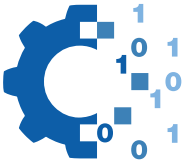

RWTH AACHEN UNIVERSITY

The present work was submitted to the Institute for Data Science in Mechanical Engineering.
Diese Arbeit wurde vorgelegt am Institut für Data Science im Maschinenbau.

Learning the stage cost in model predictive control

Lernen der Kostenfunktionen in der Modellprädiktiven Regelung

Master Thesis
Masterarbeit

Presented by / **Vorgelegt von**

Kenneth Goveas
Matr.Nr.: 404561

Supervised by / **Betreut von** Christian Fiedler M.Sc.
(Institute for Data Science in Mechanical Engineering)

Examiner / **Prüfer** Univ.-Prof. Dr. sc. Sebastian Trimpe
(Institute for Data Science in Mechanical Engineering)

Aachen, July 2, 2022

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne die Benutzung anderer als der angegebenen Hilfsmittel selbstständig verfasst habe; die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe des Literaturzitats gekennzeichnet.

Aachen, den July 2, 2022

Abstract

Model predictive control (MPC) is a control design technique in which a finite horizon optimal control problem (OCP) is solved at every time step. This yields a finite control sequence of which the first control is applied to the plant. The major factors that influence the performance of an MPC scheme are a good prediction model, sufficient horizon length, a suitable stage cost, and an appropriate terminal cost and constraint (collectively referred to as *terminal conditions*). The stage cost is an especially important design choice in the case of MPC without terminal conditions as the prediction model is usually the best one can obtain or computationally afford. This leads to the idea of learning the stage cost to improve closed loop behaviour. In this thesis, the stage cost is updated with the objective of maximising a performance measure. The parameters of the stage cost are updated through Bayesian optimisation (BO) by modelling the performance function as a Gaussian process (GP). We mathematically formulate necessary design considerations and accordingly propose a performance measure suitable for this method. Through simulation experiments, we demonstrate the validity of this performance measure, and also use it to learn the MPC stage cost for a simulated cart pole balancing system.

Kurzzusammenfassung

Die Modellprädiktive Regelung ist eine moderne Regelungsmethode, bei der ein Optimalsteuerungsproblem mit endlichem Zeithorizont in jedem Zeitschritt gelöst wird. Letzteres führt zu einer endlichen Sequenz von Regelungseingaben, aus der nur das erste Element in das zu regelnde System gegeben wird. Die wesentlichen Einflussfaktoren der Performanz dieser Methode umfassen ein gutes Prädiktionsmodell, ein hinreichend langer Zeithorizont im Optimalsteuerungsproblem, eine geeignete schrittweise Kostenfunktion sowie adäquate Endkosten und Endbedingungen. Die schrittweise Kostenfunktion spielt eine wesentliche Rolle insbesondere in der Modellprädiktiven Regelung ohne Endbedingungen oder -kosten, da das verwendete Prädiktionsmodell üblicherweise das beste verfügbare oder noch rechnerisch handbare Modell ist. Dies führt zu der Idee, die schrittweise Kostenfunktion zu lernen, um das Verhalten des geschlossenen Regelungskreises zu verbessern. In dieser Arbeit wird die schrittweise Kostenfunktion geändert mit dem Ziel der Maximimierung einer Performanzmetrik. Die Parameter der Kostenfunktion werden hierbei durch Bayessche Optimierung adaptiert, wobei die die Performanzfunktion durch einen Gaußschen Prozess modelliert wird. Wir beschreiben formal die notwendigen Designelemente und schlagen ein für diese Methode geeignetes Performanzmaß vor. Mittels Simulationsexperimenten zeigen wir, dass dieses Performanzmaß geeignet ist und verwenden es, um eine Kostenfunktion für die Modellprädiktive Regelung eines Wagen-Pendel-Systems zu lernen.

Contents

1 Introduction

Feedback control systems play an important role in a wide range of engineering applications such as industrial plants, vehicles, robots, etc. Classical linear control techniques such as PID control have been in use for a long time. However, the increasing complexity of dynamic systems has motivated the development of a variety of nonlinear control techniques too. Some simple examples include back-stepping and sliding mode control [2]. Advancements in embedded computing technology have made it possible to implement more complex nonlinear control schemes as well. One such popular method is *model predictive control* (MPC) which is known for its unique ability to accommodate practical restrictions such as state and control constraints.

1.1 What is model predictive control

MPC is an advanced control technique with a significant impact on industry and academic research [11]. Its working principle is to find an optimal solution for predicted state and control trajectories, given a suitable objective function. In contrast to most other closed loop control strategies, MPC has the ability to incorporate state and control constraints within its design. This quality has played a major role in the success of MPC [14], especially in process industries which are usually characterised by slow plant dynamics, enabling an easy MPC implementation. Additionally, MPC can easily be applied to systems for which the dynamics and constraints are nonlinear.

MPC is similar to traditional open loop optimal control in that it optimises predicted state and control trajectories to compute the control input. The difference is that the optimal control problem (OCP) is solved *online* and has a *finite* horizon. Solving this finite horizon OCP at each time step yields a finite optimal control sequence, of which the first control is applied to the plant [12]. This process is then

repeated at every time step, resulting in closed loop control with a receding horizon.

Using a discrete time model $\hat{x}_{t+1} = \hat{f}(\hat{x}_t, \hat{u}_t)$ of the plant dynamics (we distinguish this model from the actual plant dynamics $x_{t+1} = f(x_t, u_t)$ to accommodate the possibility of model inaccuracy), an OCP of horizon length N corresponding to a general MPC scheme can usually be formulated as follows,

$$[\hat{u}_0^* \cdots \hat{u}_{N-1}^*] = \arg \min_{\hat{u}_0 \cdots \hat{u}_{N-1}} \left[\sum_{n=0}^{N-1} L(\hat{x}_n, \hat{u}_n) + F(\hat{x}_N) \right] \tag{1.1a}$$

$$\text{such that} \quad \hat{x}_0 = x \tag{1.1b}$$

$$\hat{x}_{n+1} = \hat{f}(\hat{x}_n, \hat{u}_n) \; \forall \; n \in \{0 \cdots N-1\} \tag{1.1c}$$

$$\hat{x}_n \in X \; \forall \; n \in \{0 \cdots N-1\} \tag{1.1d}$$

$$\hat{u}_n \in U(\hat{x}_n) \; \forall \; n \in \{0 \cdots N-1\} \tag{1.1e}$$

$$\hat{x}_N \in X_F \tag{1.1f}$$

where $\hat{x}_n$ and $\hat{u}_n$ are predicted state and control vectors after n time steps, and x is the current actual state of the plant. The objective function is constructed from a stage cost L and a terminal cost F. The state and control constraints to be satisfied at all times are encoded in sets $X \subseteq \mathbb{R}^{n_x}$ and $U(\hat{x}_n) \subseteq \mathbb{R}^{n_u}$, and an additional terminal constraint $X_F \subseteq X$ may also be imposed. The above OCP is solved at every time step and the first optimal control $u = \hat{u}_0^*$ is applied to the plant.

1.2 Significance of terminal conditions

In fixed setpoint stabilisation problems, it is generally desired that a given state x_e act as a closed loop equilibrium which is *Lyapunov stable* (that is, for every $\epsilon > 0$, there must exist a $\delta_\epsilon > 0$ such that $\|x_0 - x_e\| < \delta_\epsilon$ implies $\|x_t - x_e\| < \epsilon$ for all $t \geq 0$) or preferably *asymptotically stable* (which means that in addition to Lyapunov stability, there exists a $\delta > 0$ such that $\|x_t - x_e\| \to 0$ as $t \to \infty$ if $\|x_0 - x_e\| < \delta$). Another desirable property is that the OCP (1.1) be feasible at all time steps. In particular, the MPC scheme must be *recursively feasible* (feasibility at time $t = 0$ must imply feasibility at all times $t \geq 0$). In the absence of model inaccuracies, stability and recursive feasibility are usually guaranteed through a suitable choice of terminal cost F and terminal constraint X_F (collectively referred to as *terminal*

conditions).

As a simple example, the terminal constraint set can be chosen to only include the target equilibrium $X_F = \{x_e\}$, and the terminal cost can be set to zero [7, Section 5.2]. In this way, stability and recursive feasibility are guaranteed for those states x from which the equilibrium x_e is reachable in N time steps. Another example is provided in [9, Chapter 2] for a linear system $x_{t+1} = Ax_t + Bu_t$ with quadratic stage cost $L(\hat{x}, \hat{u}) = \hat{x}^\intercal \hat{Q} \hat{x} + \hat{u}^\intercal \hat{R} \hat{u}$. A quadratic terminal cost $F(\hat{x}) = \hat{x}^\intercal \hat{S} \hat{x}$ was used, with $\hat{S}$ computed from the discrete algebraic Riccati equation (DARE). The terminal constraint set X_F was chosen to be the set of all states starting from which a linear quadratic regulator (LQR) applied at all future time steps would satisfy the constraints. This MPC scheme was shown to be stable and recursively feasible for those states x from which X_F is reachable in N time steps.

1.3 Model predictive control without terminal conditions

As is evident from Section 1.2, terminal conditions play an important role in MPC design by making it easy to provide guarantees of stability and recursive feasibility. With terminal conditions included, the prediction horizon may also be shortened if required. While this does shrink the feasible set for the OCP, it retains the guarantees of stability and recursive feasibility [7, Section 7.4]. Despite these advantages, there are several reasons why MPC without terminal conditions could be preferred in practice. For instance, designing terminal conditions can often be very challenging, and as discussed in Section 1.2, they restrict the feasible set for the OCP. Terminal conditions (in particular, terminal constraints) can also lead to problems while using a numerical solver for the OCP. A more detailed comparison of MPC schemes with and without terminal conditions can be found in [7, Section 7.4].

The closed loop performance of an MPC scheme without terminal conditions depends on three major factors – an accurate prediction model, an appropriate stage cost, and a sufficiently long prediction horizon. Of these, the prediction model is usually the best one can obtain, or otherwise afford with the available computational resources, and as it is desirable to keep the prediction horizon as short as possible, the stage cost becomes an especially important design ingredient. While a quadratic stage cost is often chosen in practice due to its simplicity, this strategy might fail in the absence of terminal conditions as demonstrated by an example in [13]. As

also illustrated in a different example in [7, Section 6.6], it is sometimes possible to modify the stage cost to make an MPC scheme successfully stabilise the setpoint in closed loop.

