbol{Q}_1,\boldsymbol{Q}_2,\boldsymbol{Q}_3,\bar{\boldsymbol{Z}}_1,\\ \bar{\boldsymbol{Z}}_2,\bar{\boldsymbol{Z}}_3,\bar{\boldsymbol{R}},\bar{\boldsymbol{Y}},\epsilon_1,\epsilon_2}} \quad & \frac{1}{T_\mathrm{s}}\\ \text{s.t.} \quad & \begin{bmatrix}\bar{\boldsymbol{Z}}_1 & \bar{\boldsymbol{Z}}_2 & \boldsymbol{Q}_2^\top & \boldsymbol{0} & \boldsymbol{0}\\ * & \bar{\boldsymbol{Z}}_3 & \boldsymbol{Q}_3^\top & \boldsymbol{0} & \boldsymbol{0}\\ * & * & -\bar{\boldsymbol{R}} & \boldsymbol{0} & \boldsymbol{0}\\ * & * & * & \boldsymbol{0} & \boldsymbol{0}\\ * & * & * & * & \boldsymbol{0}\end{bmatrix} \prec -\frac{1}{T_\mathrm{s}}\begin{bmatrix}\boldsymbol{Q}_2 + \boldsymbol{Q}_2^\top & \boldsymbol{\Phi}_{12} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{\Phi}_{15}\\ * & -\boldsymbol{Q}_3 - \boldsymbol{Q}_3^\top & \boldsymbol{0} & \boldsymbol{H} & \boldsymbol{0}\\ * & * & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0}\\ * & * & * & -\epsilon_1\boldsymbol{I} & \boldsymbol{0}\\ * & * & * & * & -\epsilon_1\boldsymbol{I}\end{bmatrix}\\ & (4.6),\\ & \boldsymbol{Q}_1 = \boldsymbol{Q}_1^\top \succ \boldsymbol{0}, \quad \bar{\boldsymbol{R}} = \bar{\boldsymbol{R}}^\top \succ \boldsymbol{0}, \quad \epsilon_1 > 0, \quad \epsilon_2 > 0,\end{aligned} \tag{4.12}$$

where $\boldsymbol{\Phi}_{12} = \boldsymbol{Q}_3 - \boldsymbol{Q}_2^\top + \boldsymbol{Q}_1\boldsymbol{A}^\top + \bar{\boldsymbol{Y}}^\top\boldsymbol{B}^\top$ and $\boldsymbol{\Phi}_{15} = \epsilon_1(\boldsymbol{Q}_1\boldsymbol{E}^\top + \bar{\boldsymbol{Y}}^\top\boldsymbol{F}^\top)$. If the values of ϵ_1 and ϵ_2 are fixed, the first matrix inequality in (4.12) becomes a linear fractional constraint of the form $\lambda\boldsymbol{M}(\boldsymbol{s}) + \boldsymbol{N}(\boldsymbol{s}) \prec \boldsymbol{0}$, where $\lambda \in \mathbb{R}$ and $\boldsymbol{s} \in \mathbb{R}^{n_s}$ are the optimization variables, and the matrices $\boldsymbol{M}$ and $\boldsymbol{N}$ depend affinely on $\boldsymbol{s}$. Then, (4.12) represents a GEVP similar to (2.6). As ϵ_1 and ϵ_2 must be kept fixed for (4.12) to be convex and thus easy to solve, the optimization over ϵ_1 and ϵ_2 has to be performed independently from the other optimization variables.

4.3.1 Simplifying the Optimization Problem

We can simplify the additional optimization over the scalar variables ϵ_1 and ϵ_2 by noting the following:

Lemma 4.2. *Let* $\boldsymbol{S}^* = (T_\mathrm{s}^*, \boldsymbol{Q}_1^*, \boldsymbol{Q}_2^*, \boldsymbol{Q}_3^*, \bar{\boldsymbol{Z}}_1^*, \bar{\boldsymbol{Z}}_2^*, \bar{\boldsymbol{Z}}_3^*, \bar{\boldsymbol{R}}^*, \bar{\boldsymbol{Y}}^*, \epsilon_1^*, \epsilon_2^*)$ *be an optimal solution to* (4.12)*. Then, for any* $c > 0$, $\boldsymbol{S}' = (T_\mathrm{s}^*, \frac{1}{c}\boldsymbol{Q}_1^*, \frac{1}{c}\boldsymbol{Q}_2^*, \frac{1}{c}\boldsymbol{Q}_3^*, \frac{1}{c}\bar{\boldsymbol{Z}}_1^*, \frac{1}{c}\bar{\boldsymbol{Z}}_2^*, \frac{1}{c}\bar{\boldsymbol{Z}}_3^*, \frac{1}{c}\bar{\boldsymbol{R}}^*, \frac{1}{c}\bar{\boldsymbol{Y}}^*, c\epsilon_1^*, c\epsilon_2^*)$ *is also an optimal solution to* (4.12)*.*

Proof. Both solutions $\boldsymbol{S}^*$ and $\boldsymbol{S}'$ yield the same objective value $\frac{1}{T_\mathrm{s}^*}$. Therefore, it remains to be shown that the feasibility of $\boldsymbol{S}^*$ implies the feasibility of $\boldsymbol{S}'$. We first consider the second matrix inequality constraint (4.6) of (4.12). If $\boldsymbol{S}$ is a feasible solution, then it holds that

$$\begin{bmatrix} 2\boldsymbol{Q}_1^* - \bar{\boldsymbol{R}}^* & \boldsymbol{0} & \bar{\boldsymbol{Y}}^{*\top}\boldsymbol{B}^\top & \boldsymbol{0} & \epsilon_2\boldsymbol{Y}^{*\top}\boldsymbol{F}^\top \\ * & \bar{\boldsymbol{Z}}_1^* & \bar{\boldsymbol{Z}}_2^* & \boldsymbol{0} & \boldsymbol{0} \\ * & * & \bar{\boldsymbol{Z}}_3^* & \boldsymbol{H} & \boldsymbol{0} \\ * & * & * & \epsilon_2\boldsymbol{I} & \boldsymbol{0} \\ * & * & * & * & \epsilon_2\boldsymbol{I} \end{bmatrix} \succeq \boldsymbol{0},$$

$$\Leftrightarrow \begin{bmatrix} 2\boldsymbol{Q}_1^* - \bar{\boldsymbol{R}}^* & \boldsymbol{0} & \bar{\boldsymbol{Y}}^{*\top}\boldsymbol{B}^\top \\ * & \bar{\boldsymbol{Z}}_1^* & \bar{\boldsymbol{Z}}_2^* \\ * & * & \bar{\boldsymbol{Z}}_3^* \end{bmatrix} - \epsilon_2^{-1} \begin{bmatrix} \boldsymbol{0} & \epsilon_2\bar{\boldsymbol{Y}}^{*\top}\boldsymbol{F}^\top \\ \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{H}^\top & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{H}^\top \\ \epsilon_2\boldsymbol{F}\bar{\boldsymbol{Y}}^\top & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \succeq \boldsymbol{0}, \qquad (4.13)$$

where we have applied the Schur complement. Replacing $\boldsymbol{S}^*$ by $\boldsymbol{S}'$ does not affect (4.13) except for a multiplication by a factor of $\frac{1}{c} > 0$. Proceeding in a similar way, the same property can be shown for the first matrix inequality constraint of (4.12), which concludes the proof. □

The two solutions $\boldsymbol{S}^*$ and $\boldsymbol{S}'$ also yield the same stabilizing control gain

$$\boldsymbol{K}^* = \boldsymbol{Y}^*(\boldsymbol{Q}_1^*)^{-1} = \frac{1}{c}\boldsymbol{Y}^*(\frac{1}{c}\boldsymbol{Q}_1^*)^{-1}. \qquad (4.14)$$

As a consequence of Lemma 4.2, we can reduce the number of scalar decision variables in the optimization problem (4.12) from two to one.

Corollary 4.1. *The optimization problem* (4.12) *has the same optimal solution as*

$$\begin{aligned}
&\min_{\substack{T_\mathrm{s},\boldsymbol{Q}_1,\boldsymbol{Q}_2,\boldsymbol{Q}_3,\bar{\boldsymbol{Z}}_1,\\ \bar{\boldsymbol{Z}}_2,\bar{\boldsymbol{Z}}_3,\bar{\boldsymbol{R}},\bar{\boldsymbol{Y}},\epsilon}} \quad \frac{1}{T_\mathrm{s}}\\
&\text{s.t.} \quad \begin{bmatrix} \bar{\boldsymbol{Z}}_1 & \bar{\boldsymbol{Z}}_2 & \boldsymbol{Q}_2^\top & \mathbf{0} & \mathbf{0}\\ * & \bar{\boldsymbol{Z}}_3 & \boldsymbol{Q}_3^\top & \mathbf{0} & \mathbf{0}\\ * & * & -\bar{\boldsymbol{R}} & \mathbf{0} & \mathbf{0}\\ * & * & * & \mathbf{0} & \mathbf{0}\\ * & * & * & * & \mathbf{0} \end{bmatrix} \prec -\frac{1}{T_\mathrm{s}} \begin{bmatrix} \boldsymbol{Q}_2+\boldsymbol{Q}_2^\top & \boldsymbol{\Phi}_{12} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Phi}_{15}\\ * & -\boldsymbol{Q}_3-\boldsymbol{Q}_3^\top & \mathbf{0} & \boldsymbol{H} & \mathbf{0}\\ * & * & \mathbf{0} & \mathbf{0} & \mathbf{0}\\ * & * & * & -\epsilon\boldsymbol{I} & \mathbf{0}\\ * & * & * & * & -\epsilon\boldsymbol{I} \end{bmatrix}\\
&\qquad \begin{bmatrix} 2\boldsymbol{Q}_1-\bar{\boldsymbol{R}} & \mathbf{0} & \bar{\boldsymbol{Y}}^\top\boldsymbol{B}^\top & \mathbf{0} & \epsilon^{-1}\boldsymbol{Y}^\top\boldsymbol{F}^\top\\ * & \bar{\boldsymbol{Z}}_1 & \bar{\boldsymbol{Z}}_2 & \mathbf{0} & \mathbf{0}\\ * & * & \bar{\boldsymbol{Z}}_3 & \boldsymbol{H} & \mathbf{0}\\ * & * & * & \epsilon^{-1}\boldsymbol{I} & \mathbf{0}\\ * & * & * & * & \epsilon^{-1}\boldsymbol{I} \end{bmatrix} \succeq \mathbf{0},\\
&\qquad \boldsymbol{Q}_1=\boldsymbol{Q}_1^\top \succ \mathbf{0}, \quad \bar{\boldsymbol{R}}=\bar{\boldsymbol{R}}^\top \succ \mathbf{0}, \quad \epsilon > 0.
\end{aligned} \tag{4.15}$$

Proof. Lemma 4.2 states that the set of optimal solutions to (4.12) is not a singleton, but of the form

$$\mathcal{S}^* = \left\{ \left(T_\mathrm{s}^*, \frac{1}{c}\boldsymbol{Q}_1^*, \frac{1}{c}\boldsymbol{Q}_2^*, \frac{1}{c}\boldsymbol{Q}_3^*, \frac{1}{c}\bar{\boldsymbol{Z}}_1^*, \frac{1}{c}\bar{\boldsymbol{Z}}_2^*, \frac{1}{c}\bar{\boldsymbol{Z}}_3^*, \frac{1}{c}\bar{\boldsymbol{R}}^*, \frac{1}{c}\bar{\boldsymbol{Y}}^*, c\epsilon_1^*, c\epsilon_2^* \right) \,\middle|\, c \in \mathbb{R}^+ \right\}.$$

For the proof, we only consider the optimization with respect to the scalar optimization variables ϵ_1 and ϵ_2, respectively ϵ, as (4.12) and (4.15) only differ in those and are otherwise convex. We denote projection of $\mathcal{S}^*$ onto the last two dimensions by $\mathcal{S}^*_{\epsilon_1,\epsilon_2} = \{(c\epsilon_1^*, c\epsilon_2^*) \mid c > 0\}$. Setting $c = \frac{1}{\sqrt{\epsilon_1^*\epsilon_2^*}}$ shows that $\boldsymbol{a}^* = \left(\sqrt{\frac{\epsilon_1^*}{\epsilon_2^*}}, \sqrt{\frac{\epsilon_2^*}{\epsilon_1^*}}\right) \in \mathcal{S}^*_{\epsilon_1,\epsilon_2}$. Looking at the matrix inequality constraints in (4.15), we see that $\epsilon^* = \sqrt{\frac{\epsilon_1^*}{\epsilon_2^*}}$ corresponds to the optimal value of ϵ in (4.15). Fig. 4.1 illustrates the idea of the proof. Graphically speaking, Lemma 4.2 states that moving along straight lines through the origin in the (ϵ_1, ϵ_2)-plane does not change the solution to (4.12). Therefore, the intersection of the set $\mathcal{E} = \left\{(\epsilon_1,\epsilon_2) \in \mathbb{R}^+ \times \mathbb{R}^+ \,\middle|\, \epsilon_2 = \frac{1}{\epsilon_1}\right\}$ with $\mathcal{S}^*_{\epsilon_1,\epsilon_2}$ is nonzero, and (4.15) yields the same optimal solution as (4.12), which concludes the proof. □

The total number of decision variables in the optimization problem (4.15) can be calculated as

$$1 + \frac{n(n+1)}{2} + n^2 + n^2 + n^2 + n^2 + n^2 + \frac{n(n+1)}{2} + nm + 1 = 6n^2 + nm + n + 2.$$

Remark 4.3. *In principle, other choices for the set* $\mathcal{E}$, *such as the straight line* $\mathcal{E}_\mathrm{l} = \{(\epsilon_1,\epsilon_2) \in \mathbb{R}^+ \times \mathbb{R}^+ \mid \epsilon_2 = 1 - \epsilon_1\}$, *are also possible. However, the grid*

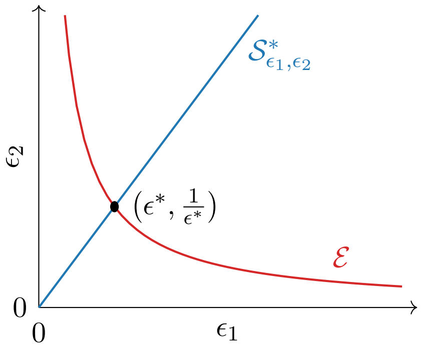

Figure 4.1: Illustration of the equivalence of the optimal solutions to (4.12) and (4.15). Restricting the set over which ϵ_1 and ϵ_2 are optimized from $\mathbb{R}^+ \times \mathbb{R}^+$ to $\mathcal{E}$ does not change the optimal solution.

search performed over ϵ should take the shape of $\mathcal{E}$ into account. For example, a (coarse) linear grid over the straight line $\mathcal{E}_1$ might not contain solutions where one of the two variables ϵ_1 and ϵ_2 is orders of magnitude larger than the other, i.e., would be unsuitable if $\mathcal{S}^*_{\epsilon_1,\epsilon_2}$ has a very flat or steep slope.