1.4 Machine learning in model predictive control

Recently, various machine learning techniques have been used to augment MPC designs [8]. As accurate prediction models are essential to the success of MPC, many learning-based methods such as [1] and [18] focused on learning a model of the plant dynamics. However, some other learning-based approaches update other components of the MPC scheme while leaving the prediction model unchanged. One such example is the learning model predictive controller (LMPC) proposed by Rosolia and Borrelli [16] in which the concept of iterative learning control [3] is used to learn appropriate terminal conditions while using a fixed prediction model. Data from previous runs of a batch process were used to improve a terminal cost and construct a safe terminal constraint set for the next run. Even more recently, Gros and Zanon [6] interpreted MPC as a policy representation and used traditional reinforcement learning techniques to update a set of tuning parameters. This process can influence any of the basic components of an MPC scheme and can in particular, be used to update the OCP stage cost for MPC without terminal conditions. Yet another approach by Marco, et al [10] used *Bayesian optimisation* (BO) [4] to update the cost function to optimise a performance measure modelled as a *Gaussian process* (GP) [15]. Marco, et al used this technique to update the weight matrices of an LQR controller, which can be interpreted as a special case of MPC.

1.5 Approach for stage cost learning

The discussion in Sections 1.3 and 1.4 immediately leads to the idea of using a learning-based method to update the MPC stage cost in the absence of terminal conditions. One such method is that proposed by Gros and Zanon [6] and was the approach used in a previous research project at DSME in this direction. In this setting, a nominal prediction model and nominal stage cost are available while the true plant model is unknown. The overarching idea is to update the stage cost in such a way that the closed loop system mimics the behaviour of the nominal plant,

had it been controlled using the nominal stage cost. However, ambiguities were observed in [6] which made it difficult to reproduce their experimental outcomes. Most importantly, the update rule for the stage cost was altogether independent of the nominal stage cost, making it unclear what stage cost was actually being learned by the algorithm. Besides, no visible advantage could be observed in experiments in which the stage cost was updated.

An alternative stage cost learning method is used in this thesis, based on the work of Marco, et al [10]. The stage cost is adjustable by varying a vector of tuning parameters θ. Different values of θ result in different closed loop behaviour. A suitable performance measure $J(\theta)$ is defined, whose value at any given θ can be estimated from an observed closed loop trajectory. The problem of stage cost learning is interpreted as one of finding the optimal parameters θ^* which maximise the performance $J(\theta)$. This is accomplished using BO by modelling $J(\theta)$ as a GP. It is important to note that for the same parameters θ, the observed closed loop trajectory generally has a significant dependence on the initial state vector. Furthermore, it may also be affected by any noise inputs present in the system. Therefore, the true performance value $J(\theta)$ cannot accurately be inferred from an observed closed loop trajectory, but can only be estimated with some uncertainty. It is also not possible to obtain performance measurements for a large number of parameter vectors as each individual evaluation is expensive (requiring the generation of a complete closed loop trajectory). Thus, the objective is to optimise an unknown function with few evaluations, each corrupted by uncertainty, for which BO is a well-suited strategy [4].

1.6 Contributions from this thesis

The problem of stage cost learning is interpreted as a controller tuning problem and a tuning method using Bayesian optimisation of a performance measure is proposed. A major outcome of this project was the development of a suitable performance measure which exhibits less dependence on the starting state of the system. This is important when learning is carried out iteratively over the span of several episodes, each of which may begin with an arbitrary state. This property will also be relevant if the learning technique is to be further developed to update the stage cost at regular intervals in a *single* episode, where separate time intervals in the same episode can be treated as separate iterations despite not sharing the same initial state.

Simulation experiments were carried out to verify the suitability of the proposed performance measure and to learn the MPC stage cost for a simple nonlinear system (one-dimensional pole balancing).

2 Background

In this chapter, we provide some theoretical background on MPC as well as BO. We begin with a discussion of the general mathematical formulation of MPC and introduce the concepts of stability and recursive feasibility. We also state how terminal conditions are used to guarantee these properties. Following this, we introduce the method of BO focusing on its two major aspects, namely, the construction of a posterior model of an unknown function from observed measurements, and the selection of the next point of evaluation using acquisition functions.

2.1 Model predictive control

An MPC scheme operates by solving a finite horizon OCP at each time step to compute an optimal control sequence, of which the first control is applied to the plant. In this section, we present the mathematical formulation of such a general MPC design.

2.1.1 Plant dynamics

In most control design paradigms, the plant being controlled is modelled by an ordinary differential equation. However, as control signals are piecewise constant in practice, it is common in MPC literature to model the plant by a discrete time equation instead. A discrete time equation may be obtained, for instance, by approximating time derivatives by finite differences. Thus, we consider a plant governed by the following nonlinear difference equation,

$$x_{t+1} = f(x_t, u_t) + w_t \quad \text{where} \quad w_t \sim \mathcal{N}(0, \Sigma_w) \tag{2.1}$$

where $x_t \in \mathbb{X} = \mathbb{R}^{n_x}$ and $u_t \in \mathbb{U} = \mathbb{R}^{n_u}$ denote the state and control vectors at time t, and w_t is a random zero mean Gaussian noise. For mathematical simplicity, we

assume that $f(0,0) = 0$ and that we wish to have a stable closed loop equilibrium at $x_e = 0$.

2.1.2 Prediction model

To predict state and control trajectories consistent with the underlying plant dynamics (2.1), an MPC scheme uses a prediction model of the following form,

$$\hat{x}_{t+1} = \hat{f}(\hat{x}_t, \hat{u}_t) \tag{2.2}$$

where $\hat{x}_t \in \mathbb{X}$ and $\hat{u}_t \in \mathbb{U}$ denote the predicted state and control vectors. The prediction model $\hat{f}$ is usually a practical approximation to the true plant model f which is not accurately known in general. In this chapter, we shall assume that $f = \hat{f}$ and $w_t = 0$ so that the model (2.2) exactly matches the plant dynamics (2.1), as this allows for an easy discussion of stability and recursive feasibility. However, we do not make this assumption in the rest of this thesis.

2.1.3 Cost function

In fixed setpoint tracking problems, the objective of control design is usually to steer the plant to the state $x_e = 0$. This is done by optimising finite predicted state and control trajectories at each time step, and applying the first control vector in the optimal sequence. The cost function to be minimised at each time is usually of the following form,

$$V(\hat{x}_0 \cdots \hat{x}_N, \hat{u}_0 \cdots \hat{u}_{N-1}) = \sum_{n=0}^{N-1} L(\hat{x}_n, \hat{u}_n) + F(\hat{x}_N). \tag{2.3}$$

Here $\hat{x}_n \in \mathbb{X}$ and $\hat{u}_n \in \mathbb{U}$ denote the predicted state and control after n time steps. Thus, $[\hat{x}_0 \cdots \hat{x}_N]$ and $[\hat{u}_0 \cdots \hat{u}_{N-1}]$ constitute the predicted state and control trajectories corresponding to a finite horizon N. The stage cost $L : \mathbb{X} \times \mathbb{U} \to \mathbb{R}^+$ and the terminal cost $F : \mathbb{X} \to \mathbb{R}^+$ are usually chosen to be positive definite functions.

2.1.4 State and control constraints

The predicted trajectories $[\hat{x}_0 \cdots \hat{x}_N]$ and $[\hat{u}_0 \cdots \hat{u}_{N-1}]$ must satisfy any state and control constraints imposed by the problem. First of all, these trajectories must be

consistent with the plant model (2.2). This condition already yields the following constraints,

$$\hat{x}_0 = x \tag{2.4a}$$
$$\hat{x}_{n+1} = \hat{f}(\hat{x}_n, \hat{u}_n) \;\forall\; n \in \{0 \cdots N-1\} \tag{2.4b}$$

where x is the current state vector of the plant. In addition to these, more constraints may also be imposed, depending on the problem at hand,

$$\hat{x}_n \in X \;\forall\; n \in \{0 \cdots N-1\} \tag{2.5a}$$
$$\hat{u}_n \in U(\hat{x}_n) \;\forall\; n \in \{0 \cdots N-1\} \tag{2.5b}$$

where $X \subseteq \mathbb{X}$ and $U(\hat{x}_n) \subseteq \mathbb{U}$ represent the sets of allowed state and control vectors at any time step. Sometimes, a terminal constraint is also imposed upon the final predicted state $\hat{x}_N$,

$$\hat{x}_N \in X_F \tag{2.6}$$

where $X_F \subseteq X$ usually encodes a tighter constraint on $\hat{x}_N$ than (2.5).

2.1.5 Optimal control problem

Putting together the ingredients described in Sections 2.1.2, 2.1.3, and 2.1.4, we can formulate the complete OCP for a general MPC scheme. If x is the current state vector, then the control $u = \pi(x)$ is computed as follows,

$$u = \pi(x) = \hat{u}_0^* \tag{2.7a}$$
$$\text{where} \quad [\hat{u}_0^* \cdots \hat{u}_{N-1}^*] = \arg\min_{\hat{u}_0 \cdots \hat{u}_{N-1}} \left[\sum_{n=0}^{N-1} L(\hat{x}_n, \hat{u}_n) + F(\hat{x}_N) \right] \tag{2.7b}$$
$$\text{such that} \quad \hat{x}_0 = x \tag{2.7c}$$
$$\hat{x}_{n+1} = \hat{f}(\hat{x}_n, \hat{u}_n) \;\forall\; n \in \{0 \cdots N-1\} \tag{2.7d}$$
$$\hat{x}_n \in X \;\forall\; n \in \{0 \cdots N-1\} \tag{2.7e}$$
$$\hat{u}_n \in U(\hat{x}_n) \;\forall\; n \in \{0 \cdots N-1\} \tag{2.7f}$$
$$\hat{x}_N \in X_F. \tag{2.7g}$$

2.1.6 Recursive feasibility

To be able to compute the control $u = \pi(x)$ for a given state x, it is necessary for the OCP (2.7) to admit a feasible solution. In particular, we wish the MPC scheme to be *recursively feasible*, which means that feasibility at any time step should imply feasibility at all future times. We define recursive feasibility following the definition of Grüne and Pannek [7, Section 7.1].