4.4 Performance Optimization

In addition to stability, we also want to study the impact of the control frequency on performance. To this end, we aim to optimize the system performance for a particular sampling time $T_\mathrm{s} \in (0, T_{\mathrm{s,max}}]$ and consider the minimization of the cost function [40]

$$J = \int_{\tau_0}^{\tau_1} \boldsymbol{x}(t_k)^\top \boldsymbol{Q}_J \boldsymbol{x}(t_k) + \boldsymbol{u}(t)^\top \boldsymbol{R}_J \boldsymbol{u}(t) \mathrm{d}t_k, \tag{4.16}$$

where $\tau_1 - \tau_0 > 0$ is the optimization period, and $\boldsymbol{Q}_J \succ \boldsymbol{0}$ and $\boldsymbol{R}_J \succ \boldsymbol{0}$ are weight matrices. Our goal is to minimize (4.16) while ensuring that the system is stable. By inserting the control law (4.2) into (4.16), we can rewrite the cost as

$$J = \int_{\tau_0}^{\tau_1} \boldsymbol{x}(t_k)^\top \left(\boldsymbol{Q}_J + \boldsymbol{K}^\top \boldsymbol{R}_J \boldsymbol{K}\right) \boldsymbol{x}(t_k) \mathrm{d}t_k. \tag{4.17}$$

To make the problem tractable, let

$$J \leq \bar{J} = \eta \int_{\tau_0}^{\tau_1} \boldsymbol{x}(t_k)^\top \boldsymbol{Q}_1^{-1} \boldsymbol{Q}_1^{-1} \boldsymbol{x}(t_k) \mathrm{d}t_k. \tag{4.18}$$

Then, we can replace the minimization of (4.16) by minimizing η and enforcing the bound (4.18), which can be written as

$$\int_{\tau_0}^{\tau_1} \boldsymbol{x}(t_k)^\top \boldsymbol{Q}_1^{-1} \Big(\boldsymbol{Q}_1 \boldsymbol{Q}_J \boldsymbol{Q}_1 + \boldsymbol{Q}_1 \boldsymbol{K}^\top \boldsymbol{R}_J \boldsymbol{K} \boldsymbol{Q}_1 - \eta \boldsymbol{I}\Big) \boldsymbol{Q}_1^{-1} \boldsymbol{x}(t_k) \mathrm{d}t_k \leq 0. \tag{4.19}$$

Using again the substitution $\boldsymbol{K} = \boldsymbol{Y}\boldsymbol{Q}_1^{-1}$ gives

$$\int_{\tau_0}^{\tau_1} \boldsymbol{x}(t_k)^\top \boldsymbol{Q}_1^{-1}\Big(\boldsymbol{Q}_1\boldsymbol{Q}_J\boldsymbol{Q}_1 + \boldsymbol{Y}^\top \boldsymbol{R}_J \boldsymbol{Y} - \eta \boldsymbol{I}\Big)\boldsymbol{Q}_1^{-1}\boldsymbol{x}(t_k)\mathrm{d}t_k \leq 0. \tag{4.20}$$

This inequality is satisfied for all optimization periods $\tau_1 - \tau_0$ if

$$\boldsymbol{Q}_1\boldsymbol{Q}_J\boldsymbol{Q}_1 + \boldsymbol{Y}^\top \boldsymbol{R}_J \boldsymbol{Y} - \eta \boldsymbol{I} \preceq \boldsymbol{0}. \tag{4.21}$$

We employ the Schur complement to rewrite the nonlinear matrix inequality (4.21) as an LMI

$$\begin{bmatrix} -\eta\boldsymbol{I} & \boldsymbol{Q}_1 & \boldsymbol{Y}^\top \\ * & -\boldsymbol{Q}_J^{-1} & \boldsymbol{0} \\ * & * & -\boldsymbol{R}_J^{-1} \end{bmatrix} \preceq \boldsymbol{0}. \tag{4.22}$$

Thus, optimization of the system performance can be achieved by solving

$$\begin{aligned} \min_{\substack{\eta,\boldsymbol{Q}_1,\boldsymbol{Q}_2,\boldsymbol{Q}_3,\bar{\boldsymbol{Z}}_1,\\ \bar{\boldsymbol{Z}}_2,\bar{\boldsymbol{Z}}_3,\bar{\boldsymbol{R}},\boldsymbol{Y},\epsilon_1,\epsilon_2}} \quad & \eta \\ \text{s.t.} \quad & \begin{bmatrix} \boldsymbol{0} & \boldsymbol{Q}_1 & \boldsymbol{Y}^\top \\ * & -\boldsymbol{Q}_J^{-1} & \boldsymbol{0} \\ * & * & -\boldsymbol{R}_J^{-1} \end{bmatrix} \preceq -\eta \begin{bmatrix} -\boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ * & \boldsymbol{0} & \boldsymbol{0} \\ * & * & \boldsymbol{0} \end{bmatrix} \\ & (4.5), \quad (4.6) \\ & \boldsymbol{Q}_1 = \boldsymbol{Q}_1^\top \succ \boldsymbol{0}, \quad \bar{\boldsymbol{R}} = \bar{\boldsymbol{R}}^\top \succ \boldsymbol{0}, \quad \epsilon_1 > 0, \quad \epsilon_2 > 0. \end{aligned} \tag{4.23}$$

The optimization problem (4.23) has a similar structure to (4.12), being a convex GEVP for fixed values of ϵ_1 and ϵ_2. However, as $\boldsymbol{Q}_1$ and $\boldsymbol{Y}$ are directly related to the objective value through (4.22), the optimization problem (4.23) does not allow for a reduction of the number of decision variables like (4.12). Thus, (4.23) has $6n^2 + nm + n + 3$ decision variables.

Remark 4.4. *The upper bound $\bar{J}$ on the cost J depends not only on η, but also on the optimization variable $\boldsymbol{Q}_1$. Consequently, the solution to* (4.23) *is not guaranteed to minimize $\bar{J}$, and thus J, in general. Nevertheless, the optimization problem* (4.23) *represents a computationally tractable way to improve the performance of the robust sampled-data controller compared to the conservative solution of* (4.12)*, which only considers stability.*

4.5 Linking the Control Frequency to Data

We now take the opposite perspective to Section 4.3 and aim to calculate the amount of data required to be able to stabilize an uncertain system at a given frequency. This problem is generally harder to solve than (4.1), and we therefore have to make a simplifying assumption compared to our previous setting.

Assumption 4.1. *The input matrix of the linearized system is known, i.e., the true linearized dynamics are captured with probability at least $p \in [0, 1)$ by*

$$\boldsymbol{x}(t) = (\boldsymbol{A} + \boldsymbol{H}_1 \boldsymbol{\Delta}_1 \bar{\boldsymbol{E}})\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t), \tag{4.24}$$

where $\boldsymbol{H}_1$, $\boldsymbol{\Delta}_1$, and $\bar{\boldsymbol{E}}$ are defined by (3.37), (3.39), and (3.41), respectively.

4.5.1 Uncertainty as a Disturbance

We can define the uncertain component of the dynamics (4.24) as an additive disturbance

$$\boldsymbol{w}(t) = \boldsymbol{H}_1 \boldsymbol{\Delta}_1 \bar{\boldsymbol{E}} \boldsymbol{x}(t) \tag{4.25}$$

and rewrite (4.24) as

$$\boldsymbol{x}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) + \boldsymbol{w}(t). \tag{4.26}$$

The disturbance $\boldsymbol{w}(t)$ can be upper bounded as follows.

Lemma 4.3. *The disturbance $\boldsymbol{w}(t)$ defined in (4.25) satisfies the upper bound*

$$\|\boldsymbol{w}(t)\|_2^2 \leq n\|\hat{\boldsymbol{A}}\|_{1,2}^2 \|\boldsymbol{x}(t)\|_2^2, \tag{4.27}$$

where $\|\cdot\|_{1,2}$ is a matrix norm induced by the vector one- and Euclidean norm, which is defined for any matrix $\boldsymbol{D} = \begin{bmatrix} \boldsymbol{d}_1 & \dots & \boldsymbol{d}_{n_c} \end{bmatrix}$ as

$$\|\boldsymbol{D}\|_{1,2} = \max_i \|\boldsymbol{d}_i\|_2. \tag{4.28}$$

Proof. For any matrix $\boldsymbol{D}$ and vector $\boldsymbol{d}$ of appropriate dimensions, the spectral norm satisfies $\|\boldsymbol{D}\boldsymbol{d}\|_2 \leq \|\boldsymbol{D}\|_2 \|\boldsymbol{d}\|_2$. Consequently,

$$\|\boldsymbol{w}(t)\|_2^2 = \|\boldsymbol{H}_1 \boldsymbol{\Delta}_1 \bar{\boldsymbol{E}} \boldsymbol{x}(t)\|_2^2 \leq \|\boldsymbol{H}_1\|_2^2 \|\boldsymbol{\Delta}_1\|_2^2 \|\bar{\boldsymbol{E}}\|_2^2 \|\boldsymbol{x}(t)\|_2^2. \tag{4.29}$$

We proceed by bounding the three spectral norm terms in (4.29) from above. Using the expression (3.37) for $\boldsymbol{H}_1$, we obtain

$$\|\boldsymbol{H}_1\|_2^2 = \lambda_{\max}\left(\boldsymbol{H}_1^\top \boldsymbol{H}_1\right) = \lambda_{\max}\left(\begin{bmatrix} \boldsymbol{1}_{n\times n} & \boldsymbol{0}_{n\times n} & \dots & \boldsymbol{0}_{n\times n} \\ \boldsymbol{0}_{n\times n} & \boldsymbol{1}_{n\times n} & \dots & \boldsymbol{0}_{n\times n} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0}_{n\times n} & \boldsymbol{0}_{n\times n} & \dots & \boldsymbol{1}_{n\times n} \end{bmatrix}\right) = \lambda_{\max}(\boldsymbol{1}_{n\times n}) = n.$$

It directly follows from (3.39) that $\|\boldsymbol{\Delta}_1\|_2 \leq 1$. To bound the third term in (4.29), we calculate

$$\begin{aligned}\bar{\boldsymbol{E}}^\top\bar{\boldsymbol{E}} &= \begin{bmatrix}\mathrm{Diag}(\hat{\boldsymbol{a}}_1) & \dots & \mathrm{Diag}(\hat{\boldsymbol{a}}_n)\end{bmatrix}\begin{bmatrix}\mathrm{Diag}(\hat{\boldsymbol{a}}_1)\\ \vdots \\ \mathrm{Diag}(\hat{\boldsymbol{a}}_n)\end{bmatrix}\\ &= \sum_{i=1}^{n}\mathrm{Diag}(\hat{\boldsymbol{a}}_i)^2\\ &= \begin{bmatrix}\sum_{i=1}^N \hat{a}_{i1}^2 & 0 & \dots & 0\\ 0 & \sum_{i=1}^N \hat{a}_{i2}^2 & \dots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \dots & \sum_{i=1}^N \hat{a}_{in}^2\end{bmatrix}.\end{aligned}$$

Since $\bar{\boldsymbol{E}}^\top\bar{\boldsymbol{E}}$ is a diagonal matrix, the spectral norm of $\bar{\boldsymbol{E}}$ is simply given by

$$\|\bar{\boldsymbol{E}}\|_2^2 = \lambda_{\max}(\bar{\boldsymbol{E}}^\top\bar{\boldsymbol{E}}) = \max_{j\in\{1,\dots,n\}}\sum_{i=1}^{N}\hat{a}_{ij}^2 = \|\hat{\boldsymbol{A}}\|_{1,2}^2.$$

Here, $\|\hat{\boldsymbol{A}}\|_{1,2}$ is an induced matrix norm, which corresponds to the maximum Euclidean norm of the columns of $\hat{\boldsymbol{A}}$. The proof is concluded by inserting the derived expressions for $\|\boldsymbol{H}_1\|_2^2$, $\|\boldsymbol{\Delta}_1\|_2^2$ and $\|\bar{\boldsymbol{E}}\|_2^2$ into (4.29). □

In [41], the stability of (4.26) is analyzed using an LKF approach based on [19]. For this, the disturbance is assumed to be bounded by

$$\boldsymbol{w}(t)^\top\boldsymbol{w}(t) \leq \alpha^2\|\tilde{\boldsymbol{H}}\boldsymbol{x}(t)\|_2^2, \tag{4.30}$$

where $\alpha > 0$ is the bounding parameter and $\tilde{\boldsymbol{H}}$ is a constant matrix with n columns. This corresponds to the bound (4.27) with $\alpha = \|\hat{\boldsymbol{A}}_{1,2}\|$ and $\tilde{\boldsymbol{H}} = n\boldsymbol{I}$. Hence, we can apply the stability conditions derived in [41] to (4.26) and thus to (4.24) to calculate the maximum admissible disturbance magnitude.

Lemma 4.4 (Adopted from [41], Theorem 3)**.** *Given a scalar $T_\mathrm{s} > 0$, if the GEVP*

$$\begin{aligned}
\min_{\substack{\boldsymbol{Q}_1,\boldsymbol{Q}_2,\boldsymbol{Q}_3,\bar{\boldsymbol{Z}}_1,\\ \bar{\boldsymbol{Z}}_2,\bar{\boldsymbol{Z}}_3,\bar{\boldsymbol{R}},\boldsymbol{Y},\gamma}} \quad & \gamma \\
\text{s.t.} \quad & \boldsymbol{Q}_1 = \boldsymbol{Q}_1^\top \succ \boldsymbol{0} \\
& \begin{bmatrix} \boldsymbol{Q}_2 + \boldsymbol{Q}_2^\top + T_\mathrm{s}\bar{\boldsymbol{Z}}_1 & \boldsymbol{\Phi}_{12} & \boldsymbol{0} & T_\mathrm{s}\boldsymbol{Q}_2^\top & n\boldsymbol{Q}_1 \\ * & -\boldsymbol{Q}_3^\top - \boldsymbol{Q}_3 + T_\mathrm{s}\bar{\boldsymbol{Z}}_3 & \boldsymbol{I} & T_\mathrm{s}\boldsymbol{Q}_3^\top & \boldsymbol{0} \\ * & * & -\boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ * & * & * & -T_\mathrm{s}\bar{\boldsymbol{R}} & \boldsymbol{0} \\ * & * & * & * & -\gamma\boldsymbol{I} \end{bmatrix} \prec \boldsymbol{0} \qquad (4.31) \\
& \begin{bmatrix} 2\boldsymbol{Q}_1 - \bar{\boldsymbol{R}} & \boldsymbol{0} & \bar{\boldsymbol{Y}}^\top\boldsymbol{B}^\top \\ * & \bar{\boldsymbol{Z}}_1 & \bar{\boldsymbol{Z}}_2 \\ * & * & \bar{\boldsymbol{Z}}_3 \end{bmatrix} \succeq \boldsymbol{0},
\end{aligned}$$

where $\boldsymbol{\Phi}_{12} = \boldsymbol{Q}_3 - \boldsymbol{Q}_2^\top + \boldsymbol{Q}_1\boldsymbol{A}^\top + \bar{\boldsymbol{Y}}^\top\boldsymbol{B}^\top + T_\mathrm{s}\bar{\boldsymbol{Z}}_2$, is feasible with optimal value $\gamma^*(T_\mathrm{s})$, then the control law (4.2) with control gain $\boldsymbol{K} = \boldsymbol{Y}\boldsymbol{Q}_1^{-1}$ robustly stabilizes the system (4.24) for $\|\hat{\boldsymbol{A}}_{1,2}\| \leq \alpha^*(T_\mathrm{s}) = \frac{1}{\sqrt{\gamma^*(T_\mathrm{s})}}$.

4.5.2 Minimum Amount of Data for Stability at a Given Control Frequency

We consider the case that the uncertain model is obtained from data using BLR, as described in Section 3.2. We are interested in the minimum amount of training samples such that the uncertain system (4.24) can be stabilized at a given control frequency $\frac{1}{T_\mathrm{s}}$ under Assumption 2.2.

Theorem 4.2. *Let $\alpha^*(T_\mathrm{s})$ be the maximum admissible disturbance magnitude obtained by solving (4.31) for a given T_s, and let $p \in [0, 1)$ be the desired confidence threshold. Then, with a probability of at least p, the number of training samples in the data set (3.3) required to guarantee stability via Lemma 4.4 at the frequency $f_\mathrm{c} = \frac{1}{T_\mathrm{s}}$ satisfies the bound*

$$N \geq N_\mathrm{min} = \max_{i \in \mathbb{I}_n} \frac{\sigma^2_{\mathrm{n},i}(\chi^2_n(\sqrt[n]{p})) - (\alpha^*(T_\mathrm{s}))^2 n^{\frac{5}{2}} \sigma^2_{\mathrm{n},i} \|\boldsymbol{\Sigma}^{-1}_{i,0}\|_2}{r(\alpha^*(T_\mathrm{s}))^2 n^{\frac{5}{2}}}, \qquad (4.32)$$

where r captures the size of the training input set and is defined in (3.22).

Before proceeding with the proof of Theorem 4.2, we present three results that are used in the proof.

Lemma 4.5 ([42])**.** *For any vector $\boldsymbol{d} \in \mathbb{R}^n$, the one-norm, Euclidean norm and infinity norm satisfy the inequality*

$$\|\boldsymbol{d}\|_\infty \leq \|\boldsymbol{d}\|_2 \leq \|\boldsymbol{d}\|_1 \leq \sqrt{n_d}\|\boldsymbol{d}\|_2 \leq n_d\|\boldsymbol{d}\|_\infty. \qquad (4.33)$$

Lemma 4.6. *Let* $\boldsymbol{D} \in \mathbb{R}^{n\times n}$ *be a positive definite matrix and* $\boldsymbol{d} = \operatorname{diag}(\boldsymbol{D}) \in \mathbb{R}^n$*. Then,*

$$\|\sqrt{\boldsymbol{d}}\|_2 \geq \frac{1}{n^{\frac{3}{4}}}\sqrt{\|\boldsymbol{D}\|_2}. \tag{4.34}$$

Proof. We denote $\boldsymbol{d} = [d_1, \dots, d_n]^\top$ and define the matrix $\boldsymbol{D}_\mathrm{d} = \operatorname{Diag}(\boldsymbol{d}) \in \mathbb{R}^{n\times n}$, which only contains the diagonal elements of $\boldsymbol{D}$. Since $\boldsymbol{D}$ is positive definite, we have $d_i > 0$, $\forall i \in \mathbb{I}_n$. Let $j = \arg\max_i\, d_i$. It is straightforward to prove that $d_j = \max_{i,k}\, |D_{ik}|$ as otherwise $\boldsymbol{D}$ would not be positive definite. Consequently, for any $\boldsymbol{v} \in \mathbb{R}^n$, we have

$$\begin{aligned}
\|\boldsymbol{D}\boldsymbol{v}\|_2 &\leq \sqrt{n}\max_i\, |(\boldsymbol{D}\boldsymbol{v})_i| \\
&= \sqrt{n}\max_i \sum_{k=1}^{n} |D_{ik}v_k| \\
&\leq \sqrt{n}\max_i \sum_{k=1}^{n} |D_{ik}| \underbrace{\max_k\, |v_k|}_{=\|\boldsymbol{v}\|_\infty \leq \|\boldsymbol{v}\|_2} \\
&\leq \sqrt{n}\|\boldsymbol{v}\|_2 n \max_{i,k}\, |D_{ik}| \\
&= n\sqrt{n}\|\boldsymbol{v}\|_2 d_j,
\end{aligned}$$

where the first and the third inequality follow from Lemma 4.5. As a result,

$$\|\boldsymbol{D}\|_2 = \max_{\|\boldsymbol{v}\|_2 \leq 1}\, \|\boldsymbol{D}\boldsymbol{v}\|_2 \leq n\sqrt{n}d_j.$$

Moreover, we have

$$\|\sqrt{\boldsymbol{d}}\|_2^2 = \left(\sqrt{d_1}\right)^2 + \ldots + \left(\sqrt{d_n}\right)^2 = \|\boldsymbol{d}\|_1 \geq \|\boldsymbol{d}\|_2 \geq d_j \geq \frac{1}{n^{\frac{3}{2}}}\|\boldsymbol{D}\|_2, \tag{4.35}$$

where the first inequality is due to Lemma 4.5. The result follows by applying the square root to (4.35). □

Lemma 4.7. *Let* $\boldsymbol{D} = \left[\boldsymbol{d}_{r,1}\ \dots\ \boldsymbol{d}_{r,n}\right]^\top \in \mathbb{R}^{n\times n}$*. Then,*

$$\|\boldsymbol{D}\|_{1,2} \geq \frac{1}{\sqrt{n}}\max_i\, \|\boldsymbol{d}_{r,i}\|_2. \tag{4.36}$$

Proof. We denote

$$\boldsymbol{D} = \begin{bmatrix} D_{11} & \dots & D_{1n} \\ \vdots & \ddots & \vdots \\ D_{n1} & \dots & D_{nn} \end{bmatrix} = \begin{bmatrix} \boldsymbol{d}_{r,1}^\top \\ \vdots \\ \boldsymbol{d}_{r,n}^\top \end{bmatrix} = \begin{bmatrix} \boldsymbol{d}_{c,1} & \dots & \boldsymbol{d}_{c,n} \end{bmatrix}.$$

Then,

$$\left\|\boldsymbol{D}\right\|_{1,2} = \max_i \left\|\boldsymbol{d}_{c,i}\right\|_2 \geq \max_{i,j} |D_{ij}| = \max_i \|\boldsymbol{d}_{r,i}\|_\infty \geq \frac{1}{\sqrt{n}} \max_i \left\|\boldsymbol{d}_{r,i}\right\|_2,$$

where the second inequality follows from the last inequality in Lemma 4.5. □

Proof of Theorem 4.2. We prove this by contradiction. Assume that (4.32) does not hold, i.e., $N < N_{\min}$. Then, we have

$$\left\|\boldsymbol{\Sigma}_{\boldsymbol{\theta}_i}\right\|_2 > \frac{(\alpha^*(T_\mathrm{s}))^2 n^{\frac{5}{2}}}{(\chi_n^2(\sqrt[n]{p}))^2}, \qquad \forall i \in \mathbb{I}_n, \tag{4.37}$$

due to Lemma 3.1. Since a covariance matrix is always positive definite, we can apply Lemma 4.6 to show that (4.37) implies

$$\left\|\sqrt{\operatorname{diag}(\boldsymbol{\Sigma}_{\boldsymbol{\theta}_i})}\right\|_2 > \frac{\alpha^*(T_\mathrm{s})\sqrt{n}}{\chi_n^2(\sqrt[n]{p})}, \qquad \forall i \in \mathbb{I}_n.$$

Then, due to Lemma 4.7 and the definition of $\hat{\boldsymbol{A}}$ in (3.15), we have

$$\|\hat{\boldsymbol{A}}\|_{1,2} \geq \frac{1}{\sqrt{n}} \chi_n^2(\sqrt[n]{p}) \max_i \left\|\sqrt{\operatorname{diag}(\boldsymbol{\Sigma}_{\boldsymbol{\theta}_i})}\right\|_2 > \alpha^*(T_\mathrm{s}).$$

Hence, we cannot guarantee stability via Lemma 4.4 if (4.32) is not satisfied, which concludes the proof. □

Although Theorem 4.2 only holds under certain rather restrictive assumptions, it shows which quantities affect how many data points must at least be collected for stability at a certain control frequency:

- $N_{\min}$ is directly proportional to the observation noise variance and increases with the confidence threshold p.
- $N_{\min}$ decreases with the size r of the set from which the training inputs are drawn, with the norm of the prior precision matrices $\boldsymbol{\Sigma}_{i,0}^{-1}$ and with the maximum admissible disturbance $\alpha^*(T_\mathrm{s})$, i.e., the control frequency.

4.6 Extension to Online Learning and Impact on Stability

Since the methods presented so far in this chapter allow us to relate model learning to the control frequency, naturally, the question arises whether we could change the control frequency during operation when the learned model is updated online. In practice, the computation time for solving GEVPs (4.12) and (4.23) would be an obstacle, and the optimization problems would likely have to be reformulated such that they are somewhat independent of the learned model itself and can be solved offline. However, the first question we have to ask in this context is whether the stability of the system would be affected by switching the control frequency and the control gain.

Assume that we have a finite collection of control frequencies $f_{\mathrm{c},i} = \frac{1}{T_{\mathrm{s},i}}$ and corresponding controllers $\boldsymbol{K}_i$, $i = 0, \dots, M$, where each pair $(T_{\mathrm{s},i}, \boldsymbol{K}_i)$ is via Theorem 4.1 individually guaranteed to stabilize the system (4.3). At time zero, we set $T_\mathrm{s} = T_{\mathrm{s},0}$ and $\boldsymbol{K} = \boldsymbol{K}_0$. The control frequency and the control gain are changed online at time instants T_i, $i = 1, \dots, M$, i.e., the dynamics in phase $i \in \mathbb{I}_M^0$ are

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{K}_i\boldsymbol{x}(t_k), \qquad t \in [t_k, t_{k+1}) \subseteq [T_i, T_{i+1}), \tag{4.38}$$

where $t_0 = T_0 = 0$. During phase i, i.e., for $t_k \in [T_i, T_{i+1})$, we have $t_{k+1} - t_k \leq T_{\mathrm{s},i}$. We assume for simplicity that the control frequency and the controller are only changed at sampling instants, i.e., $T_i = t_k$ for some $k \in \mathbb{N}_0$ for all $i \in \mathbb{I}_M$.

We denote the LKF corresponding to the i-th phase by

$$V_i(t) = \boldsymbol{x}(t)^\top \boldsymbol{P}_{1,i}\boldsymbol{x}(t) + \int_{-T_\mathrm{s}}^{0}\int_{t+\theta}^{t} \boldsymbol{y}(s)^\top \boldsymbol{R}_i \boldsymbol{y}(s)\mathrm{d}s\mathrm{d}\theta, \tag{4.39}$$

where $\boldsymbol{P}_{1,i} \succ \boldsymbol{0}$ and $\boldsymbol{R}_i \succ \boldsymbol{0}$. As pointed out in Section 2.3, the LKF (4.39) satisfies

$$a_i\|\boldsymbol{x}(t)\|_2^2 \leq V_i(t) \leq b_i \sup_{s\in[-T_\mathrm{s},0]} \|\bar{\boldsymbol{x}}(t+s)\|_2^2, \tag{4.40}$$

for some $a_i > 0$ and $b_i > 0$. Applying the stable combination of control gain $\boldsymbol{K}_i$ and sampling time $T_{\mathrm{s},i}$ implies that

$$\begin{aligned} \dot{V}_i(t) &\leq 0, \qquad \forall t \in [T_i, T_{i+1}), \quad \forall i \in \mathbb{I}_{M-1}, \\ \dot{V}_M(t) &\leq 0, \qquad \forall t \geq T_M, \end{aligned} \tag{4.41}$$

with equality holding only for $\bar{\boldsymbol{x}}(t) = \boldsymbol{0}$. To analyze asymptotic stability of the system for the sequence of different control frequencies and sampling times, we consider an augmented Lyapunov function $V(t)$ defined piecewise as

$$\begin{aligned} V(t) &= V_i'(t), \qquad \forall t \in [T_i, T_{i+1}), \quad \forall i \in \mathbb{I}_{M-1}, \\ V(t) &= V_M'(t), \qquad \forall t \geq T_M, \end{aligned} \tag{4.42}$$

where

$$\begin{aligned} V_0'(t) &= V_0(t), \\ V_i'(t) &= \gamma_i V_i(t), \qquad \gamma_i = \frac{V_{i-1}'(T_i)}{V_i(T_i)} > 0, \quad \forall i \in \mathbb{I}_M. \end{aligned} \tag{4.43}$$

Fig. 4.2 visualizes the construction of V. The idea is to scale the LKFs $V_1, \dots, V_M$ by factors γ_i, $i = 1, \dots, M$, such that the augmented LKF is continuous. Let us verify

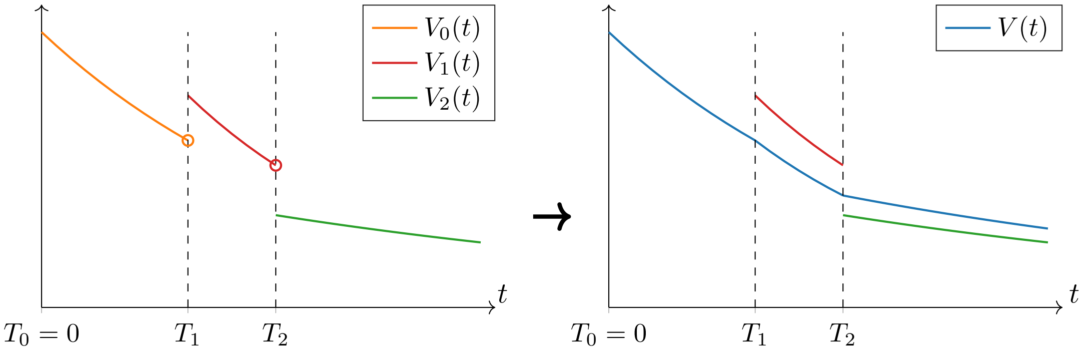

Figure 4.2: Construction of the LKF for proving asymptotic stability for updating the control frequency and the control gain online.

that V is indeed a valid LKF. By design, it satisfies the inequalities

$$a\|\boldsymbol{x}(t)\|_2^2 \leq V(t) \leq b \sup_{s\in[-\bar{T}_\mathrm{s},0]} \|\bar{\boldsymbol{x}}(t+s)\|_2^2, \tag{4.44}$$

where $a = \min\{a_0, \gamma_1 a_1, \ldots, \gamma_M a_M\} > 0$, $b = \max\{b_0, \gamma_1 b_1, \ldots, \gamma_M b_M\} > 0$, and $\bar{T}_\mathrm{s} = \max\{T_{\mathrm{s},0}, \ldots, T_{\mathrm{s,M}}\} > 0$, which are similar to (2.44). It follows from (4.42) and (4.43) that $\dot{V}(t) < 0$ for all $\bar{\boldsymbol{x}}(t) \neq 0$, which proves the asymptotic stability of the system under the considered sequence of different control frequencies and controllers. This insight can serve as a motivation to develop a scheme for changing the control frequency online based on updating the learned model.

Chapter 5

Evaluation

We use our proposed learning-based sampled-data control approach to study the impact of control frequency and data on stability and closed-loop performance in simulation and real-world experiments. The chapter concludes with a comparative discussion of the obtained results.

5.1 Simulation

In this section, we apply the methods discussed in Chapters 3 and 4 to a simulated nonlinear quadrotor system. We first introduce the quadrotor's dynamics model and our simulation setup, then present our results and conclude with a brief discussion.

5.1.1 2D Quadrotor Model

We consider a quadrotor flying in the vertical x-z-plane, as illustrated in Fig. 5.1. The position and velocity of its center of mass (CoM) are denoted by (x, z) and $(\dot{x}, \dot{z})$, respectively, θ denotes the pitch angle and $\dot{\theta}$ its rate of change. As inputs, we consider the thrusts (T_1, T_2) generated by the two propellers. Each propeller generates a torque around the CoM acting with effective moment arm length d. Simple Newtonian mechan-

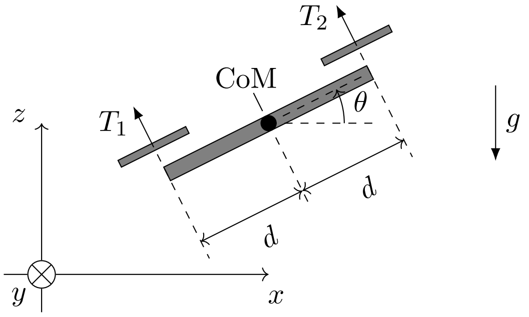

Figure 5.1: Schematics of the simulated 2D quadrotor system.

Parameter	Symbol	Value
Mass	m	$0.1\,\mathrm{kg}$
Effective moment arm length	d	$0.1\,\mathrm{m}$
Moment of inertia	I_{yy}	$8.3 \times 10^{-5}\,\mathrm{kgm}^2$
Gravitational acceleration	g	$9.81\,\frac{\mathrm{m}}{\mathrm{s}^2}$

Table 5.1: Physical parameters of the simulated 2D quadrotor dynamics model.

ics yield the simplified equations of motion [43]

$$\begin{aligned} m\ddot{x} &= -(T_1 + T_2)\sin(\theta) \\ m\ddot{z} &= (T_1 + T_2)\cos(\theta) - mg \\ I_{yy}\ddot{\theta} &= (T_2 - T_1)d, \end{aligned} \tag{5.1}$$

where $I_{yy} = \frac{1}{12}md^2$ is the moment of inertia about the body y-axis, and $g = 9.81\,\frac{\mathrm{m}}{\mathrm{s}^2}$ is the gravitational acceleration. Here, we have omitted the dependence of position and pitch angle, their derivatives, and the thrusts on time $t \in \mathbb{R}_{\geq 0}$. The dynamic parameters of the quadrotor and their respective values chosen for the simulation are summarized in Table 5.1.