Definition 1. *A state $x \in X$ is said to be a* feasible state *if the OCP* (2.7) *admits a solution $[\hat{u}_0 \cdots \hat{u}_{N-1}]$ satisfying all the imposed constraints. The* feasible set *for the MPC scheme* (2.7) *is the set $\mathcal{F} = \{x \in X \mid x \text{ is a feasible state}\}$.*

Definition 2. *The MPC scheme* (2.7) *is said to be* recursively feasible *on a set $\mathcal{A} \subseteq \mathcal{F}$ if $f(x, \pi(x)) \in \mathcal{A}$ for all states $x \in \mathcal{A}$.*

The typical method to guarantee recursive feasibility on the entire feasible set $\mathcal{F}$ is to impose an appropriate terminal constraint X_F. Specifically, if X_F is *controlled positively invariant* (CPI) according to Definition 3, then the MPC scheme is recursively feasible for all feasible states.

Definition 3. *A set $\mathcal{X} \subseteq X$ is said to be* controlled positively invariant *(CPI) if for every $x \in \mathcal{X}$, there exists a $u \in U(x)$ such that $f(x, u) \in \mathcal{X}$.*

Theorem 1. *If the terminal constraint set X_F is CPI, then the MPC scheme* (2.7) *is recursively feasible on the entire feasible set $\mathcal{F}$.*

Proof. If $x \in \mathcal{F}$, then there exists an optimal solution $[\hat{u}_0^* \cdots \hat{u}_{N-1}^*]$ to the OCP (2.7) and $\pi(x) = \hat{u}_0^*$. Also, the predicted states $[\hat{x}_0^* \cdots \hat{x}_N^*]$ corresponding to the optimal solution must satisfy all the imposed constraints. Therefore, $\hat{x}_N^* \in X_F$ and because X_F is CPI, there exists a $\hat{u}_N \in U(\hat{x}_N^*)$ such that $f(\hat{x}_N^*, \hat{u}_N) \in X_F$. After applying the control $\pi(x)$, the state vector at the next time will be $f(x, \pi(x)) = \hat{x}_1^*$. At this time step, it is clear that $[\hat{u}_1^* \cdots \hat{u}_{N-1}^*, \hat{u}_N]$ is a feasible solution to the OCP (2.7). This establishes that $f(x, \pi(x)) \in \mathcal{F}$ and therefore, the MPC scheme is recursively feasible over $\mathcal{F}$. ■

2.1.7 Lyapunov stability

Constrained systems controlled by an MPC scheme generally exhibit nonlinear closed loop behaviour, and therefore, Lyapunov stability theory is the preferred tool for

stability analysis. We define stability of the closed loop equilibrium $x_e = 0$ following Slotine and Li [17, Section 3.2].

Definition 4. *The equilibrium $x_e = 0$ of the closed loop system $x_{t+1} = f(x_t, \pi(x_t))$ is said to be* Lyapunov stable *if for every $\epsilon > 0$, there exists a $\delta_\epsilon > 0$ such that $\|x_0\| < \delta_\epsilon$ implies $\|x_t\| < \epsilon$ for all $t \geq 0$. We say that the equilibrium is* asymptotically stable *if in addition to Lyapunov stability, there exists a $\delta > 0$ such that $\|x_0\| < \delta$ implies $\|x_t\| \to 0$ as $t \to \infty$.*

As with recursive feasibility, stability can also be guaranteed through the choice of suitable terminal conditions. Mayne, et al [12] derived conditions on the terminal constraint set X_F, satisfying which guarantees asymptotic stability.

Theorem 2. *If the terminal constraint set $X_F \subseteq X$ contains the equilibrium $x_e = 0$ and is such that $f(x, \pi(x)) \in X_F$ for all $x \in X_F$, and if the terminal cost satisfies the condition $F(f(x, \pi(x))) - F(x) \leq -L(x, \pi(x))$ for all $x \in X_F$, then $x_e = 0$ is asymptotically stable.*

Proof. A common method of proof of stability in the MPC literature uses the total cost (2.3) as a Lyapunov function. Under the above conditions, asymptotic stability can be established with this method. For details of the proof, we refer the reader to [12]. ■

2.2 Bayesian optimisation

Bayesian optimisation (BO) is a learning-based approach for optimising an unknown function $J(\theta)$ where $\theta \in \mathbb{P} \subseteq \mathbb{R}^{n_\theta}$ and $J : \mathbb{P} \to \mathbb{R}$. It is well-suited for functions J which are expensive to evaluate, typically limiting the total number of evaluations to a few hundred [4]. The individual function evaluations may also be corrupted by noise. Given m previous evaluations $(\theta_1, \hat{J}_1) \cdots (\theta_m, \hat{J}_m)$, two major steps are carried out in any iteration. First, a posterior model of $J(\theta)$ is fitted, conditioned on the m evaluations. This model is usually a Gaussian process (GP), which provides a posterior probability distribution of $J(\theta)$ for each value of θ. After this, the next sampling point θ_{m+1} is selected by optimising an appropriate *acquisition function.*

2.2.1 Gaussian processes

Gaussian process (GP) regression is a method of modelling an unknown function $J : \mathbb{P} \to \mathbb{R}$ by describing it as a distribution over functions. For any arbitrary sequence $[\theta_1 \cdots \theta_m]$, the vector $[J(\theta_1) \cdots J(\theta_m)]^\intercal$ is described as having a zero mean Gaussian distribution (it is possible to construct GP models with non-zero mean, but we assume a zero mean for simplicity). Such a GP model is completely characterised by its *kernel* function $k : \mathbb{P} \times \mathbb{P} \to \mathbb{R}$ defined as follows,

$$k(\theta_i, \theta_j) = \mathbb{E}[J(\theta_i)J(\theta_j)]. \tag{2.8}$$

Thus, the vector $J(\theta_{1:m}) = [J(\theta_1) \cdots J(\theta_m)]^\intercal$ has the following distribution,

$$J(\theta_{1:m}) \sim \mathcal{N}(0, k(\theta_{1:m}, \theta_{1:m})) \tag{2.9}$$

where $k(\theta_{1:m}, \theta_{1:m}) = [k(\theta_1, \theta_1) \cdots k(\theta_1, \theta_m); \cdots ; k(\theta_m, \theta_1) \cdots k(\theta_m, \theta_m)]$. Suppose noisy evaluations of $J(\theta)$ performed at m points $\theta_1 \cdots \theta_m$ yielded outcomes $\hat{J}_1 \cdots \hat{J}_m$ where,

$$\hat{J}_i = J(\theta_i) + v_i \quad \text{where} \quad v_i \sim \mathcal{N}\Big(0, \sigma_v^2\Big) \tag{2.10}$$

then the posterior distribution of $J(\theta)$ conditioned on these observations is the following,

$$J(\theta) \sim \mathcal{N}\Big(\mu_m(\theta), \sigma_m^2(\theta)\Big) \tag{2.11a}$$

$$\text{where} \quad \mu_m(\theta) = k(\theta, \theta_{1:m})^\intercal \Big(k(\theta_{1:m}, \theta_{1:m}) + \sigma_v^2 I\Big)^{-1} \hat{J}_{1:m} \tag{2.11b}$$

$$\sigma_m^2(\theta) = k(\theta, \theta) - k(\theta, \theta_{1:m})^\intercal \Big(k(\theta_{1:m}, \theta_{1:m}) + \sigma_v^2 I\Big)^{-1} k(\theta, \theta_{1:m}) \tag{2.11c}$$

where $k(\theta, \theta_{1:m}) = [k(\theta, \theta_1) \cdots k(\theta, \theta_m)]^\intercal$. The proof can be found in [15, Section 2.2].

2.2.2 Acquisition functions

Having constructed the GP posterior (2.11) from the previous m evaluations, the next step is to select the point θ_{m+1} at which to sample next. This selection is done

by maximising an appropriately chosen acquisition function $a : \mathbb{P} \to \mathbb{R}$,

$$\theta_{m+1} = \arg\max_{\theta \in \mathbb{P}} a(\theta). \tag{2.12}$$

One of the commonly used acquisition functions is the *expected improvement* (EI) which is defined as the expected value of increase in the observed maximum value of J. Suppose $\hat{J}_m^{\max}$ is the highest value of J observed in the previous m evaluations, or $\hat{J}_m^{\max} = \max(\hat{J}_1 \cdots \hat{J}_m)$. If we choose θ as the next point to sample, and observe the function value $J(\theta)$, then the new observed maximum is either $J(\theta)$ or $\hat{J}_m^{\max}$ depending on whether or not $J(\theta) > \hat{J}_m^{\max}$. Thus, the net improvement in the observed maximum is $\max(0, J(\theta) - \hat{J}_m^{\max})$. The rationale behind the EI acquisition function is to maximise the expected value of this improvement,

$$a(\theta) = \mathbb{E}\Big[\max\Big(0, J(\theta) - \hat{J}_m^{\max}\Big)\Big]. \tag{2.13}$$

Another well-known acquisition function is the *entropy search* (ES) which seeks to maximise the gain in information about the location of the optimum. The GP posterior (μ_m, σ_m^2) encodes a probability distribution p_m for the optimum θ^*. Let $p_{m,\theta,J(\theta)}$ denote this probability distribution if we then sample the point θ and observe $J(\theta)$. Then the ES acquisition function is defined as the expected entropy decrease due to sampling at θ,

$$a(\theta) = H(p_m) - \mathbb{E}\Big[H\Big(p_{m,\theta,J(\theta)}\Big)\Big]. \tag{2.14}$$

Loosely speaking, the EI acquisition function samples points which are themselves likely to optimise $J(\theta)$ whereas ES finds points by sampling at which we learn the most about the location of the optimum. EI is a good strategy in scenarios where individual evaluations matter (for example, an online optimisation of controller tuning parameters). ES on the other hand, is suitable for instance, in prototyping experiments, where suboptimal performances are acceptable as the objective is only to learn as much as possible about the optimum. Other popular acquisition functions also exist, a more detailed description of which can be found in [4].

3 Problem statement

The problem of learning the MPC cost function can be interpreted as a controller tuning problem. In this chapter, we explain how this is so, formulate the problem setting, and precisely state the objective of cost function learning in mathematical terms.