Defining the state as $\boldsymbol{x} = \left[x,\, \dot{x},\, z,\, \dot{z},\, \theta,\, \dot{\theta}\right]^\top \in \mathbb{R}^6$ and the input as $\boldsymbol{u} = \left[T_1,\, T_2\right]^\top \in \mathbb{R}^2$, we can express the equations of motion (5.1) in state-space form similar to (3.1) as

$$\dot{\boldsymbol{x}} = \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u}) = \begin{bmatrix} x_2 \\ -\frac{1}{m}\sin(x_5)(u_1 + u_2) \\ x_4 \\ \frac{1}{m}\cos(x_5)(u_1 + u_2) - g \\ x_6 \\ \frac{1}{I_{yy}}d(u_1 - u_2) \end{bmatrix}. \tag{5.2}$$

This nonlinear system has infinitely many equilibria $\boldsymbol{z}_\mathrm{e} = (\boldsymbol{x}_\mathrm{e}, \boldsymbol{u}_\mathrm{e})$ satisfying $\boldsymbol{h}(\boldsymbol{z}_\mathrm{e}) = \boldsymbol{0}$, where $\boldsymbol{x}_\mathrm{e} = [x_\mathrm{e}, z_\mathrm{e}, 0, 0, 0, 0]^\top$ and $\boldsymbol{u}_\mathrm{e} = \left[\frac{mg}{2}, \frac{mg}{2}\right]^\top$, which correspond to hovering at the position $(x_\mathrm{e}, z_\mathrm{e})$. The linearization of (5.2) about any equilibrium is given by

$$\dot{\tilde{\boldsymbol{x}}} = \underbrace{\left.\frac{\partial \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_\mathrm{e} \\ \boldsymbol{u}=\boldsymbol{u}_\mathrm{e}}}}_{\boldsymbol{A}} \tilde{\boldsymbol{x}} + \underbrace{\left.\frac{\partial \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_\mathrm{e} \\ \boldsymbol{u}=\boldsymbol{u}_\mathrm{e}}}}_{\boldsymbol{B}} \tilde{\boldsymbol{u}}, \tag{5.3}$$

where $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x}_\mathrm{e}$ and $\tilde{\boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{u}_\mathrm{e}$ are deviations from the equilibrium, and

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & g & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{m} & \frac{1}{m} \\ 0 & 0 \\ \frac{d}{I_{yy}} & -\frac{d}{I_{yy}} \end{bmatrix}. \tag{5.4}$$

5.1.2 Setup

As the quadrotor exhibits nonlinear dynamics, we employ GPs to learn a dynamics model and then linearize it, as explained in Section 3.3. To illustrate the broad applicability of the GP learning approach, we assume no prior knowledge about the dynamics is available, i.e., $\boldsymbol{h} = \boldsymbol{g}$. We create a training data set consisting of N samples $(\boldsymbol{z}^{(i)}, \boldsymbol{y}^{(i)})$ by drawing the training inputs uniformly from the compact set

$$\mathcal{Z} = \left\{ (\boldsymbol{x}, \boldsymbol{u}) \in \mathbb{R}^8 \;\middle|\; \begin{bmatrix} 0 \\ -5\,\frac{\mathrm{m}}{\mathrm{s}} \\ 0 \\ -5\,\frac{\mathrm{m}}{\mathrm{s}} \\ -\frac{\pi}{2}\,\mathrm{rad} \\ -5\,\frac{\mathrm{rad}}{\mathrm{s}} \end{bmatrix} \leq \boldsymbol{x} \leq \begin{bmatrix} 2\,\mathrm{m} \\ 5\,\frac{\mathrm{m}}{\mathrm{s}} \\ 2\,\mathrm{m} \\ 5\,\frac{\mathrm{m}}{\mathrm{s}} \\ \frac{\pi}{2}\,\mathrm{rad} \\ 5\,\frac{\mathrm{rad}}{\mathrm{s}} \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix} \leq \boldsymbol{u} \leq \begin{bmatrix} 2\,\mathrm{N} \\ 2\,\mathrm{N} \end{bmatrix} \right\} \tag{5.5}$$

and calculating the targets as $\boldsymbol{y}^{(i)} = \boldsymbol{h}(\boldsymbol{z}^{(i)}) + \boldsymbol{w}^{(i)} = \dot{\boldsymbol{x}}^{(i)} + \boldsymbol{w}^{(i)}$, where $\boldsymbol{w}^{(i)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_\mathrm{n})$, $\forall i \in \mathbb{I}_N$. The covariance matrix of the observation noise is set to $\boldsymbol{\Sigma}_\mathrm{n} = 0.1^2\boldsymbol{I}$. We set the confidence threshold in Theorem 3.2 to $p = 0.99^6 \approx 0.94$, which yields $\sqrt{\chi^2_{n_z}\left(\sqrt[n]{p}\right)} = 4.5$. Hence, the learned linearized dynamics capture the true dynamics at the equilibrium with a probability of at least 94%. The simulations are performed in MATLAB. We use the YALMIP toolbox [44] to define and simplify the GEVPs (4.15) and (4.23) solve them using the MOSEK optimization software [45]. For the optimization with respect to the scalar optimization variables, we define the grid $\mathcal{G} = \{10^{-3}, 10^{-2.7}, \ldots, 10^{3}\}$ and consider $\epsilon \in \mathcal{G}$ in (4.15) and $(\epsilon_1, \epsilon_2) \in \mathcal{G} \times \mathcal{G}$ in (4.23).

5.1.3 Results

Minimum Control Frequency

First, we study the relationship between the amount of training data and the frequency at which the learned system must be controlled to guarantee stability with high probability, which is discussed in Section 4.3. To do this, we randomly draw 1000 training samples from $\mathcal{Z}$ and learn the dynamics from data sets $\mathcal{D}_N = \{\boldsymbol{z}^{(i)}, \boldsymbol{y}^{(i)}\}_{i=1}^{N}$ containing the first $N \in \{100, 150, \ldots, 1000\}$ samples. For each data set $\mathcal{D}_N$, we compute the MCF

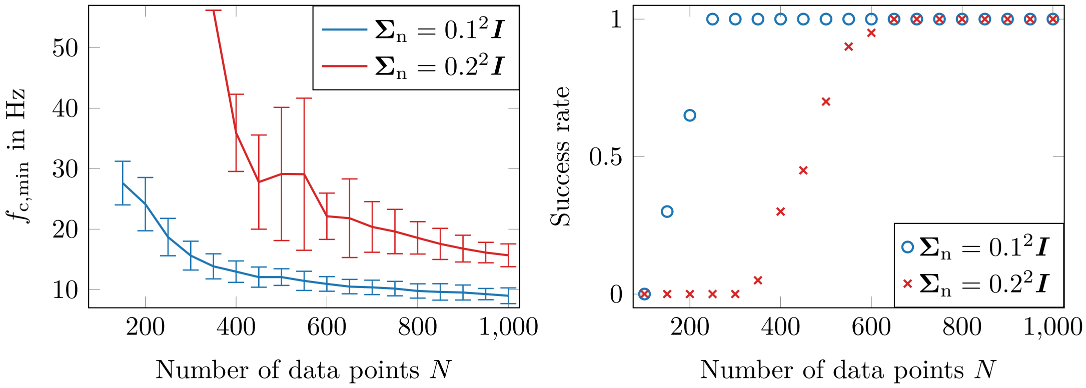

Figure 5.2: Left: Minimum control frequency required to ensure robust stability for different amounts of randomly drawn training data. The error bars represent ± one standard deviation. Right: Proportion of datasets for which (4.12) is feasible, i.e., for which a stabilizing controller can be found.

based on the learned uncertain model by solving the optimization problem (4.15). We repeat this procedure 20 times to account for the randomness of data generation. For comparison, we also consider a higher observation noise variance $\boldsymbol{\Sigma}_{\mathrm{n}} = 0.2^2\boldsymbol{I}$.

The obtained mean and standard deviation of the MCF are shown on the left of Fig. 5.2. In addition, we provide the proportion of data sets for which the optimization problem (4.15) is feasible, i.e., for which a stabilizing controller can be found, on the right of Fig. 5.2. If there is insufficient data, then (4.15) is always infeasible. A stabilizing controller can be found for some of the training data sets with $\boldsymbol{\Sigma}_{\mathrm{n}} = 0.1^2\boldsymbol{I}$ if at least $N = 150$ samples are drawn. This threshold increases to $N = 350$ for the larger noise variance. From $N \approx 250$ for the small and $N \approx 650$ for the large noise variance, (4.15) can be solved for all data sets. With an increasing amount of data, the MCF decreases significantly, for example, from around 28 Hz to less than 10 Hz for $\boldsymbol{\Sigma}_{\mathrm{n}} = 0.1^2\boldsymbol{I}$. The variance for the MCF is zero for $N = 350$ and $\boldsymbol{\Sigma}_{\mathrm{n}} = 0.2^2\boldsymbol{I}$ because in this case, (4.15) can only be solved for one data set. In general, however, more data reduces the variance of the computed MCF.

Quadrotor Trajectories

We also investigate the impact of control frequency and model uncertainty on the trajectories of the quadrotor system. For this, we set the initial state to $\boldsymbol{x}(0) = [1.2\,\mathrm{m}, 0, 0.2\,\mathrm{m}, 0, 0, 0]^\top$ and the desired hovering position to $(x_{\mathrm{e}}, z_{\mathrm{e}}) = (1\,\mathrm{m}, 0)$. To encourage aggressive control behavior, we set the weight matrices in the cost (4.16) to $\boldsymbol{Q}_J = \mathrm{diag}([100, 1, 100, 1, 100, 1])$ and $\boldsymbol{R}_J = 0.01\boldsymbol{I}_2$, respectively. As discussed in Section 4.2, robust stability of the linearized system is guaranteed with high probability for all control frequencies $f_{\mathrm{c}} = \xi f_{\mathrm{c,min}}$ with $\xi \geq 1$. We evaluate $\xi = \{1, 1.25, 1.5, 2\}$ with 20 randomly drawn training sets each and compute the optimized controller by

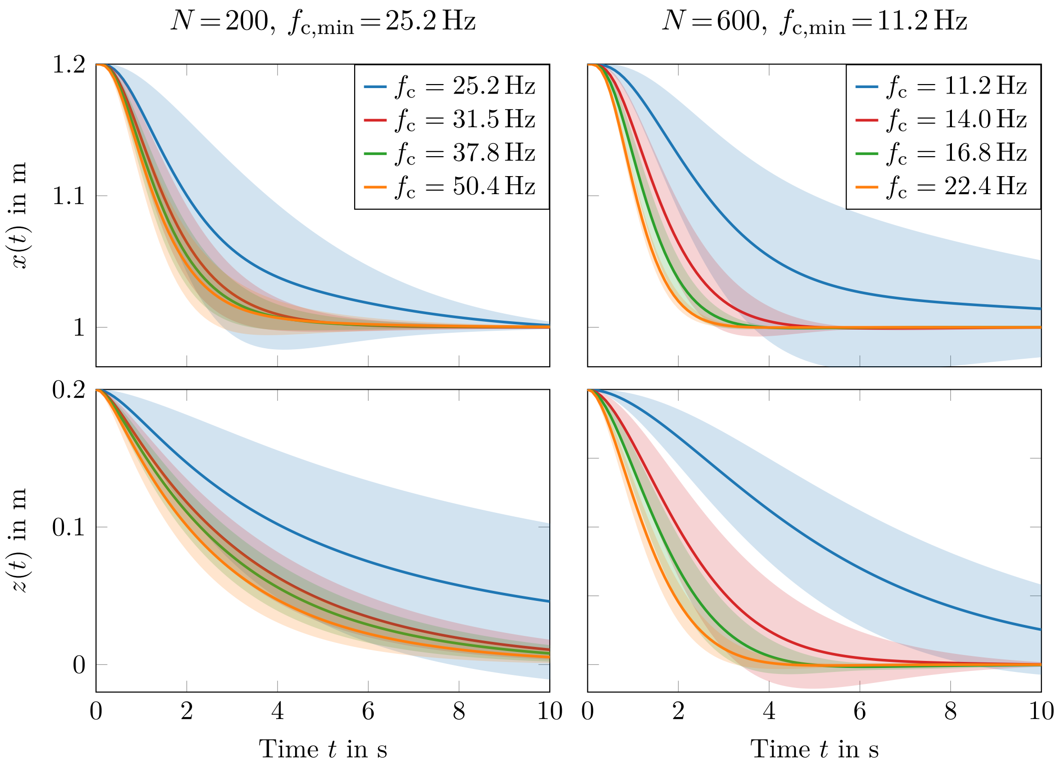

Figure 5.3: Quadrotor trajectories for different control frequencies $f_{\mathrm{c}} = \xi f_{\mathrm{c,min}}$, where $\xi \in \{1, 1.25, 1.5, 2\}$, and different amounts of training data. The shaded areas represent ± one standard deviation. Convergence to the setpoint significantly improves, and variance reduces if the control frequency is increased from its minimum value $f_{\mathrm{c}} = f_{\mathrm{c,min}}$.

solving (4.15). The system is simulated for a duration of 10 s for each combination of control gain and control frequency.

Fig. 5.3 shows the trajectories of the quadrotor position obtained for $N = 200$ and $N = 600$, respectively. The shaded areas represent ± one standard deviation. When controlling the system at the MCF, it converges only slowly to the setpoint, and the variance in the trajectories is significant for both values of N. Increasing the control frequency from its minimum value leads to a substantial improvement in the transient behavior and a reduced variance. Comparing the left and right plots, we observe that by tripling N from 200 to 600 and keeping the frequency approximately the same (22.4 Hz compared to 25.2 Hz), the desired x-position is reached about four times faster within around 2.5 s compared to around 10 s. A substantial improvement in transient response can also be seen for the z-position. This demonstrates that both the control frequency and the amount of data impact the control performance. In the following, we study these relationships in more detail.

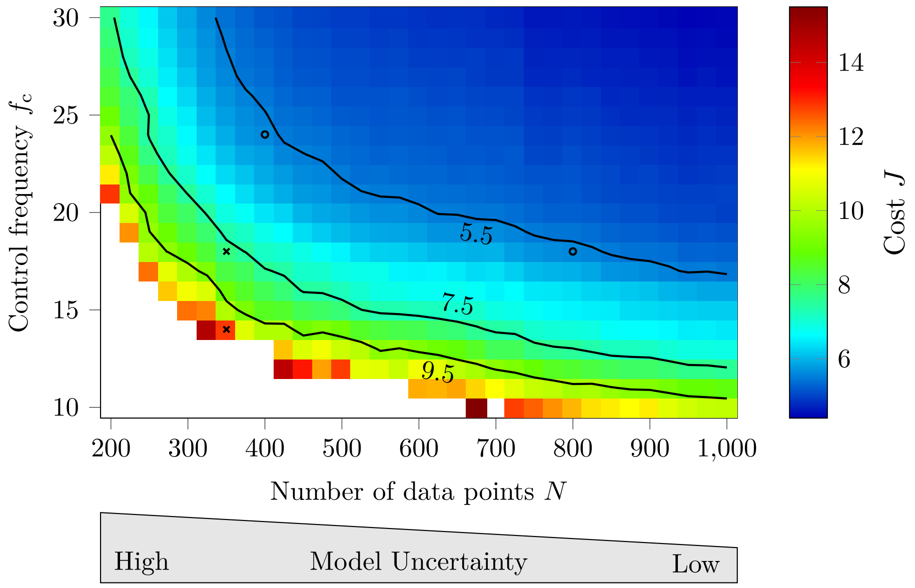

Figure 5.4: Illustration of the tradeoff between the control frequency and the amount of data in terms of performance. The white area indicates (N, f_c) pairs for which a stabilizing controller is only found in less than 50% of the cases. As shown by the contour lines of the average cost, for an increasing amount of data, similar performance can be achieved at a lower control frequency.

Control Performance

For a systematic analysis of performance, we optimize the controller via (4.23) for many different control frequencies $f_c = 10\,\text{Hz}, \dots, 30\,\text{Hz}$ and simulate the regulation to hovering at the location $(x_e, z_e) = (1\,\text{m}, 0)$ for five different initial conditions. We compute the continuous-time quadratic cost for a horizon of 10 s using the weight matrices $\boldsymbol{Q}_J$ and $\boldsymbol{R}_J$ defined above. The simulation is repeated 20 times with randomly drawn training data sets, and we consider $N = 200, 225, \dots, 1000$.

Fig. 5.4 shows the average cost and its contour lines for three different values $J \in \{5.5, 7.5, 9.5\}$. The corresponding standard deviation of the cost is depicted in Fig. 5.5. The white area on the bottom left in Figures 5.4 and 5.5 represents (N, f_c) pairs for which (4.23) is infeasible in more than 50% of the cases and corresponds to the results shown in Fig. 5.2. The contour lines of the cost in Fig. 5.4 illustrate there is a tradeoff between data and control frequency in terms of the control performance: As model uncertainty decreases through the addition of more data, a similar cost can be obtained at a lower control frequency. On the other hand, the control frequency strongly affects how much data must be available to achieve a certain performance. For example, if we increase f_c by 33% from 18 Hz to 24 Hz, only half the amount of data ($N = 400$ instead of $N = 800$) is needed to get the same cost $J = 5.6$. Increasing the control frequency for a given amount of data reduces the cost, for instance, by 42% when increasing f_c by 29% from 14 Hz to 18 Hz for $N = 350$. It is evident from

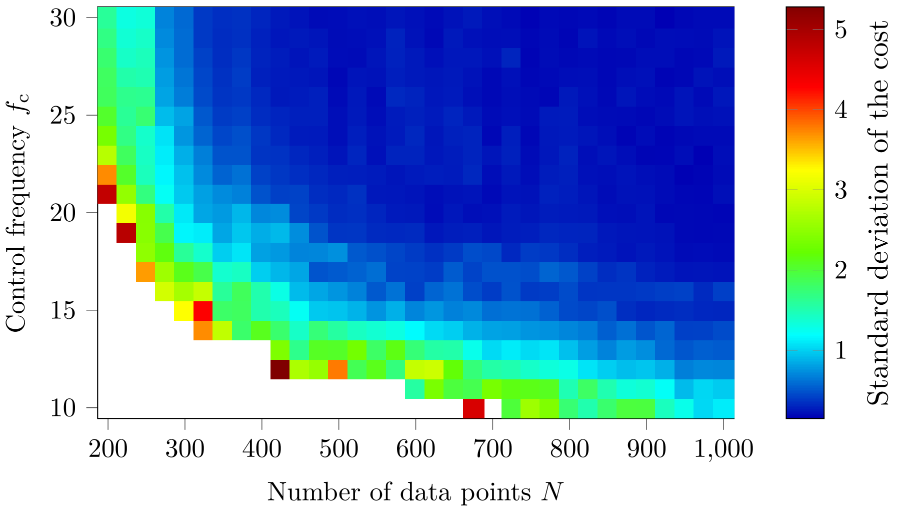

Figure 5.5: Standard deviations of the costs corresponding to the means provided in Fig. 5.4. The decrease in cost with the control frequency and the amount of data observed shown in Fig. 5.4 is accompanied by an even greater reduction in standard deviation.

the contour lines' shape that the sensitivity of the performance with respect to the control frequency increases with the size of the training data set. We can also observe from Fig. 5.5 that a decrease in the cost mean is accompanied by an even greater reduction of the standard deviation, which is consistent with Fig. 5.3.

5.1.4 Discussion

Fig. 5.3 and Fig. 5.4 demonstrate that choosing a larger control frequency than the MCF improves performance and reduces variance. Considering this, the MCF should be seen as a lower bound that allows us to reduce the control frequency without sacrificing stability if needed, for example, to save computation or communication resources on quadrotors.

As illustrated in Fig. 5.2 and Fig. 5.4, the amount of data required for stability or achieving a specific performance depends on the frequency at which we can run the controller. A slight increase in control frequency can compensate for even a significant lack of data. This insight provides a potentially cheaper alternative to collecting large amounts of data from real systems, which is often expensive and tedious [7].

Stable regulation of the quadrotor is achieved for all simulated initial conditions, even when operating at the MCF, as shown in Fig. 5.3. This is remarkable, as the computation of the MCF via (4.15) is based on the linearized dynamics (4.1) and only considers the uncertainty corresponding to the GP variance, not the error due to linearizing the nonlinear system (3.1). One reason is that the stability conditions stated in Theorem 4.1 are sufficient but not necessary. Consequently, the computation of the MCF introduces some conservatism that may compensate for the linearization error, as is the case for

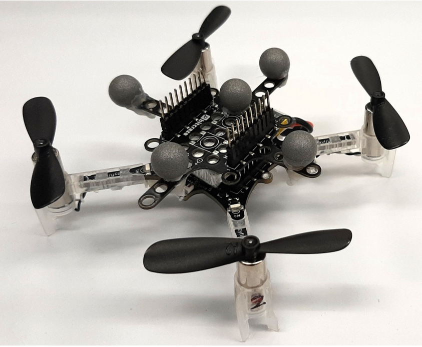

Figure 5.6: The Crazyflie 2.1 quadrotor used for the experiments. The motion capture system tracks the five reflective markers attached to the body.

our example.

5.2 Hardware Experiment

We conduct hardware experiments with a quadrotor to study the role of control frequency in a real-world scenario.

5.2.1 Hardware: Quadrotor and Motion Capture System

The Crazyflie 2.1 quadrotor is an open-source development platform developed by Bitcraze [46]. We use the quadrotor depicted in Fig. 5.6 for all our experiments. It has a total mass of $m = 35\,\mathrm{g}$ and its four propellers can produce a maximum total thrust of about $0.6\,\mathrm{N}$, which is about 1.8 times the thrust required for hovering. The maximum flight time is about 7 minutes with a fully charged battery, but we replace the battery after about 2 minutes of flying to ensure constant behavior of the system. The Crazyflie 2.1 has three onboard 3-axis accelerometer and gyroscope sensors to measure its linear acceleration and angular velocity. In principle, these measurements can be used to estimate the quadrotor's position and orientation through numerical integration. However, the quality of these estimates is generally low due to sensor noise and the resulting integration drift.

Therefore, we employ a Vicon motion capture system [47] consisting of six cameras to precisely track the quadrotor's position and orientation. For this purpose, we attach five spherical markers coated with reflective material to the Crazyflie's body, as shown in Fig. 5.6. Each motion capture camera measures the relative 2D coordinates of the markers in its image coordinate system. Before tracking can begin, the Vicon system needs to be calibrated to define a world frame in the experimental space and determine the 3D coordinates of each camera in the world frame. We define the world frame such that its z-axis points upwards. Calibration ensures the system can accurately triangulate the markers' position in 3D space from the 2D image coordinate pairs. Once the 3D positions of the markers on the quadrotor are determined, the Vicon software calculates the quadrotor's position and orientation in the world frame. The obtained data is published on a ROS topic [48] at a frequency of 200 Hz, from where it can be read, for instance, by a motion control algorithm.

In addition to the quadrotor's pose, its velocity and attitude rates are required for accurate motion control. An extended Kalman filter (EKF) is used to estimate these values based on incoming measurements, including data from motion capture and the onboard accelerometers and gyroscopes, the measurement model, and the model of the system itself. Details on the employed state estimation method can be found in [49].

5.2.2 Simplified Dynamics Model

Our approach for controlling the quadrotor's motion is based on the Crazyswarm ROS package [50]. It features a cascaded control architecture consisting of an outer loop for position control and an inner loop for attitude control, explained in [51]. We set the desired x- and y-position and the desired yaw rate, i.e., the speed of rotation around the z-axis, to zero, thus restricting the quadrotor's motion to the vertical z-direction in the experiments.

We can therefore define a simplified dynamics model with state $\boldsymbol{x} = [z - z_{\mathrm{e}},\, \dot{z}]^\top \in \mathbb{R}^2$ and input $u = T - T_{\mathrm{e}} \in \mathbb{R}$, where z_{e} is the desired height, T is the total thrust produced by the four propellers, and $T_{\mathrm{e}} = mg$ is the thrust required for hovering. In addition to the upper bound on the total thrust mentioned above, we specify a lower bound $T \geq 0.3\, T_{\mathrm{e}}$ to prevent the quadrotor from falling uncontrolled. Hence, the control input is constrained by $u \in [-0.7,\, 0.8] T_{\mathrm{e}}$ in our experiments. However, we want to operate within these bounds as the sampled-data control approach considered in this thesis does not consider input constraints.

We assume the simplified dynamics in z-direction to have a linear structure

$$\dot{\boldsymbol{x}} = \boldsymbol{h}(\boldsymbol{x}, u) = \begin{bmatrix} 0 & 1 \\ \theta_1 & \theta_2 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} 0 \\ \theta_3 \end{bmatrix} u, \tag{5.6}$$

where the first row follows from the definition of the state vector, and the second row contains three unknown scalar parameters θ_1, θ_2 and θ_3. In the case of $\theta_1 = \theta_2 = 0$ and $\theta_3 \neq 0$, (5.6) corresponds to a simple double integrator system as considered, for example, in Example 3.1. If we neglect air drag, time delays, or other disturbances, the dynamics in z-direction can be modeled as $m\ddot{z} = T - mg$ based on Newton's law. This idealized model corresponds to (5.6) having a double integrator structure with $\theta_3 = \frac{1}{m}$. We consider this prior knowledge to define prior distributions on the parameters.

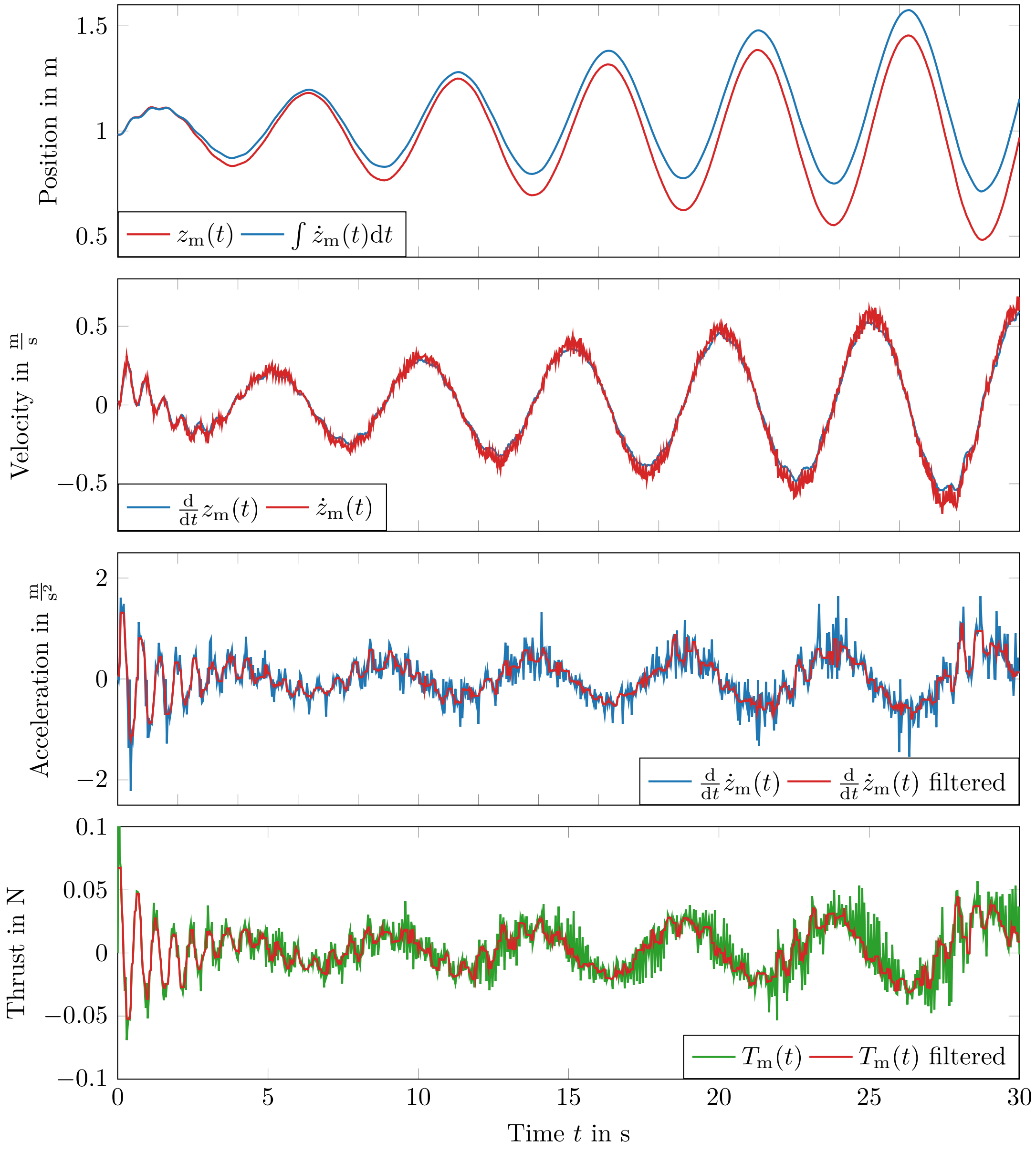

Figure 5.7: The measured data consists of position $z_\mathrm{m}(t)$, velocity $\dot{z}_\mathrm{m}(t)$ and thrust $T_\mathrm{m}(t)$. The first plot shows the measured position and the integration of the measured velocity and the second plot shows the measured velocity and the derivative of the measured position with respect to time. We obtain the acceleration by differentiating the measured velocity and applying a median filter, as illustrated in the third plot. The fourth plot shows the thrust, which is also median-filtered. We use the red trajectories as training data for model learning.

5.2.3 Data Collection

We define a sinusoidal reference trajectory for the z-position with a length of 30 s, a period of 5 s and an amplitude that increases linearly from 0.1 m to 0.5 m during the

Index i	Prior mean $\mu_i^{(0)}$	Prior standard deviation $\sigma_i^{(0)}$
1	0	$1\,\frac{1}{\mathrm{s}^2}$
2	0	$1\,\frac{1}{\mathrm{s}}$
3	$\frac{1}{m} = 28.57\,\frac{1}{\mathrm{kg}}$	$10\,\frac{1}{\mathrm{kg}}$

Table 5.2: Chosen means and variances of the prior distributions $\theta_i^{(0)} \sim \mathcal{N}\left(\mu_i^{(0)}, \sigma_i^{(0)}\right)$ of the unknown dynamics parameters in (5.6).

duration of the trajectory. We use a PID controller to track the reference trajectory, which updates the desired attitude rates and total thrust at a frequency of 30 Hz. The same frequency is used for recording the quadrotor position, velocity, and thrust.