3.1 Plant dynamics

We consider the same plant (2.1) described in Section 2.1.1. Unlike the discussion in Chapter 2 however, we do not assume that $w_t = 0$ or that f is exactly known. Therefore, an approximate plant model shall be used in control design. We do still assume that $f(0, 0) = 0$ and that $x_e = 0$ is the desired closed loop equilibrium. We also assume that x_t can be measured without uncertainty, and is therefore available to compute the control u_t in a feedback control system.

3.2 Control system

In our problem setting, the plant (2.1) is controlled by the MPC scheme $u_t = \pi_\theta(x_t)$ described below, with the objective of stabilising the setpoint at $x_e = 0$,

$$u = \pi_\theta(x) = \hat{u}_0^* \tag{3.1a}$$

$$\text{where} \quad [\hat{u}_0^* \cdots \hat{u}_{N-1}^*] = \arg\min_{\hat{u}_0 \cdots \hat{u}_{N-1}} \left[\sum_{n=0}^{N-1} L_\theta(\hat{x}_n, \hat{u}_n) + F_\theta(\hat{x}_N) \right] \tag{3.1b}$$

$$\text{such that} \quad \hat{x}_0 = x \tag{3.1c}$$

$$\hat{x}_{n+1} = \hat{f}(\hat{x}_n, \hat{u}_n) \; \forall \; n \in \{0 \cdots N-1\} \tag{3.1d}$$

$$\hat{x}_n \in X \; \forall \; n \in \{0 \cdots N-1\} \tag{3.1e}$$

$$\hat{u}_n \in U(\hat{x}_n) \; \forall \; n \in \{0 \cdots N-1\} \tag{3.1f}$$

$$\hat{x}_N \in X_F. \tag{3.1g}$$

Here, $\hat{f}$ is the prediction model used by the controller and will not, in general, match the true system model f due to modelling errors, parameter uncertainties, etc. The constraint sets $X \subseteq \mathbb{X}$, $U(\hat{x}_n) \subseteq \mathbb{U}$, and $X_F \subseteq X$ are as described in Section 2.1.4. Note that while the constraints are imposed on the predicted state and control, model mismatch and process noise can cause the actual system to violate them nonetheless. If they must necessarily be satisfied by the true system, tighter constraints may be required in (3.1). However, these considerations are beyond the scope of this thesis. We focus on the cost functions L_θ and F_θ which depend on a parameter vector $\theta \in \mathbb{P} \subseteq \mathbb{R}^{n_\theta}$. Different values of θ will yield different closed loop state and control trajectories. Thus, the stage cost can be updated by tuning the controller parameters θ.

3.3 Performance measure

In addition to the plant and controller described in Sections 3.1 and 3.2, we assume that a performance measure $J : \mathbb{X} \times \mathbb{P} \to \mathbb{R}$ is also defined, which maps every combination of initial state x_{in} and parameter vector θ to a closed loop performance value $J(x_{\text{in}}, \theta)$. We assume that while $J(x_{\text{in}}, \theta)$ cannot directly be computed from

x_{in} and θ, it can be estimated from an observed closed loop trajectory,

$$J(x_{\text{in}}, \theta) \approx \hat{J}(x_0 \cdots x_{T-1}, u_0 \cdots u_{T-1}) \tag{3.2}$$

where $[x_0 \cdots x_{T-1}]$ and $[u_0 \cdots u_{T-1}]$ are the state and control trajectories observed when the plant (2.1) starts in state x_{in} and is allowed to run for T time steps (which shall henceforth be referred to as one *episode*) with parameters θ used by the controller (3.1). The dependence of J on x_{in} is apparent from the fact that for the same parameters θ, different initial states x_{in} can potentially lead to very different closed loop trajectories.

3.4 Objective of cost function learning

The cost functions L_θ and F_θ may be updated by modifying the parameters θ. Learning the appropriate cost functions can therefore, be interpreted as finding an optimal parameter vector θ^*. The objective of this controller tuning problem is twofold. First of all, a suitable performance function J must be defined which adequately captures the notion of "good performance" while also being independent of x_{in}. In this way, the performance becomes solely a property of the control system, and not the initial conditions of an episode. Having chosen such a performance measure, the objective is then to find the optimal parameters θ^* that maximise $J(x_{\text{in}}, \theta)$. The independence of $J(x_{\text{in}}, \theta)$ from x_{in} ensures that the value of θ^* does not vary with x_{in}, so that the problem is well-posed.

3.5 Simple example

As an example of the controller tuning problem described in this chapter, consider a plant governed by the following linear dynamic equation,

$$x_{t+1} = Ax_t + Bu_t + w_t \quad \text{where} \quad w_t \sim \mathcal{N}(0, \Sigma_w) \tag{3.3}$$

where the matrices A and B are not accurately known and approximations $\hat{A}$ and $\hat{B}$ are used to construct the following prediction model instead,

$$\hat{x}_{t+1} = \hat{A}\hat{x}_t + \hat{B}\hat{u}_t. \tag{3.4}$$

Consider the following quadratic cost function to be used to formulate an MPC scheme to stabilise the plant (3.3) at $x_e = 0$,

$$V_\theta(\hat{x}_0 \cdots \hat{x}_N, \hat{u}_0 \cdots \hat{u}_{N-1}) = \sum_{n=0}^{N-1} \left(\|\hat{x}_n\|_{\hat{Q}}^2 + \|\hat{u}_n\|_{\hat{R}}^2 \right) + \|\hat{x}_N\|_{\hat{S}}^2 \tag{3.5}$$

for which the tuning parameters are $\theta = \{\hat{Q}, \hat{R}, \hat{S}\}$. The MPC scheme is as follows,

$$u = \pi_\theta(x) = \hat{u}_0^* \tag{3.6a}$$

$$\text{where} \quad [\hat{u}_0^* \cdots \hat{u}_{N-1}^*] = \arg \min_{\hat{u}_0 \cdots \hat{u}_{N-1}} \left[\sum_{n=0}^{N-1} \left(\|\hat{x}_n\|_{\hat{Q}}^2 + \|\hat{u}_n\|_{\hat{R}}^2 \right) + \|\hat{x}_N\|_{\hat{S}}^2 \right] \tag{3.6b}$$

$$\text{such that} \quad \hat{x}_0 = x \tag{3.6c}$$

$$\hat{x}_{n+1} = \hat{A}\hat{x}_n + \hat{B}\hat{u}_n \;\forall\; n \in \{0 \cdots N-1\}. \tag{3.6d}$$

A commonly used performance measure is the expected average quadratic cost,

$$J(x_{\text{in}}, \theta) = \mathbb{E}_{x_0 = x_{\text{in}}, u_t = \pi_\theta(x_t)} \left[-\frac{1}{T} \sum_{t=0}^{T-1} \left(\|x_t\|_Q^2 + \|u_t\|_R^2 \right) \right] \tag{3.7}$$

which can be estimated from an observed closed loop trajectory as follows,

$$\hat{J}(x_0 \cdots x_{T-1}, u_0 \cdots u_{T-1}) = -\frac{1}{T} \sum_{t=0}^{T-1} \left(\|x_t\|_Q^2 + \|u_t\|_R^2 \right). \tag{3.8}$$

Note that $\hat{Q}$ and $\hat{R}$ in (3.6) are not identical to Q and R in (3.7) and (3.8). The latter are constants that define J while the former are tuning parameters. The objective of stage cost learning is to find the parameters θ^* that maximise the performance measure (3.7). As shown in Chapter 4 however, this performance function has a pronounced dependence on x_{in} and may therefore, not be a suitable choice. Instead, an alternative performance measure is proposed in Chapter 4 and the details of the learning algorithm are presented.

4 Learning approach

In this chapter, we describe the technique used to learn the optimal parameters θ^* for the MPC scheme (3.1). The performance function is modelled as a GP and the parameters θ are updated after every episode using BO. We also show that the quadratic performance measure (3.7) has a significant dependence on the initial state x_{in} and suggest an alternative.

4.1 Algorithm for cost function learning

The process of learning the optimal parameters θ^* spans multiple episodes. In the m-th episode, the system (2.1) starts in a random initial state x_{in}^m and is controlled using the MPC scheme (3.1) with parameters θ_m. The system is allowed to run for a fixed number of time steps T, after which the observed state and control trajectories are used to compute an estimate $\hat{J}_m$ of the performance. We assume that the estimates $\hat{J}_m$ are simply evaluations of the true performance $J(\theta_m)$ corrupted by Gaussian random error,

$$\hat{J}_m = J(\theta_m) + v_m \quad \text{where} \quad v_m \sim \mathcal{N}\Big(0, \sigma_v^2\Big). \tag{4.1}$$

The performance $J(\theta)$ is modelled as a GP with a *squared exponential* (SE) kernel,

$$k(\theta_i, \theta_j) = \sigma^2 \exp\left[-\frac{1}{2}\|\theta_i - \theta_j\|_W^2\right] \tag{4.2}$$

The posterior distribution of $J(\theta)$ conditioned on the observations up to the m-th episode can be computed from (2.11),

$$\mu_m(\theta) = k(\theta, \theta_{1:m})^\intercal \Big(k(\theta_{1:m}, \theta_{1:m}) + \sigma_v^2 I\Big)^{-1} \hat{J}_{1:m} \tag{4.3a}$$

$$\sigma_m^2(\theta) = k(\theta, \theta) - k(\theta, \theta_{1:m})^\intercal \Big(k(\theta_{1:m}, \theta_{1:m}) + \sigma_v^2 I\Big)^{-1} k(\theta, \theta_{1:m}) \tag{4.3b}$$

The parameter θ_{m+1} to be used in the next episode is then computed by maximising the EI acquisition function (2.13) and the process is repeated,

$$\theta_{m+1} = \arg\max_{\theta \in \mathbb{P}} \mathbb{E}\Big[\max\Big(0, J(\theta) - \hat{J}_m^{\max}\Big)\Big]. \tag{4.4}$$

4.2 Choice of performance measure

The performance measure is a crucial design choice for the controller tuning problem outlined in Chapter 3. In the initial portion of this project, the quadratic performance function (3.7) was used. However, as we prove further in this section, this function has a significant dependence on the initial state x_{in} of an episode. As already mentioned in Section 3.4, it is desirable to design J independent of x_{in}. Unfortunately, designing a performance measure that retains this property for any arbitrary plant (2.1) and controller (3.1) is extremely challenging, if not altogether impossible. For practical purposes therefore, it suffices to design J in a manner such that it "does not depend too much" on the initial state x_{in}. In this section, we formalise this notion and design a performance measure that satisfies this property when the plant is linear and the cost functions satisfy a homogeneity condition. This design is also applicable to nonlinear plants operating in the vicinity of the target equilibrium, as they may be approximated by their linearised forms in this domain. We begin by stating a few theorems which are necessary for the remaining discussion.

Theorem 3. *Let $V_\theta : \mathbb{X}^{N+1} \times \mathbb{U}^N \to \mathbb{R}^+$ be defined as follows,*

$$V_\theta(\hat{x}_0 \cdots \hat{x}_N, \hat{u}_0 \cdots \hat{u}_{N-1}) = \sum_{n=0}^{N-1} L_\theta(\hat{x}_n, \hat{u}_n) + F_\theta(\hat{x}_N) \tag{4.5}$$

where $L_\theta : \mathbb{X} \times \mathbb{U} \to \mathbb{R}^+$ and $F_\theta : \mathbb{X} \to \mathbb{R}^+$ are such that there exists a function $\kappa : \mathbb{R} \to \mathbb{R}^+$ for which the following homogeneity condition is satisfied,

$$L_\theta(\alpha\hat{x}, \alpha\hat{u}) = \kappa(\alpha) L_\theta(\hat{x}, \hat{u}) \tag{4.6a}$$

$$F_\theta(\alpha\hat{x}) = \kappa(\alpha) F_\theta(\hat{x}) \tag{4.6b}$$

for all $\hat{x} \in \mathbb{X}$, $\hat{u} \in \mathbb{U}$, and $\alpha \in \mathbb{R}$. Let $[\hat{x}_0^* \cdots \hat{x}_N^*]$, $[\hat{u}_0^* \cdots \hat{u}_{N-1}^*]$ be the solution to the

problem,

$$\min_{\hat{x}_0 \cdots \hat{x}_N, \hat{u}_0 \cdots \hat{u}_{N-1}} \quad V_\theta(\hat{x}_0 \cdots \hat{x}_N, \hat{u}_0 \cdots \hat{u}_{N-1}) \tag{4.7a}$$

$$\textit{such that} \quad \hat{x}_0 = x \tag{4.7b}$$

$$\hat{x}_{n+1} = \hat{A}\hat{x}_n + \hat{B}\hat{u}_n \;\forall\; n \in \{0 \cdots N-1\}. \tag{4.7c}$$

Then, $[\alpha\hat{x}_0^* \cdots \alpha\hat{x}_N^*]$, $[\alpha\hat{u}_0^* \cdots \alpha\hat{u}_{N-1}^*]$ *is the solution to the following problem,*

$$\min_{\hat{x}_0 \cdots \hat{x}_N, \hat{u}_0 \cdots \hat{u}_{N-1}} \quad V_\theta(\hat{x}_0 \cdots \hat{x}_N, \hat{u}_0 \cdots \hat{u}_{N-1}) \tag{4.8a}$$

$$\textit{such that} \quad \hat{x}_0 = \alpha x \tag{4.8b}$$

$$\hat{x}_{n+1} = \hat{A}\hat{x}_n + \hat{B}\hat{u}_n \;\forall\; n \in \{0 \cdots N-1\}. \tag{4.8c}$$

Proof. As $[\hat{x}_0^* \cdots \hat{x}_N^*]$, $[\hat{u}_0^* \cdots \hat{u}_{N-1}^*]$ is the optimal solution to (4.7), it must satisfy its constraints. Therefore, for all $n \in \{0 \cdots N-1\}$,

$$\hat{x}_0^* = x \tag{4.9a}$$

$$\hat{x}_{n+1}^* = \hat{A}\hat{x}_n^* + \hat{B}\hat{u}_n^* \tag{4.9b}$$

and therefore,

$$\alpha\hat{x}_0^* = \alpha x \tag{4.10a}$$

$$\alpha\hat{x}_{n+1}^* = \hat{A}(\alpha\hat{x}_n^*) + \hat{B}(\alpha\hat{u}_n^*) \tag{4.10b}$$

which means that the solution $[\alpha\hat{x}_0^* \cdots \alpha\hat{x}_N^*]$, $[\alpha\hat{u}_0^* \cdots \alpha\hat{u}_{N-1}^*]$ satisfies the constraints of problem (4.8). To show that it is indeed the optimal solution, let us add arbitrary increments $[\alpha\delta\hat{x}_0 \cdots \alpha\delta\hat{x}_N]$, $[\alpha\delta\hat{u}_0 \cdots \alpha\delta\hat{u}_{N-1}]$ such that the perturbed solution $[\alpha(\hat{x}_0^* + \delta\hat{x}_0) \cdots \alpha(\hat{x}_N^* + \delta\hat{x}_N)]$, $[\alpha(\hat{u}_0^* + \delta\hat{u}_0) \cdots \alpha(\hat{u}_{N-1}^* + \delta\hat{u}_{N-1})]$ also satisfies the constraints of (4.8),

$$\alpha(\hat{x}_0^* + \delta\hat{x}_0) = \alpha x \tag{4.11a}$$

$$\alpha(\hat{x}_{n+1}^* + \delta\hat{x}_{n+1}) = \hat{A}(\alpha(\hat{x}_n^* + \delta\hat{x}_n)) + \hat{B}(\alpha(\hat{u}_n^* + \delta\hat{u}_n)) \tag{4.11b}$$

from which it follows that

$$(\hat{x}_0^* + \delta\hat{x}_0) = x \tag{4.12a}$$

$$(\hat{x}_{n+1}^* + \delta\hat{x}_{n+1}) = \hat{A}(\hat{x}_n^* + \delta\hat{x}_n) + \hat{B}(\hat{u}_n^* + \delta\hat{u}_n). \tag{4.12b}$$

Thus, the solution $[\hat{x}_0^* + \delta\hat{x}_0 \cdots \hat{x}_N^* + \delta\hat{x}_N]$, $[\hat{u}_0^* + \delta\hat{u}_0 \cdots \hat{u}_{N-1}^* + \delta\hat{u}_{N-1}]$ satisfies the constraints of problem (4.7). As $[\hat{x}_0^* \cdots \hat{x}_N^*]$, $[\hat{u}_0^* \cdots \hat{u}_{N-1}^*]$ is the optimal solution to (4.7), we conclude that,

$$\begin{aligned} V_\theta(\hat{x}_0^* + \delta\hat{x}_0 \cdots \hat{x}_N^* + \delta\hat{x}_N, \hat{u}_0^* + \delta\hat{u}_0 \cdots \hat{u}_{N-1}^* + \delta\hat{u}_{N-1}) \\ \geq V_\theta(\hat{x}_0^* \cdots \hat{x}_N^*, \hat{u}_0^* \cdots \hat{u}_{N-1}^*) \end{aligned} \tag{4.13a}$$

$$\begin{aligned} \therefore \kappa(\alpha) V_\theta(\hat{x}_0^* + \delta\hat{x}_0 \cdots \hat{x}_N^* + \delta\hat{x}_N, \hat{u}_0^* + \delta\hat{u}_0 \cdots \hat{u}_{N-1}^* + \delta\hat{u}_{N-1}) \\ \geq \kappa(\alpha) V_\theta(\hat{x}_0^* \cdots \hat{x}_N^*, \hat{u}_0^* \cdots \hat{u}_{N-1}^*) \end{aligned} \tag{4.13b}$$

$$\begin{aligned} \therefore V_\theta(\alpha(\hat{x}_0^* + \delta\hat{x}_0) \cdots \alpha(\hat{x}_N^* + \delta\hat{x}_N), \alpha(\hat{u}_0^* + \delta\hat{u}_0) \cdots \alpha(\hat{u}_{N-1}^* + \delta\hat{u}_{N-1})) \\ \geq V_\theta(\alpha\hat{x}_0^* \cdots \alpha\hat{x}_N^*, \alpha\hat{u}_0^* \cdots \alpha\hat{u}_{N-1}^*). \end{aligned} \tag{4.13c}$$

Thus $[\alpha\hat{x}_0^* \cdots \alpha\hat{x}_N^*]$, $[\alpha\hat{u}_0^* \cdots \alpha\hat{u}_{N-1}^*]$ is the optimal solution to problem (4.8). ■

Theorem 4. *If $\pi_\theta : \mathbb{X} \to \mathbb{U}$ is the following model predictive control law,*

$$\pi_\theta(x) = \hat{u}_0^* \tag{4.14a}$$

$$\textit{where} \quad [\hat{u}_0^* \cdots \hat{u}_{N-1}^*] = \arg\min_{\hat{u}_0 \cdots \hat{u}_{N-1}} \left[\sum_{n=0}^{N-1} L_\theta(\hat{x}_n, \hat{u}_n) + F_\theta(\hat{x}_N) \right] \tag{4.14b}$$

$$\textit{such that} \quad \hat{x}_0 = x \tag{4.14c}$$

$$\hat{x}_{n+1} = \hat{A}\hat{x}_n + \hat{B}\hat{u}_n \;\forall\; n \in \{0 \cdots N-1\} \tag{4.14d}$$

where L_θ and F_θ satisfy (4.6), then $\pi_\theta(\alpha x) = \alpha\pi_\theta(x)$ for all $x \in \mathbb{X}$ and $\alpha \in \mathbb{R}$.