The first two plots in Fig. 5.7 show the measured position $z_\mathrm{m}(t)$ and velocity $\dot{z}_\mathrm{m}(t)$, respectively, in red color. In addition, we provide the numerical integration of $\dot{z}_\mathrm{m}(t)$ in the first plot and the numerical differentiation of $z_\mathrm{m}(t)$ in the second plot. We observe that the signals obtained through integration and differentiation do not deviate strongly from the measured trajectories even after 30 s, demonstrating the high quality of the velocity estimation with the EKF. To obtain the acceleration, we differentiate $\dot{z}_\mathrm{m}(t)$ with respect to time. As the obtained signal contains spikes, we smooth it using a median filter with a window size of five. Both signals are provided in the third plot in Fig. 5.7. The fourth plot shows the measured thrust $T_\mathrm{m}(t)$, which we median-filter similar to the acceleration. The acceleration and thrust signals have spikes at the beginning, which is probably related to the fact that the quadrotor cannot immediately follow the sinusoidal reference due to its inertia. We nevertheless keep this part of the measured trajectory, as we want the model learned below to improve as much as possible with more training data. So, we create a training data set $\mathcal{D}$ similar to (3.3) that consists of $N = 900$ samples from the measured position and velocity and the filtered acceleration and thrust signals, i.e., the training inputs and targets are given by

$$\begin{aligned} \boldsymbol{z}^{(i)} &= \left[z_\mathrm{m}(t_i),\ \dot{z}_\mathrm{m}(t_i),\ T_\mathrm{m,filtered}(t_i)\right]^\top, \\ y^{(i)} &= \ddot{z}_\mathrm{m,filtered}(t_i), \end{aligned} \tag{5.7}$$

$i = 1, \dots, N$, where the timestamp of the i-th observation is approximately $t_i \approx \frac{i}{30}$ s.

5.2.4 Results

Model Learning with BLR

We use BLR, as explained in Section 3.2, to learn probabilistic estimates of the unknown parameters in (5.6) from the training samples (5.7). As can be observed from Fig. 5.7, the measured trajectory signals are more informative towards the end due to higher signal amplitudes, which yield larger feature values in the mean and covariance updates (2.22) and (2.23), respectively. We account for this difference in data quality by randomly reshuffling the training samples in $\mathcal{D}$ but continue with the same notation for clarity.

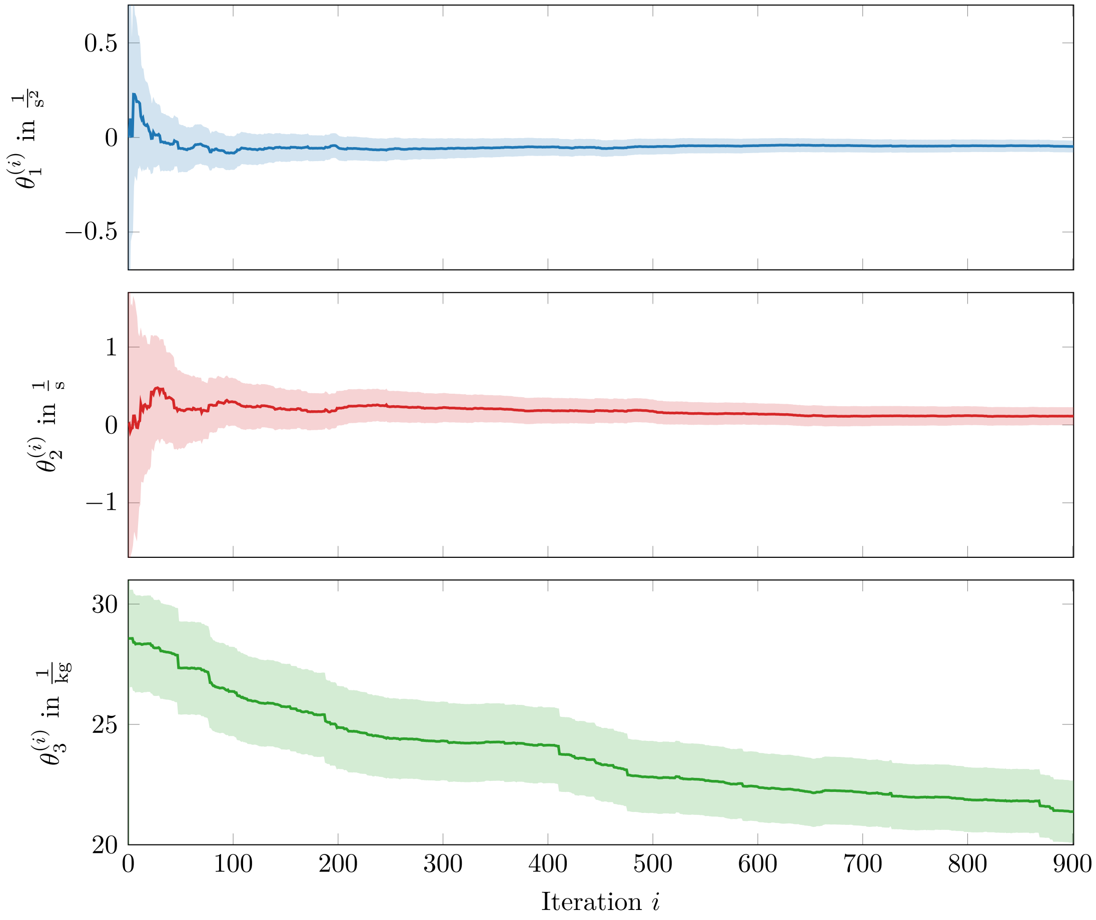

Figure 5.8: Illustration of the learning curves for the unknown parameters in (5.6), which are learned using iterative BLR. The plots show the estimated mean after i training samples. The shaded areas correspond to ± two standard deviations.

For the prior distributions $\theta_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, $i = 1, 2, 3$, of the unknown parameters, we choose the prior means and variances provided in Table 5.2 based on the idealized double integrator dynamics model discussed above. The observation noise variance also represents a hyperparameter to be chosen. We take the following approach, again making use of prior knowledge: We calculate the difference between the observed training targets and the targets that were to be expected from the prior parameter means in Table 5.2 and compute the standard deviation of the differences. We then double the obtained value to account for the fact that the idealized prior model is itself subject to uncertainty, yielding the total estimated noise standard deviation

$$\hat{\sigma}_\mathrm{n} = 2\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(\boldsymbol{\mu}^{(0)\top}\boldsymbol{z}^{(i)} - y^{(i)}\right)^2} = 0.6570\,\frac{1}{\mathrm{s}^2}, \tag{5.8}$$

where $\boldsymbol{\mu}^{(0)} = \left[\mu_1^{(0)}, \mu_2^{(0)}, \mu_3^{(0)}\right]^\top$ is the prior mean vector. We update the parameter estimates iteratively using the 900 training samples. The learning curves for the un-

Iteration i	$\boldsymbol{A}$	$\hat{\boldsymbol{A}}$	$\boldsymbol{b}$	$\hat{\boldsymbol{b}}$
50	$\begin{bmatrix} 0 & 1 \\ -0.0587 & 0.1905 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 \\ 0.1714 & 0.6865 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 27.3400 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 2.6707 \end{bmatrix}$
100	$\begin{bmatrix} 0 & 1 \\ -0.0826 & 0.2944 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 \\ 0.1201 & 0.4160 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 26.3659 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 2.5588 \end{bmatrix}$
300	$\begin{bmatrix} 0 & 1 \\ -0.0591 & 0.2231 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 \\ 0.0713 & 0.2513 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 24.3175 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 2.2544 \end{bmatrix}$
500	$\begin{bmatrix} 0 & 1 \\ -0.0489 & 0.1767 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 \\ 0.0541 & 0.2013 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 22.8195 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 2.0605 \end{bmatrix}$
900	$\begin{bmatrix} 0 & 1 \\ -0.0477 & 0.1127 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 \\ 0.0400 & 0.1505 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 21.3777 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1.7705 \end{bmatrix}$

Table 5.3: Parameters of the learned uncertain model (3.17) of the dynamics (5.6) after observing different numbers of training samples.

known parameters are provided in Fig. 5.8 together with ± two standard deviations. It can be seen that the estimated means of θ_1 and θ_2 converge to around zero after a few hundred data points, and the corresponding uncertainty strongly decreases. The uncertainty associated with θ_3 behaves differently. Although being initialized with a large value of $10\frac{1}{\mathrm{kg}}$, the standard deviation drops sharply after observing the first data point. From that point on, the uncertainty decreases slowly during the learning process. We use Theorem 3.1 to compute an uncertain linear system of the form (3.17) from the parameter estimates that captures the true dynamics at the equilibrium with a probability of at least $p = 0.99$. Table 5.3 provides the learned system matrices after increasing amounts of training data. As expected, the element-wise uncertainty terms in $\hat{\boldsymbol{A}}$ and $\hat{\boldsymbol{b}}$ decrease with more data.

Model Validation

Before designing robust sampled-data controllers based on the learned model, we validate our choice of hyperparameters and check if the model overfits the training data. We create a validation data set similar to the training data set shown in Fig. 5.7. We start from a random initial condition $\hat{\boldsymbol{x}}(0) = \left[\hat{z}(0), \dot{\hat{z}}(0)\right]^\top = \boldsymbol{x}_\mathrm{m}(t_0)$, where t_0 is drawn randomly from $\left[0, \frac{1}{30}, \dots, 25\right]$ s, on the validation trajectory. From this initial condition, we apply the corresponding input sequence from the validation data set and predict the state trajectory forward in time through numerical integration of the learned dynamics using a Runge-Kutta fourth-order method [52]. The accuracy of the predicted trajectory is evaluated for a prediction horizon of $T_\mathrm{pred} = \frac{M}{30}$ s, where $M \in \mathbb{N}$ is chosen randomly between 60 and 150 such that $T_\mathrm{pred} \in [2, 5]$ s, by calculating the root-mean-square error (RMSE) for the z-position

$$\mathrm{RMSE} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(\hat{z}(t_i) - z_\mathrm{m}(t_0 + t_i))^2}. \tag{5.9}$$

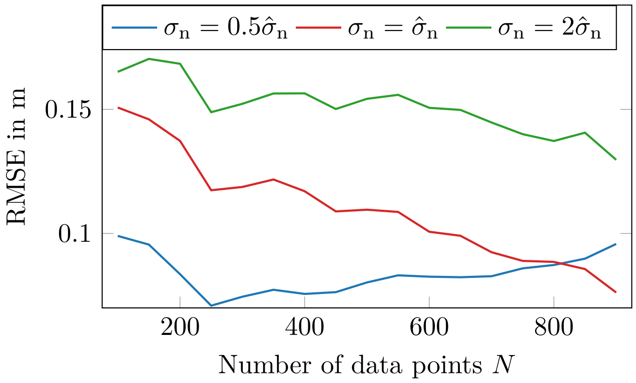

Figure 5.9: Validation of the chosen hyperparameter value $\hat{\sigma}_n$ for the observation noise standard deviation σ_n. We compare the measured validation trajectory to the predicted trajectory that is obtained through numerical integration of the learned model with the measured input sequence. Our choice for σ_n allows the model to significantly improve through the addition of more data.

We consider 100 random initial conditions on the validation trajectory and calculate the average RMSE.

The results are shown in Fig. 5.9. We observe the model keeps improving until all 900 data points are used for training, i.e., the training procedure does not result in overfitting. We also consider different values than in (5.8) for the standard deviation of the observation noise σ_n. Halving the noise standard deviation leads to overfitting, and doubling it results in RMSE decreasing only very slowly. Therefore, our choice for σ_n is suitable as it ensures the model can significantly improve with adding more data, which is a prerequisite to fairly assess the impact of data.

Minimum Control Frequency

We compute the MCF by solving the GEVP (4.12) using the learned model after $N \in \{200, 205, \dots, 900\}$ training samples. The results are shown in Fig. 5.10. We observe that for $\sigma_n = \hat{\sigma}_n$, at least 225 samples are required for computing a stabilizing sampled-data controller. The MCF strongly decreases when the model is updated with more data and converges to around 3 Hz. For comparison, we also consider a 50% larger standard deviation of the noise, i.e., $\sigma_n = 1.5\hat{\sigma}_n$. In this case, (4.12) is infeasible for less than 475 data points, and the MCF converges to around 5 Hz. Next, we study stability and performance with a learning-based sampled-data controller for different control frequencies and data sets, assuming $\sigma_n = \hat{\sigma}_n$.

Quadrotor Trajectories

We consider the task of tracking a desired setpoint, which is defined as hovering at $z_e = 0.4\,\text{m}$, and set the initial condition to hovering at $z(0) = 0.7\,\text{m}$. We compute sampled-data controllers for four different control frequencies $f_c \in \{10,\ 15,\ 30,\ 60\}\,\text{Hz}$

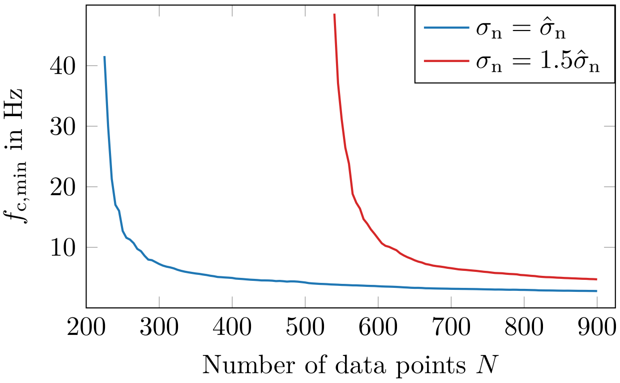

Figure 5.10: Minimum control frequency required to ensure robust stability for different amounts of training data. If we increase the standard deviation of the observation noise by 50% compared to its estimated value (5.8), much more data is required to be able to compute a stabilizing controller, and the controller must operate at a higher frequency.

		f_c in Hz			
		10	15	30	60
	250	unstable	5.27	4.30	3.08
N	500	unstable	5.22	3.53	2.99
	750	unstable	4.61	3.38	3.19

Table 5.4: Quadratic costs of the quadrotor trajectories shown in Figures 5.11 and 5.12. Increasing the control frequency leads to a greater decrease in cost than using more data points.

by solving (4.23) with the learned models after 250, 500 and 750 training samples. To encourage fast convergence to the desired setpoint, we set the weight matrices in the cost (4.16) to $\boldsymbol{Q} = \mathrm{Diag}([100, 1])$ and $R = 1$, mainly penalizing deviations from the desired position.

Fig. 5.11 consists of three plots showing the quadrotor's position, each corresponding to a specific number of training samples. In that way, the impact of the control frequency on the tracking behavior is illustrated for different learned models. In all three cases, the quadrotor cannot be stabilized at a control frequency of 10 Hz. This is unexpected at first glance, as the calculation of the MCF shown in Fig. 5.10 suggests that an even smaller frequency should be sufficient for stability if enough data is available. We discuss potential reasons for this discrepancy in Section 5.3. For the three higher control frequencies $f_\mathrm{c} \in \{15, 30, 60\}$ Hz, we observe that the height of the quadrotor decreases faster if the control frequency is increased. This effect occurs for all three data sets and is particularly strong for the smallest one.