Proof. Let $[\hat{x}_0^* \cdots \hat{x}_N^*]$, $[\hat{u}_0^* \cdots \hat{u}_{N-1}^*]$ be the solution to the following optimisation

problem,

$$\min_{\hat{x}_0 \cdots \hat{x}_N, \hat{u}_0 \cdots \hat{u}_{N-1}} \left[\sum_{n=0}^{N-1} L_\theta(\hat{x}_n, \hat{u}_n) + F_\theta(\hat{x}_N) \right] \tag{4.15a}$$

$$\text{such that} \quad \hat{x}_0 = x \tag{4.15b}$$

$$\hat{x}_{n+1} = \hat{A}\hat{x}_n + \hat{B}\hat{u}_n \;\forall\; n \in \{0 \cdots N-1\}. \tag{4.15c}$$

By Theorem 3, $[\alpha\hat{x}_0^* \cdots \alpha\hat{x}_N^*]$, $[\alpha\hat{u}_0^* \cdots \alpha\hat{u}_{N-1}^*]$ is the solution to the following problem,

$$\min_{\hat{x}_0 \cdots \hat{x}_N, \hat{u}_0 \cdots \hat{u}_{N-1}} \left[\sum_{n=0}^{N-1} L_\theta(\hat{x}_n, \hat{u}_n) + F_\theta(\hat{x}_N) \right] \tag{4.16a}$$

$$\text{such that} \quad \hat{x}_0 = \alpha x \tag{4.16b}$$

$$\hat{x}_{n+1} = \hat{A}\hat{x}_n + \hat{B}\hat{u}_n \;\forall\; n \in \{0 \cdots N-1\} \tag{4.16c}$$

and therefore, by definition, $\pi_\theta(x) = \hat{u}_0^*$ and $\pi_\theta(\alpha x) = \alpha\hat{u}_0^*$. Thus, $\pi_\theta(\alpha x) = \alpha\pi_\theta(x)$. ■

Theorem 5. *If a plant is governed by the following linear dynamic model,*

$$x_{t+1} = Ax_t + Bu_t \tag{4.17}$$

where $x_t \in \mathbb{X}$ and $u_t \in \mathbb{U}$, and if $u_t = \pi_\theta(x_t)$ is computed from the following MPC scheme,

$$\pi_\theta(x) = \hat{u}_0^* \tag{4.18a}$$

$$\textit{where} \quad [\hat{u}_0^* \cdots \hat{u}_{N-1}^*] = \arg\min_{\hat{u}_0 \cdots \hat{u}_{N-1}} \left[\sum_{n=0}^{N-1} L_\theta(\hat{x}_n, \hat{u}_n) + F_\theta(\hat{x}_N) \right] \tag{4.18b}$$

$$\textit{such that} \quad \hat{x}_0 = x \tag{4.18c}$$

$$\hat{x}_{n+1} = \hat{A}\hat{x}_n + \hat{B}\hat{u}_n \;\forall\; n \in \{0 \cdots N-1\} \tag{4.18d}$$

where L_θ and F_θ satisfy the homogeneity criterion (4.6)*, then if $[x_0 \cdots x_{T-1}]$ and $[u_0 \cdots u_{T-1}]$ are the closed loop state and control trajectories for an episode with initial state x_{in}, then $[\alpha x_0 \cdots \alpha x_{T-1}]$ and $[\alpha u_0 \cdots \alpha u_{T-1}]$ are the corresponding tra-*

jectories for an initial state αx_{in} for all $x_{\text{in}} \in \mathbb{X}$ and $\alpha \in \mathbb{R}$.

Proof. From the closed loop trajectories with initial state x_{in}, we conclude that,

$$x_0 = x_{\text{in}} \tag{4.19a}$$
$$x_{t+1} = Ax_t + Bu_t \ \forall\ t \in \{0 \cdots T-2\} \tag{4.19b}$$
$$u_t = \pi_\theta(x_t) \ \forall\ t \in \{0 \cdots T-1\}. \tag{4.19c}$$

Multiplying by α and using Theorem 4, we get,

$$\alpha x_0 = \alpha x_{\text{in}} \tag{4.20a}$$
$$\alpha x_{t+1} = A(\alpha x_t) + B(\alpha u_t) \ \forall\ t \in \{0 \cdots T-2\} \tag{4.20b}$$
$$\alpha u_t = \pi_\theta(\alpha x_t) \ \forall\ t \in \{0 \cdots T-1\}. \tag{4.20c}$$

Thus, $[\alpha x_0 \cdots \alpha x_{T-1}]$ and $[\alpha u_0 \cdots \alpha u_{T-1}]$ are the closed loop state and control trajectories for an episode with initial state αx_{in}. ■

We can now demonstrate that the quadratic performance measure (3.7) has a significant dependence on x_{in} when the controller (3.6) is used with the plant (3.3). For simplicity, we consider the case when $w_t = 0$ so that Theorems 3, 4, and 5 are applicable. Let $[x_0 \cdots x_{T-1}]$ and $[u_0 \cdots u_{T-1}]$ be the state and control trajectories for an episode with initial state x_{in} and tuning parameters θ. By Theorem 5, the corresponding trajectories for an episode with initial state αx_{in} will be $[\alpha x_0 \cdots \alpha x_{T-1}]$ and $[\alpha u_0 \cdots \alpha u_{T-1}]$. The performance values for these episodes are then related as follows,

$$J(\alpha x_{\text{in}}, \theta) = -\frac{1}{T}\sum_{t=0}^{T-1}\left(\|\alpha x_t\|_Q^2 + \|\alpha u_t\|_R^2\right) \tag{4.21a}$$
$$\therefore J(\alpha x_{\text{in}}, \theta) = -\frac{\alpha^2}{T}\sum_{t=0}^{T-1}\left(\|x_t\|_Q^2 + \|u_t\|_R^2\right) \tag{4.21b}$$
$$\therefore J(\alpha x_{\text{in}}, \theta) = \alpha^2 J(x_{\text{in}}, \theta). \tag{4.21c}$$

Thus, we see that the quadratic performance function exhibits significant variation with respect to x_{in}. Simply scaling up x_{in} can cause a large change in $J(x_{\text{in}}, \theta)$. As

an alternative, the following performance measure can be considered instead,

$$J(x_{\text{in}},\theta)=\mathbb{E}_{x_0=x_{\text{in}},u_t=\pi_\theta(x_t)}\left[-\frac{1}{T-K}\sum_{t=0}^{T-K-1}\sum_{i=1}^{n_x}\lambda_i\frac{|x_{t+K,i}|-|x_{t,i}|}{K|x_{t,i}|}\right] \tag{4.22}$$

which can be estimated from the observed closed loop trajectory for an episode as follows,

$$\hat{J}(x_0\cdots x_{T-1},u_0\cdots u_{T-1})=-\frac{1}{T-K}\sum_{t=0}^{T-K-1}\sum_{i=1}^{n_x}\lambda_i\frac{|x_{t+K,i}|-|x_{t,i}|}{K|x_{t,i}|} \tag{4.23}$$

where $x_{t,i}$ denotes the i-th component of the vector x_t and $\lambda_i\in\mathbb{R}^+$ are weighting factors. The form of this performance measure is inspired from the expression $-\dot{x}_{t,i}/x_{t,i}$ which, in a continuous time setting, characterises the rate of convergence of $x_{t,i}$ to zero. In discrete time, the expression $-(x_{t+K,i}-x_{t,i})/Kx_{t,i}$ serves as an approximation. However, this approximation may also have large positive values near oscillations about zero. This is avoided in (4.22) by taking absolute values of the state vector components.

While this performance measure is not altogether independent of x_{in}, it is independent of its norm $\|x_{\text{in}}\|$ for a linear plant without process noise. More precisely, if the plant dynamics are described by (4.17) and the controller (4.18) is used, then $J(\alpha x_{\text{in}},\theta)=J(x_{\text{in}},\theta)$ as shown below. If $[x_0\cdots x_{T-1}]$ and $[u_0\cdots u_{T-1}]$ are the closed loop state and control trajectories for an initial state x_{in}, then by Theorem 5, the corresponding trajectories for an initial state αx_{in} are $[\alpha x_0\cdots\alpha x_{T-1}]$ and $[\alpha u_0\cdots\alpha u_{T-1}]$. Therefore, the performance values for the two cases are related as follows,

$$J(\alpha x_{\text{in}},\theta)=-\frac{1}{T-K}\sum_{t=0}^{T-K-1}\sum_{i=1}^{n_x}\lambda_i\frac{|\alpha x_{t+K,i}|-|\alpha x_{t,i}|}{K|\alpha x_{t,i}|} \tag{4.24a}$$

$$\therefore J(\alpha x_{\text{in}},\theta)=-\frac{1}{T-K}\sum_{t=0}^{T-K-1}\sum_{i=1}^{n_x}\lambda_i\frac{|\alpha|(|x_{t+K,i}|-|x_{t,i}|)}{K|\alpha||x_{t,i}|} \tag{4.24b}$$

$$\therefore J(\alpha x_{\text{in}},\theta)=-\frac{1}{T-K}\sum_{t=0}^{T-K-1}\sum_{i=1}^{n_x}\lambda_i\frac{|x_{t+K,i}|-|x_{t,i}|}{K|x_{t,i}|} \tag{4.24c}$$

$$\therefore J(\alpha x_{\text{in}},\theta)=J(x_{\text{in}},\theta). \tag{4.24d}$$

Despite not being entirely independent of x_{in}, the independence of J from $\|x_{\text{in}}\|$ is a significant improvement over the quadratic performance function (3.7). However, this result is also subject to a few assumptions, namely that noise is absent from the system, the plant is linear, state and control constraints are absent, and the cost functions satisfy the homogeneity condition (4.6). In practice, it is generally only possible to satisfy the last assumption, through careful choice of cost functions. It is therefore important to consider the effects of violating the other assumptions. We first consider the case when process noise and/or measurement noise is present. For a stable system, such noises are expected to result in random variations in the closed loop trajectory for any given x_{in} and θ, which can affect $J(x_{\text{in}}, \theta)$. However, if these noise signals are sufficiently small, no major changes are expected in the long term convergence properties. The presence of noise can still affect the individual terms summed in (4.23), but with an appropriate choice of K, it should be possible to diminish this effect. Next, we consider the case of nonlinear plants with state and control constraints. Such plants can still be approximated as being linear while operating in close proximity to the equilibrium. If the constraints are not active in this region (as is often the case with bound constraints, for instance) then the plant behaves like a linear unconstrained system. As we move away from the equilibrium, we intuitively expect the effect of nonlinearities and constraints to gradually increase, causing some variation in $J(\alpha x_{\text{in}}, \theta)$ with respect to α. However, depending on how greatly this variation increases away from the equilibrium, the performance measure might still be usable, in contrast to the quadratic performance function, which always has a dependence on α.

5 Simulation experiments

Simulation experiments were carried out to verify the method developed in Chapter 4. In particular, a comparison was made between the quadratic performance function (3.7) and the convergence rate performance function (4.22) to show that the latter is better suited for applications where the initial state x_{in} is not fixed. Experiments were carried out with a simulated cart pole system to test the cost function learning method for MPC with, as well as without terminal conditions. In Section 5.1, we describe the overall scenario in which simulations were performed. In particular, we describe the cart pole simulator, the formulation of the control system, and the performance measure used. Following this, we present the results of individual experiments in Section 5.2.

5.1 Simulation setup

In this section, we describe in detail, the various components that make up the simulation experiments. We introduce the overall simulation environment by describing the parts that are common to all simulation experiments.

5.1.1 Plant dynamics

Consider the cart pole system depicted in Figure 5.1. A cart of mass m_c can slide smoothly on a rail. It is connected by a massless rod of length l to a load of mass m_l. A horizontal force f can be applied to the cart in order to control the motion of the system.

Let ϕ denote the angle of inclination of the pendulum with the vertical, and let $\omega = \dot{\phi}$ denote its angular velocity. Then $x = [\phi, \omega]^\intercal$ is the state vector for this system, and $u = [f]^\intercal$ is the control vector. The objective of control design is to stabilise this system at $x_e = [0, 0]^\intercal$ which is an unstable equilibrium for the uncontrolled system.

Using the Lagrangian formulation of mechanics [5, Section 1.4], it is easy to derive the equations of motion for the cart pole system,

$$\dot{\phi} = \omega \tag{5.1a}$$

$$\dot{\omega} = \frac{(m_c + m_l)g \sin\phi - m_l l \omega^2 \sin\phi \cos\phi - f \cos\phi}{l(m_c + m_l \sin^2\phi)}. \tag{5.1b}$$

The above system is simulated in discrete time using Euler's method of numerical integration with a time step Δt. Table 5.1 shows the numerical values of the parameters used in the simulation.

5.1.2 Control system

The cart pole system is controlled by an MPC scheme which uses a linearised prediction model of the plant dynamics. It is easy to derive the discrete time linearised equation of motion of the system (5.1) with a time step Δt,

$$\begin{bmatrix} \hat{\phi}_{t+1} \\ \hat{\omega}_{t+1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ g l^{-1} \Delta t (1 + m_l/m_c) & 1 \end{bmatrix} \begin{bmatrix} \hat{\phi}_t \\ \hat{\omega}_t \end{bmatrix} + \begin{bmatrix} 0 \\ -\Delta t/(m_c l) \end{bmatrix} \begin{bmatrix} \hat{f}_t \end{bmatrix}. \tag{5.2}$$

In other words, the prediction model is as follows,

$$\hat{x}_{t+1} = \hat{A}\hat{x}_t + \hat{B}\hat{u}_t \tag{5.3a}$$

$$\text{where} \quad \hat{A} = \begin{bmatrix} 1 & \Delta t \\ g l^{-1} \Delta t (1 + m_l/m_c) & 1 \end{bmatrix} \quad \hat{B} = \begin{bmatrix} 0 \\ -\Delta t/(m_c l) \end{bmatrix} \tag{5.3b}$$

State and control constraints are also applied. Hard constraints are imposed on the control vector but soft constraints are imposed on the state with the help of slack variables. The following are the constraint equations for the predicted state and control vectors,

$$\hat{x}_{\min} - \sigma_n \leq \hat{x}_n \leq \hat{x}_{\max} + \sigma_n \;\forall\; n \in \{0 \cdots N\} \tag{5.4a}$$

$$\hat{u}_{\min} \leq \hat{u}_n \leq \hat{u}_{\max} \;\forall\; n \in \{0 \cdots N-1\} \tag{5.4b}$$

$$\sigma_n \geq 0 \;\forall\; n \in \{0 \cdots N\}. \tag{5.4c}$$

Parameter	Value
m_c	0.05
m_l	0.05
l	0.5
g	10
Δt	10^{-3}

Table 5.1: Parameters used to simulate the motion of the cart pole system (5.1). Euler's method of numerical integration was used to run the simulation in discrete time with a time step of Δt. All values indicated are in SI units.

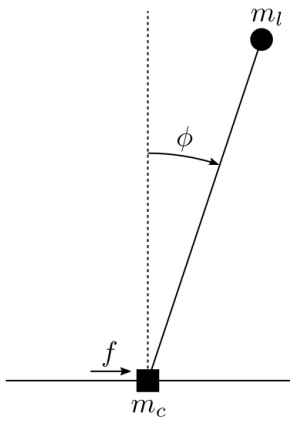

Figure 5.1: Schematic diagram of a simple one-dimensional cart pole system. A cart of mass m_c slides without friction on an infinite, linear rail. A massless pendulum of length l pivoted at the cart connects it to a load of mass m_l. The inclination of the pendulum can be controlled applying a horizontal force f and moving the cart. The equilibria of the uncontrolled system lie at $\phi = \pm\pi$ and $\phi = 0$, of which the latter is unstable. The objective of control design is to stabilise the pendulum in its vertical position $\phi = 0$.

The common quadratic cost function is chosen for the OCP but with an additional term to penalise non-zero slack variables. Thus, the total cost function is given by the following equation,

$$V(\hat{x}_0 \cdots \hat{x}_N, \hat{u}_0 \cdots \hat{u}_{N-1}) = \sum_{n=0}^{N-1} \left(\|\hat{x}_n\|_{\hat{Q}}^2 + \|\hat{u}_n\|_{\hat{R}}^2 \right) + \|\hat{x}_N\|_{\hat{S}}^2 + \sum_{n=0}^{N} w_\sigma^\intercal \sigma_n \tag{5.5}$$

Altogether, the complete MPC scheme can be expressed as follows,

$$u = \pi(x) = \hat{u}_0^* \tag{5.6a}$$

$$\text{where} \quad [\hat{u}_0^* \cdots \hat{u}_{N-1}^*] = \arg \min_{\hat{u}_0 \cdots \hat{u}_{N-1}} \left[\sum_{n=0}^{N-1} \left(\|\hat{x}_n\|_{\hat{Q}}^2 + \|\hat{u}_n\|_{\hat{R}}^2 \right) + \|\hat{x}_N\|_{\hat{S}}^2 + \sum_{n=0}^{N} w_\sigma^\intercal \sigma_n \right] \tag{5.6b}$$

$$\text{such that} \quad \hat{x}_0 = x \tag{5.6c}$$

$$\hat{x}_{n+1} = \hat{A}\hat{x}_n + \hat{B}\hat{u}_n \; \forall \; n \in \{0 \cdots N-1\} \tag{5.6d}$$

$$\hat{x}_{\min} - \sigma_n \leq \hat{x}_n \leq \hat{x}_{\max} + \sigma_n \; \forall \; n \in \{0 \cdots N\} \tag{5.6e}$$

$$\hat{u}_{\min} \leq \hat{u}_n \leq \hat{u}_{\max} \; \forall \; n \in \{0 \cdots N-1\} \tag{5.6f}$$

$$\sigma_n \geq 0 \; \forall \; n \in \{0 \cdots N\}. \tag{5.6g}$$

For experiments using MPC without terminal conditions, the control scheme is as follows,

$$u = \pi(x) = \hat{u}_0^* \tag{5.7a}$$

$$\text{where} \quad [\hat{u}_0^* \cdots \hat{u}_{N-1}^*] = \arg \min_{\hat{u}_0 \cdots \hat{u}_{N-1}} \left[\sum_{n=0}^{N-1} \left(\|\hat{x}_n\|_{\hat{Q}}^2 + \|\hat{u}_n\|_{\hat{R}}^2 \right) + \sum_{n=0}^{N-1} w_\sigma^\intercal \sigma_n \right] \tag{5.7b}$$

$$\text{such that} \quad \hat{x}_0 = x \tag{5.7c}$$

$$\hat{x}_{n+1} = \hat{A}\hat{x}_n + \hat{B}\hat{u}_n \; \forall \; n \in \{0 \cdots N-1\} \tag{5.7d}$$

$$\hat{x}_{\min} - \sigma_n \leq \hat{x}_n \leq \hat{x}_{\max} + \sigma_n \; \forall \; n \in \{0 \cdots N-1\} \tag{5.7e}$$

$$\hat{u}_{\min} \leq \hat{u}_n \leq \hat{u}_{\max} \; \forall \; n \in \{0 \cdots N-1\} \tag{5.7f}$$

$$\sigma_n \geq 0 \; \forall \; n \in \{0 \cdots N-1\}. \tag{5.7g}$$

The parameters used to set up the MPC are given in Table 5.2. The weight matrices $\hat{Q}$, $\hat{R}$, and $\hat{S}$ are not specified as they are tuning parameters and have different (and

possibly varying) values in different experiments. Likewise, different prediction horizon lengths N are used in experiments with, and without terminal conditions. The values of these parameters are therefore mentioned in Section 5.2 in the description of the individual experiments and not in Table 5.2.

5.1.3 Performance measure

Both, the quadratic performance (3.7) as well as the convergence rate performance (4.22) are used depending on the details of the experiment. The following parameters are used in the quadratic performance function (3.7),

$$Q = \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix} \quad R = \begin{bmatrix} 1 \end{bmatrix} \tag{5.8}$$

and those used in the convergence rate performance (4.22) are as follows,

$$\lambda = \begin{bmatrix} 5 \\ 1 \end{bmatrix} \quad K = 20. \tag{5.9}$$

While estimating the performance using equation (4.23), only those terms are considered in the summation for which $|x_{t,i}|$ is greater than 10^{-3} to avoid numerical problems. For each i-th component, the factor $1/(T-K)$ in (4.23) is also accordingly adjusted to account for only as any terms as have been summed.

5.2 Experimental results

In this section, we present the results of individual simulation experiments. The experiments were performed to establish that (4.22) is a more suitable performance measure than (3.7), as well as to test the stage cost learning algorithm.

5.2.1 Comparison of performance functions

In this experiment, the simulated cart pole system was controlled by an MPC scheme with terminal conditions, to observe the variation in performance values with the initial state x_{in} as well as tuning parameters θ. A prediction horizon of $N = 10$ was chosen for the MPC scheme. The control weight matrix $\hat{R}$ was chosen as the tuning

parameter θ and 6 uniformly spaced values of $\hat{R}$ were chosen spanning the range $[1, 100]$. The state weight matrix $\hat{Q}$ was fixed at $[10, 0; 0, 10]$ and for each value of $\hat{R}$, $\hat{S}$ was computed from the discrete algebraic Riccati equation (DARE). For each value of $\hat{R}$, 36 different initial states x_{in} were chosen in a uniformly spaced 6×6 grid spanning the rectangle $[-\bar{\phi}, \bar{\phi}] \times [-\bar{\omega}, \bar{\omega}]$, thus yielding 216 different combinations of $\hat{R}$ and x_{in}. The experiment was run twice, once with $(\bar{\phi}, \bar{\omega}) = (\pi/6, \pi/6)$ and once more with $(\bar{\phi}, \bar{\omega}) = (\pi/12, \pi/12)$ to observe the effect on the performance values when x_{in} was scaled. Also, observe that in the former case, the soft state constraints are active at the initial time step while this is not so in the latter case. In this way, we can observe the effect on the performance function when x_{in} is far enough from the equilibrium for constraints to have an influence. For each combination of $\hat{R}$ and x_{in}, an episode of $T = 7500$ simulation time steps (corresponding to a time interval of 7.5 seconds) was carried out, after which both performance functions were evaluated.