Table 5.4 provides the quadratic costs of the tracking trajectories. Increasing the control frequency by a factor of four from 15 Hz to 60 Hz reduces the cost on average by 39%, whereas tripling the number of training samples from 250 to 750 only leads to

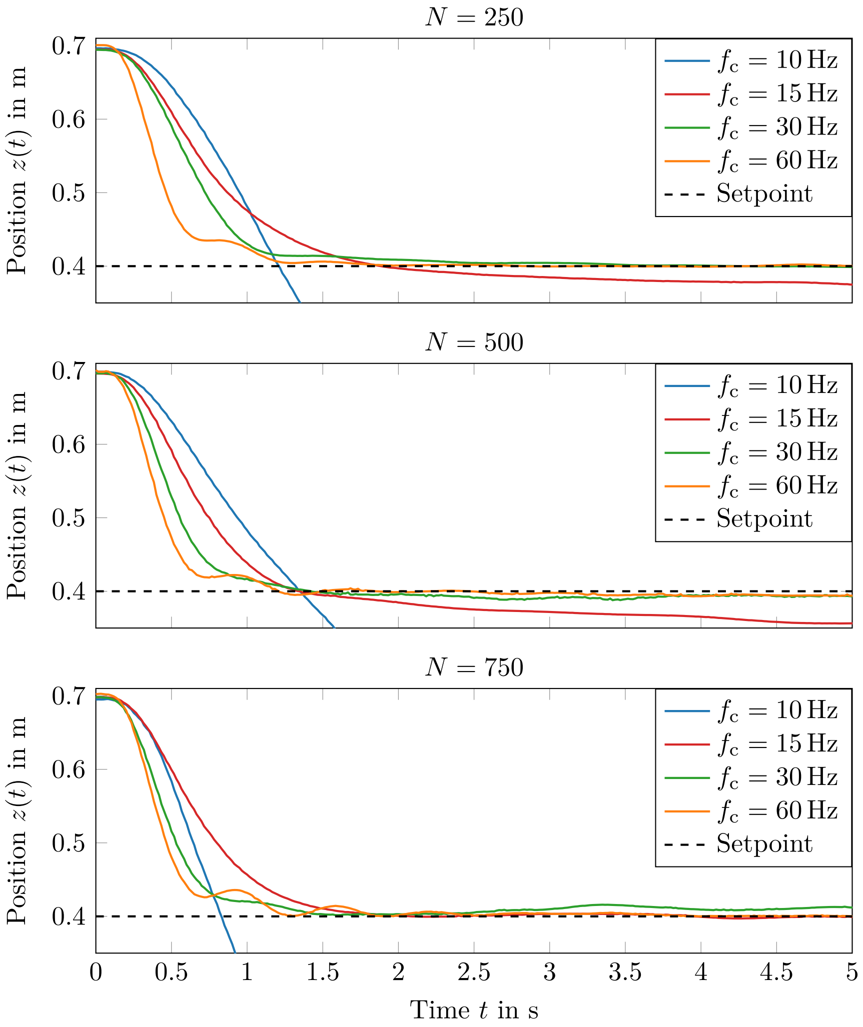

Figure 5.11: Impact of the control frequency on tracking the desired setpoint for different amounts of data. The transient behavior improves significantly if the controller runs at a higher frequency. In our experiments, a control frequency of 10 Hz is insufficient to stabilize the quadrotor, which does not align with the results for the minimum control frequency required in theory; see Fig. 5.10 .

a 10% average reduction in cost. For $f_c = 60$ Hz, the cost even increases when using the largest data set.

In Fig. 5.12, we group the tracking trajectories by the control frequency. We observe a significantly smaller change in the trajectories when using more data than when increasing the control frequency; see Fig. 5.11. Remarkably, all trajectories obtained for $f_c = 60$ Hz show oscillations that become stronger the more data is used. This represents another difference compared to simulation in Section 5.1, where no oscillations occur. We also investigate this behavior in Section 5.3.

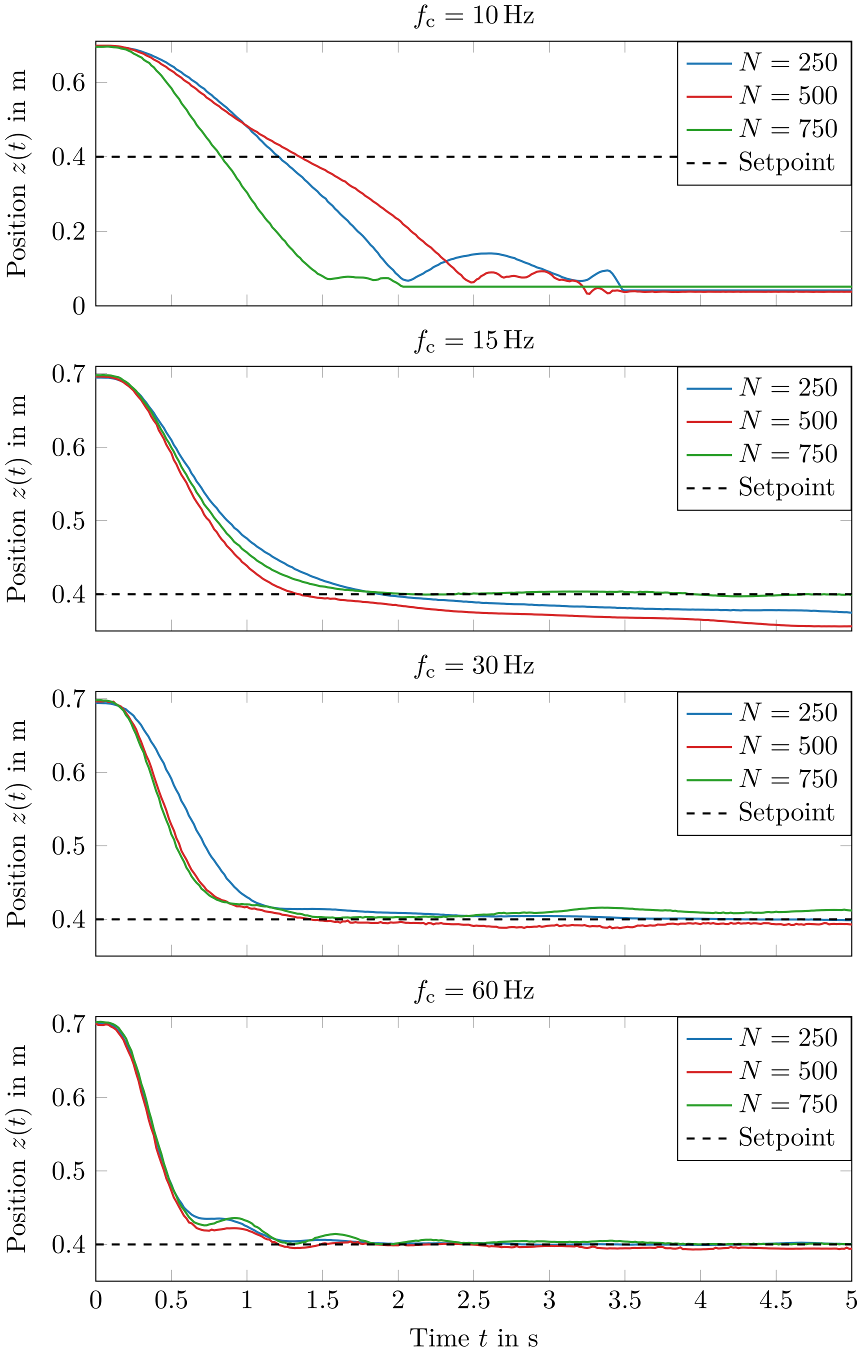

Figure 5.12: The transient response slightly improves with the amount of data, but the improvement is much smaller than when varying the control frequency; see Fig. 5.11. The steady-state offset decreases with a higher control frequency.

5.2.5 Discussion

As shown in Table 5.3, the uncertainty $\hat{\boldsymbol{A}}$ associated with the dynamics matrix decreases by a factor of around four to five when increasing the number of data points from 50 to 900. In contrast, the magnitude of the uncertainty $\hat{\boldsymbol{b}}$ in the input vector decreases only by around 34%. Still, as shown in Fig. 5.10, the model learned from all $N = 900$ data points yields an MCF that is smaller by a factor of around 15 than for $N = 220$. This illustrates that the MCF is mainly affected by $\hat{\boldsymbol{A}}$ in the considered scenario. One reason for this is that the uncertainty $\hat{\boldsymbol{b}}$ associated with $\boldsymbol{b}$ is much smaller than the estimated value of $\boldsymbol{b}$, whereas the uncertainty in the dynamics matrix is of the same magnitude as the estimate.

It can be seen from Fig. 5.11 or Fig. 5.12 that the quadrotor does not exactly reach the desired height of 0.4 m even after 5 s. As seen from Fig. 5.12, the deviation generally becomes smaller when using a higher control frequency. This behavior is likely because the assumed hovering thrust T_e, which directly affects the applied thrust $T = T_\mathrm{e} + u$, is not exactly equivalent to the actual thrust required for hovering $T_\mathrm{e,true}$. If we assume $\theta_1 = \theta_2 = 0$ in (5.6), then it is easy to show that the steady-state height of the closed-loop system is given by

$$z = z_\mathrm{e} + \frac{T_\mathrm{e,true} - T_\mathrm{e}}{\theta_3 k_1}, \tag{5.10}$$

where $k_1 \in \mathbb{R}$ is the first element of the control gain vector. Any difference between the assumed and the actual hovering thrust leads to a constant offset between the actual equilibrium of the closed-loop system and the desired hovering state. This offset can be made smaller by increasing the value of k_1. Increasing the control frequency gives the optimization of the robust sampled-data controller via (4.23) more freedom to reduce the cost, which results in higher control gains. Therefore, an indirect consequence of increasing f_c is a reduction of the steady-state offset.

5.3 Comparative Discussion of Simulation and Experiment

Compared to simulation, real-world experiments are usually affected by various additional and unmodeled effects, commonly referred to as “sim-to-real-gap” [53, 54]. In this section, we compare the results reported in Sections 5.1 and 5.2, focusing on the differences.

Simulation and experiment yield qualitatively similar results regarding the role of control frequency and data. In both scenarios, a certain amount of training data is needed such that a sampled-data controller for which robust stability is guaranteed for the learned model can be found via the optimization problem (4.12). Adding more training data reduces the lower bound on the control frequency for which the optimization problem (4.23) is feasible. The control performance improves with both the control frequency and the size of the training data set, and the control frequency generally has a more substantial influence on the achieved cost in the considered simulation and real-world scenarios. Still, we identify two apparent differences between simulation and

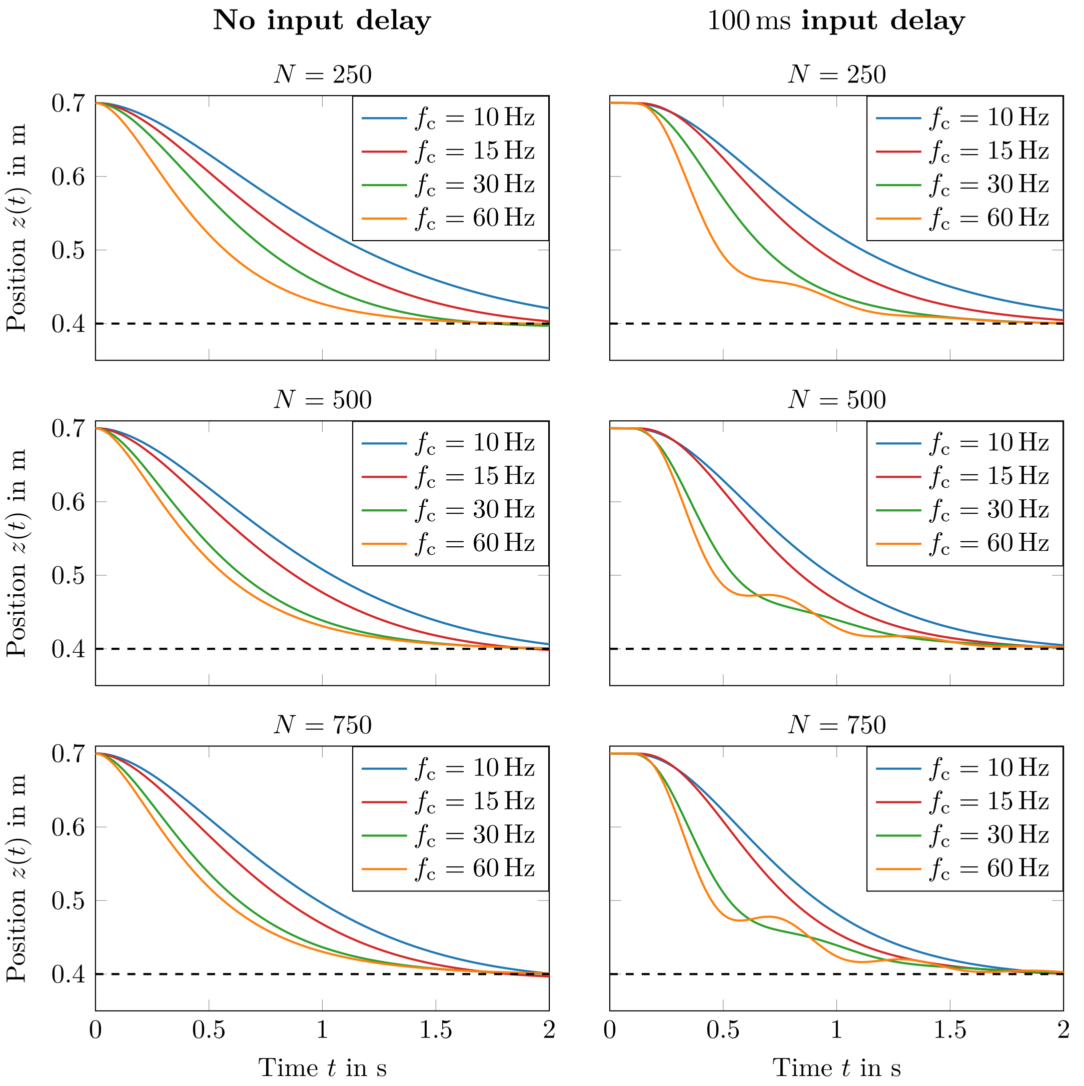

Figure 5.13: We simulate the learned model with the same control gains and control frequencies as those used to generate the trajectories in Fig. 5.11 and Fig. 5.12 without input delay (left) and with an input delay of 100 ms (right). With input delay, we observe oscillations similar to the experiments.

experiment, which we discuss in the following.

5.3.1 Oscillations

As already mentioned, the tracking trajectories in the experiment exhibit oscillations for a control frequency of 60 Hz, and the amplitude of the oscillations increases the more data is used for model learning. In principle, one reason for this non-ideal behavior could be the BLR model overfitting the training data. However, as demonstrated by Fig. 5.9, overfitting does not occur with our choice for the observation noise variance.

A second possible explanation could be that the input constraints of the quadrotor mentioned in Section 5.2.1, which are not accounted for in the controller design, become active during the experiments. However, the control input calculated with the sampled-data control law (4.2) always stays within the constraint set even with the most aggressive controller. We have ensured this by choosing the weight matrices $\boldsymbol{Q}_J$ and $\boldsymbol{R}_J$ of the cost and the difference of 0.3 m between the initial and the desired height accordingly.

As a third potential reason for the oscillations, we investigate delays in the control loop. The stability conditions derived in Section 4.2 consider the delay introduced to the system by the discrete-time feedback. However, we assume throughout Chapter 4 that at the sampling instant t_k, the control input immediately changes to $\boldsymbol{u}(t_k) = \boldsymbol{K}\boldsymbol{x}(t_k)$, i.e., input delays are not accounted for. A close inspection of the tracking trajectories in Figures 5.11 or 5.12 shows that after setting the desired heightto 0.4 m at time zero, it takes about 50 to 100 ms for the quadrotor to accelerate. This could be an indication of there being a non-negligible delay between calculating the desired thrust and the propellers generating this thrust. To verify if input delays are responsible for the oscillations, we conduct a numerical simulation with the dynamics learned from all 900 data points with and without input delay. Here, we use the same control frequencies and control gains as in the experiments. The simulation results are shown in Fig. 5.13, grouped similarly as in Fig. 5.11. Without input delay, the behavior is the same as in the simulation of the 2D quadrotor in Fig. 5.3, i.e., the transient response improves with a higher control frequency but never exhibits oscillations. However, from an input delay of 100 ms, oscillations that are very similar to those observed in the experiment occur in the simulation. Therefore, we can state with high confidence that input delays are at least to a large extent responsible for the oscillations at 60 Hz. That there are no oscillations at the smaller control frequencies $f_\mathrm{c} \in \{15, 30\}$ Hz is likely because the corresponding controllers are more conservative and, thus, amplify the effects of the delay less strongly. Multiple effects contribute to this delay, including the motion capture system, the communication latency between the PC and the quadrotor, and the inertia of the propellers, which prevents an instant change in the propeller speed. While our simple simulation suggests the magnitude of the delay to be in the order of 100 ms in the experiments, this value cannot be precisely determined without specific measurements. However, in [55], the latency in the communication alone is specified as 22 to 46 ms for an expensive quadrotor system, suggesting that 100 ms is a realistic estimate for the total delay with the relatively cheap Crazyflie 2.1 quadrotors.

5.3.2 Sufficiency of the Minimum Control Frequency

The second notable difference is that in contrast to the simulation, the MCF provided in Fig. 5.10 is too small to stabilize the quadrotor in the experiments. For example, for 750 training samples, a control frequency of 3 Hz should, in theory, be sufficient to robustly stabilize the system, but the quadrotor becomes unstable even when controlled at a frequency of 10 Hz. This is even though the simplified dynamics (5.6) of the z-

motion are approximately linear, unlike the nonlinear 2D quadrotor dynamics (5.1) in the simulation.

In theory, the MCF guarantees robust stability of the system despite model uncertainty. Hence, we can conclude that the uncertainty of the learned simplified dynamics (5.6) does not capture the full uncertainty in the real quadrotor system, as otherwise, the MCF would be sufficient for stability. The unstable behavior can be attributed to the failing stabilization in the horizontal direction, which we run at the same frequency as the z-position controller. We made this decision for the experiments as we wanted all possible effects of varying the control frequency to come into play. However, as the simplified dynamics model is learned only from data about the vertical motion (5.7), the uncertainty associated with the learned model likely cannot fully capture the uncertainty in the horizontal dynamics. We suggest two approaches to ensure the system can be controlled at (roughly) the computed MCF. First, we could learn a full-dimensional dynamics model of the quadrotor, for example, with GPs, which would also include the horizontal dynamics and their associated uncertainty. Second, decoupling the control frequencies of the vertical and the horizontal controller should lead to the overall system's stability being much less affected by the x-y-control and depending almost exclusively on the z-controller and its frequency. Confirming these ideas would require testing them in additional experiments, which is beyond the scope of this thesis.

Chapter 6

Conclusions and Outlook

6.1 Conclusions

In this thesis, we consider the control frequency as a design parameter in learning-based control of uncertain systems. We investigate if and how fast feedback can compensate for a lack of knowledge about a system's dynamic behavior. Since this analysis can neither be performed with a fully continuous-time nor a fully discrete-time system description, we adopt a sampled-data control approach. In Chapters 3 and 4, we develop a framework combining GPs and BLR for probabilistic model learning of partially unknown continuous-time dynamics with robust sampled-data control. To this end, we derive sufficient conditions for robust stability based on the learned dynamics model. We formulate two optimization problems: one to compute the MCF and the other to optimize the control performance for a specific control frequency not smaller than the MCF. As a result, we can vary the frequency with which an uncertain system is controlled without having to adapt or relearn the model or sacrificing stability guarantees.

We use our framework to study the impact of control frequency compared to the amount of training data in simulation and real-world experiments. In doing so, we show that both the control frequency and the number of training samples must not fall below a certain threshold so that the system can still be stabilized. On the one hand, if the model uncertainty is too high, i.e., if the training data contains insufficient information, no stabilizing controller can be found for any frequency. On the other hand, if the control frequency is too small, the system cannot be stabilized regardless of the data, as shown by the experiments. This demonstrates that feedback and system identification must each be employed to some extent to control uncertain systems.

A major contribution is that we show and quantify the tradeoff between the control frequency and data regarding stability and performance: A lack of data can, to some extent, be made up for by a higher control frequency and vice versa. However, in the scenarios considered, a greater performance improvement can be achieved by increasing the control frequency than by collecting more data. We can conclude that despite the recent advances in learning for control, fast feedback is crucial for achieving high performance while satisfying robust stability guarantees.

In addition to allowing us to study the effects of control frequency and data independently, our robust learning-based sampled-data control approach offers another advantage: It provides a theoretically justified way to flexibly select the control frequency based on the uncertainty of the learned model. This can be leveraged, for example, to reduce the real-time computational and communication demand on low-resource systems such as quadrotors by running the controller only as fast as necessary.

Ultimately, any discussion about changing the control frequency is pointless if the existing hardware has a fixed control frequency. Hence, our results motivate increasing the availability of hardware components with flexible sampling rates.

6.2 Outlook

The theoretical analysis and experiments in this work can be continued in many directions. An obvious next step would be to extend the proposed framework to include tracking a reference trajectory while providing the same theoretical guarantees as for set-point stabilization. For this purpose, one could, for example, investigate the combination of sampled-data control with feedback linearization [23] or differential flatness [56]. Since trajectory tracking is usually done over a longer period, the uncertain dynamics model could then be updated online. This would, in principle, allow us to change the control frequency during operation based on the updated model using the methods presented in this thesis. However, the computation time for solving GEVPs prevents us from employing the methods proposed in Chapter 4 directly online, except for very small-dimensional systems. Instead, we would need to formulate the optimization problems in such a way that we can solve them in advance but use their solution online. This could be achieved, for instance, by considering model uncertainty as a disturbance similar to Section 4.5.1 and precomputing thresholds for the magnitude of the disturbance for different control frequencies. Online, only a scalar condition would then have to be evaluated. However, a disadvantage of this approach would be that it introduces additional conservatism.

Another interesting direction for future research would be to consider a direct connection between the amount of data or, more generally speaking, model complexity [57] and the control frequency. This is of high practical relevance for many state-of-the-art learning-based control methods such as GP-based model predictive control (GP-MPC) [2]. With GP-MPC, the accuracy of the learned dynamics model increases with the amount of inducing points of the GP. This amount, in turn, directly affects the computation time for model inference and, thus, the rate at which the optimal control problem can be solved online. In general, for learning-based control approaches that directly incorporate the learned model into the controller, the achievable control frequency is directly coupled to the complexity of the model. For these approaches to become more established outside of the academic world, it is essential to know to what degree we should rely on learning and to what degree on feedback to make optimal use of the available computational resources.

Bibliography

[1] Felix Berkenkamp and Angela P Schoellig. Safe and robust learning control with Gaussian processes. In *Proceedings of the European Control Conference*, pages 2496–2501, 2015.

[2] Chris J Ostafew, Angela P Schoellig, and Timothy D Barfoot. Robust constrained learning-based NMPC enabling reliable mobile robot path tracking. *The International Journal of Robotics Research*, 35(13):1547–1563, 2016.

[3] Alexander von Rohr, Matthias Neumann-Brosig, and Sebastian Trimpe. Probabilistic robust linear quadratic regulators with Gaussian processes. In *Proceedings of the Learning for Dynamics and Control Conference*, pages 324–335. PMLR, 2021.

[4] L. Brunke, M. Greeff, A. W. Hall, Z. Yuan, S. Zhou, J. Panerati, and A. P. Schoellig. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5(1):411–444, 2022.

[5] Armin Lederer, Alexandre Capone, Thomas Beckers, Jonas Umlauft, and Sandra Hirche. The impact of data on the stability of learning-based control. In *Proceedings of the Learning for Dynamics and Control Conference*, pages 623–635. PMLR, 2021.

[6] Abishek Padalkar et al. Open X-embodiment: Robotic learning datasets and RT-X models. *arXiv preprint arXiv 2310.08864*, 2023.

[7] Duy Nguyen-Tuong and Jan Peters. Model learning for robot control: A survey. *Cognitive Processing*, 12:319–340, 2011.

[8] John C Doyle, Bruce A Francis, and Allen R Tannenbaum. *Feedback control theory.* Courier Corporation, 2013.

[9] Xian-Ming Zhang, Qing-Long Han, Xiaohua Ge, Boda Ning, and Bao-Lin Zhang. Sampled-data control systems with non-uniform sampling: A survey of methods and trends. *Annual Reviews in Control*, 2023.

[10] Rudolf Emil Kalman. Contributions to the theory of optimal control. *Boletín de la Sociedad Matemática Mexicana*, 5(2):102–119, 1960.

[11] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning.* Springer, 2006.

[12] Lukas Brunke, Siqi Zhou, and Angela P. Schoellig. Barrier Bayesian linear regression: Online learning of control barrier conditions for safety-critical control of uncertain systems. In *Proceedings of the Learning for Dynamics and Control Conference*, pages 881–892. PMLR, 2022.

[13] Carl Edward Rasmussen, Christopher KI Williams, et al. *Gaussian processes for machine learning*, volume 1. Springer, 2006.

[14] Frank L Lewis, Draguna Vrabie, and Vassilis L Syrmos. *Optimal control*. John Wiley & Sons, 2012.

[15] Stuart M Melzer and Benjamin C Kuo. Sampling period sensitivity of the optimal sampled data linear regulator. *Automatica*, 7(3):367–370, 1971.

[16] Karl Worthmann, Marcus Reble, Lars Grune, and Frank Allgower. The role of sampling for stability and performance in unconstrained nonlinear model predictive control. *SIAM Journal on Control and Optimization*, 52(1):581–605, 2014.

[17] Laurentiu Hetel, Christophe Fiter, Hassan Omran, Alexandre Seuret, Emilia Fridman, Jean-Pierre Richard, and Silviu Iulian Niculescu. Recent developments on the stability of systems with aperiodic sampling: An overview. *Automatica*, 76:309–335, 2017.

[18] Emilia Fridman. *Introduction to time-delay systems: Analysis and control*. Springer, 2014.

[19] Emilia Fridman, Alexandre Seuret, and Jean-Pierre Richard. Robust sampled-data stabilization of linear systems: an input delay approach. *Automatica*, 40(8):1441–1446, 2004.

[20] Emilia Fridman. A refined input delay approach to sampled-data control. *Automatica*, 46(2):421–427, 2010.

[21] Alexandre Seuret. A novel stability analysis of linear systems under asynchronous samplings. *Automatica*, 48(1):177–182, 2012.

[22] Julian Berberich, Stefan Wildhagen, Michael Hertneck, and Frank Allgöwer. Data-driven analysis and control of continuous-time systems under aperiodic sampling. *IFAC-PapersOnLine*, 54(7):210–215, 2021.

[23] Xiaobing Dai, Armin Lederer, Zewen Yang, and Sandra Hirche. Can learning deteriorate control? Analyzing computational delays in Gaussian process-based event-triggered online learning. In *Proceedings of the Learning for Dynamics and Control Conference*, pages 445–457. PMLR, 2023.

[24] Alberto Maria Metelli, Flavio Mazzolini, Lorenzo Bisi, Luca Sabbioni, and Marcello Restelli. Control frequency adaptation via action persistence in batch reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, pages 6862–6873. PMLR, 2020.

[25] Lukas Hewing, Andrea Carron, Kim P Wabersich, and Melanie N Zeilinger. On a correspondence between probabilistic and robust invariant sets for linear systems. In *Proceedings of the European Control Conference*, pages 1642–1647, 2018.

[26] Stephen Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan. *Linear matrix inequalities in system and control theory.* SIAM, 1994.

[27] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization.* Cambridge university press, 2004.

[28] Richard Arnold Johnson, Dean W Wichern, et al. Applied multivariate statistical analysis. 2002.

[29] Jack Sherman and Winifred J. Morrison. Adjustment of an Inverse Matrix Corresponding to a Change in One Element of a Given Matrix. *The Annals of Mathematical Statistics*, 21(1):124 – 127, 1950.

[30] Armin Lederer, Jonas Umlauft, and Sandra Hirche. Uniform error bounds for Gaussian process regression with application to safe control. *Advances in Neural Information Processing Systems*, 32, 2019.

[31] David Duvenaud. *Automatic model construction with Gaussian processes.* PhD thesis, University of Cambridge, 2014.

[32] Dong C Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

[33] Payam Naghshtabrizi, Joao P Hespanha, and Andrew R Teel. Exponential stability of impulsive systems with application to uncertain sampled-data systems. *Systems & Control Letters*, 57(5):378–385, 2008.

[34] Emilia Fridman. Stability of linear descriptor systems with delay: A Lyapunov-based approach. *Journal of Mathematical Analysis and Applications*, 273(1):24–44, 2002.

[35] Jean-Jacques E Slotine and Weiping Li. *Applied nonlinear control.* Prentice Hall, 1991.

[36] Felix Berkenkamp, Angela P Schoellig, and Andreas Krause. Safe controller optimization for quadrotors with Gaussian processes. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 491–496, 2016.

[37] Felix Berkenkamp, Riccardo Moriconi, Angela P. Schoellig, and Andreas Krause. Safe learning of regions of attraction for uncertain, nonlinear systems with gaussian processes. In *Proceedings of the Conference on Decision and Control*, pages 4661–4666, 2016.

[38] Márcio F Braga, Víctor CS Campos, and Luciano Frezzatto. Improved discretization method for uncertain linear systems: a descriptor system based approach. In *Prceedings of the Conference on Decision and Control*, pages 7069–7074, 2019.

[39] Lihua Xie. Output feedback h control of systems with parameter uncertainty. *International Journal of control*, 63(4):741–750, 1996.

[40] Hak-Keung Lam and Frank HF Leung. Design and stabilization of sampled-data neural-network-based control systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 36(5):995–1005, 2006.

[41] Mei Yu, Long Wang, and Tianguang Chu. Robust stabilization of nonlinear sampled-data systems. In *Proceedings of the American Control Conference*, pages 3421–3426, 2005.

[42] Lawrence Narici and Edward Beckenstein. *Topological vector spaces.* CRC Press, 2010.

[43] Zhaocong Yuan, Adam W Hall, Siqi Zhou, Lukas Brunke, Melissa Greeff, Jacopo Panerati, and Angela P Schoellig. safe-control-gym: A unified benchmark suite for safe learning-based control and reinforcement learning in robotics. *IEEE Robotics and Automation Letters*, 7(4):11142–11149, 2022.

[44] Yalmip: A toolbox for modeling and optimization in MATLAB. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 284–289, 2004.

[45] MOSEK ApS. *The MOSEK optimization toolbox for MATLAB manual. Version 10.0.*, 2022.

[46] Crazyflie 2.1 — Bitcraze. URL https://www.bitcraze.io/products/crazyflie-2-1/. [Online; accessed 01-October-2023].

[47] Vicon Valkyrie. URL https://www.vicon.com/hardware/cameras/valkyrie/. [Online; accessed 01-October-2023].

[48] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, Andrew Y Ng, et al. ROS: An open-source robot operating system. In *ICRA Workshop on Open Source Software*, volume 3, 2009.

[49] Mark W Mueller, Michael Hamer, and Raffaello D'Andrea. Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1730–1736, 2015.

[50] Crazyswarm ROS package. URL `https://github.com/USC-ACTLab/crazyswarm`. [Online; accessed 20-October-2023].

[51] James A Preiss, Wolfgang Honig, Gaurav S Sukhatme, and Nora Ayanian. Crazyswarm: A large nano-quadcopter swarm. In *IEEE International Conference on Robotics and Automation*, pages 3299–3304, 2017.

[52] John R Dormand and Peter J Prince. A family of embedded runge-kutta formulae. *Journal of computational and applied mathematics*, 6(1):19–26, 1980.

[53] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. pages 3803–3810, 2018.

[54] Elia Kaufmann, Leonard Bauersfeld, and Davide Scaramuzza. A benchmark comparison of learned control policies for agile quadrotor flight. *International Conference on Robotics and Automation*, pages 10504–10510, 2022.

[55] Sergei Lupashin, Markus Hehn, Mark W Mueller, Angela P Schoellig, Michael Sherback, and Raffaello D'Andrea. A platform for aerial robotics research and demonstration: The flying machine arena. *Mechatronics*, 24(1):41–54, 2014.

[56] Adam W. Hall, Melissa Greeff, and Angela P. Schoellig. Differentially flat learning-based model predictive control using a stability, state, and input constraining safety filter. *IEEE Control Systems Letters*, 7:2191–2196, 2023.

[57] Lennart Ljung. Perspectives on system identification. *Annual Reviews in Control*, 34(1):1–12, 2010.