Figures 5.2 and 5.3 show the observed convergence rate performance $\hat{J}$ plotted against $\hat{R}$ for $(\bar{\phi}, \bar{\omega}) = (\pi/6, \pi/6)$ and $(\bar{\phi}, \bar{\omega}) = (\pi/12, \pi/12)$ respectively. For each of the 6 values of $\hat{R}$, 36 different $\hat{J}$ values were observed corresponding to the 36 states x_{in}, and have been plotted in a scatter diagram. The performance function is not altogether independent of x_{in} as is evident from the varied values of $\hat{J}$ for each individual $\hat{R}$. However, it is practically independent of $\|x_{\text{in}}\|$ as both Figures 5.2 and 5.3 are almost identical, indicating that scaling the initial states x_{in} has had little effect on the performance. We also observe that despite the variation in performance values for the same value of $\hat{R}$, there is an overarching trend in the performance with respect to $\hat{R}$, which means that $\hat{R}$ can in principle, be updated to improve performance. In contrast, Figures 5.4 and 5.5 are the same plots when the quadratic performance measure is used. Neither do we observe any trend in $\hat{J}$ values that could be used to define an optimal value of $\hat{R}$, nor are the two plots similar to each other. The quadratic performance is therefore, a much less well-defined function of $\hat{R}$ in comparison to the convergence rate performance.

5.2.2 Stage cost learning with terminal conditions

In this experiment, we learn the tuning parameter $\theta = \hat{R}$ for the MPC scheme with terminal conditions. The cart pole system is controlled by the MPC scheme with

terminal conditions, described in Section 5.1.2. A prediction horizon of $N = 10$ was selected. The state weight matrix $\hat{Q}$ was fixed at $[10, 0; 0, 10]$ and BO was used to update $\hat{R}$ after every episode. The terminal weight $\hat{S}$ was computed from the DARE every time $\hat{R}$ was updated. The convergence rate performance function was used to assess the closed loop trajectories, and was modelled as a GP with the following SE kernel,

$$k(\theta_i, \theta_j) = 4 \times 10^{-5} \exp\Big[-0.0125(\theta_i - \theta_j)^2\Big]. \tag{5.10}$$

It was assumed that the performance estimates $\hat{J}_m$ computed from the closed loop state and control trajectories were given by,

$$\hat{J}_m = J(\theta_m) + v_m \quad \text{where} \quad v_m \sim \mathcal{N}\Big(0, 10^{-5}\Big) \tag{5.11}$$

After m episodes, the value of $\hat{R}$ for the $(m + 1)$-th episode was computed by maximising the EI acquisition function in the domain $\hat{R} \in [1, 100]$. For each episode, the initial state x_{in} was randomly chosen from a uniform distribution over the set $[-\pi/6, \pi/6] \times [-\pi/6, \pi/6]$. Each episode lasted for $T = 7500$ time steps. A total of 50 episodes were performed in the learning process.

Figure 5.6 shows the confidence bounds $\mu_{50}(\theta) \pm 3\sigma_{50}(\theta)$ on $J(\theta)$ where $\mu_{50}(\theta)$ and $\sigma_{50}(\theta)$ are the posterior mean and standard deviation functions after 50 episodes. Figure 5.7 shows the selected values of $\hat{R}$ and the observed performance $\hat{J}$ for each episode. We observe a gradually increasing trend in the performance values and we can also see that the system favours values of $\hat{R}$ close to 100. This is consistent with the results shown in Figure 5.2 which indicates that the performance is maximised near $\hat{R} = 100$.

5.2.3 Stage cost learning without terminal conditions

This experiment was similar to that in Section 5.2.2, but the major difference is that the MPC scheme used no terminal conditions. The terminal cost was therefore zero. In order to be able to stabilise the system without terminal conditions, the horizon length was increased to $N = 30$. The kernel parameters were also slightly modified,

$$k(\theta_i, \theta_j) = 10^{-5} \exp\Big[-0.0125(\theta_i - \theta_j)^2\Big]. \tag{5.12}$$

and so was the performance measurement noise,

$$\hat{J}_m = J(\theta_m) + v_m \quad \text{where} \quad v_m \sim \mathcal{N}\Big(0, 2.5 \times 10^{-6}\Big). \tag{5.13}$$

Figure 5.8 shows the confidence bounds for the learned posterior model of the performance function after 50 episodes and Figure 5.9 shows the values of $\hat{R}$ and $\hat{J}$ for each episode. Unlike the experiment with terminal conditions, we observe in this case that after increasing for a few episodes, $\hat{R}$ begins to oscillate. We also observe two large peaks in the observed performance, which could possibly have significantly affected the GP posterior model. One possible cause is that the hyperparameters of the GP model (the performance measurement noise and the parameters defining the kernel) are selected partly based on trial and error, and are not the best suited values for the experiment. It might be worth investigating a more sophisticated GP design involving the selection of hyperparameters based on observed data [4].

Parameter	Value
m_c	0.05
m_l	0.05
l	0.5
g	10
Δt	10^{-2}
w_σ	$[30, 30]$
$\hat{x}_{\min}$	$[-\pi/8, -\pi/8]$
$\hat{x}_{\max}$	$[\pi/8, \pi/8]$
$\hat{u}_{\min}$	$[-1]$
$\hat{u}_{\max}$	$[1]$

Table 5.2: Parameters used in the MPC schemes (5.6) and (5.7). Only parameter values that are common to all experiments are shown. Values of other parameters are mentioned in Section 5.2 in the individual experiment descriptions. All values indicated are in SI units.

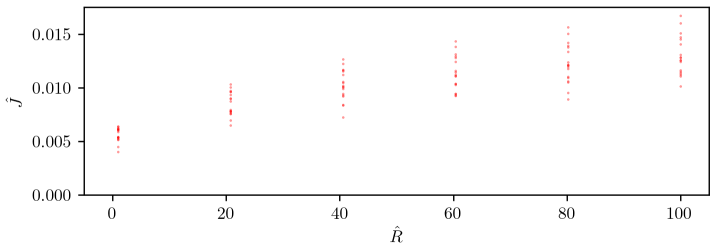

Figure 5.2: Convergence rate performance versus tuning parameter $\hat{R}$ with initial state set corresponding to $(\bar{\phi}, \bar{\omega}) = (\pi/6, \pi/6)$. While there is some variation in $\hat{J}$ with x_{in} for the same $\hat{R}$, a overall trend in $\hat{J}$ can still be observed. This allows us to attempt to maximise $\hat{J}$ by varying $\hat{R}$. Besides, the above plot is almost identical to Figure 5.3 indicating that scaling x_{in} has very little effect on $\hat{J}$.

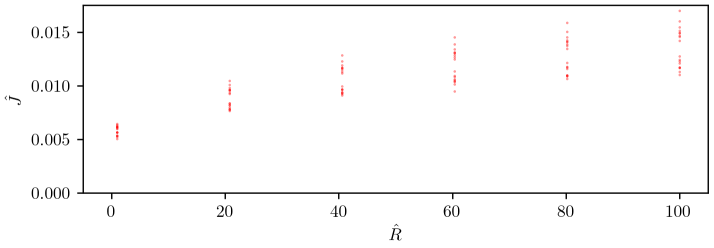

Figure 5.3: Convergence rate performance versus tuning parameter $\hat{R}$ with initial state set corresponding to $(\bar{\phi}, \bar{\omega}) = (\pi/12, \pi/12)$. While there is some variation in $\hat{J}$ with x_{in} for the same $\hat{R}$, a overall trend in $\hat{J}$ can still be observed. This allows us to attempt to maximise $\hat{J}$ by varying $\hat{R}$. Besides, the above plot is almost identical to Figure 5.2 indicating that scaling x_{in} has very little effect on $\hat{J}$.

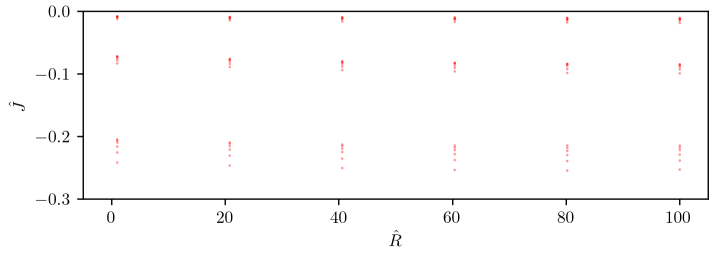

Figure 5.4: Quadratic performance versus tuning parameter $\hat{R}$ with initial state set corresponding to $(\bar{\phi}, \bar{\omega}) = (\pi/6, \pi/6)$. No visible trend exists in $\hat{J}$ with respect to $\hat{R}$. The above plot is also significantly different from Figure 5.5, indicating that scaling x_{in} has a pronounced impact on $\hat{J}$.

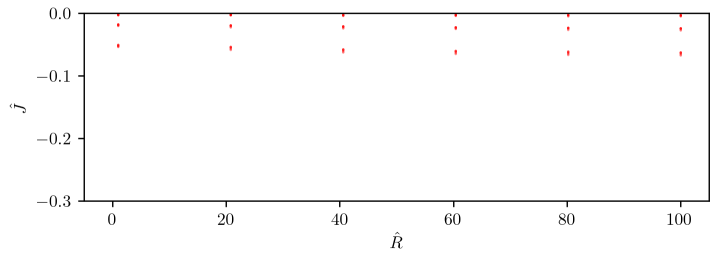

Figure 5.5: Quadratic performance versus tuning parameter $\hat{R}$ with initial state set corresponding to $(\bar{\phi}, \bar{\omega}) = (\pi/12, \pi/12)$. No visible trend exists in $\hat{J}$ with respect to $\hat{R}$. The above plot is also significantly different from Figure 5.4, indicating that scaling x_{in} has a pronounced impact on $\hat{J}$.

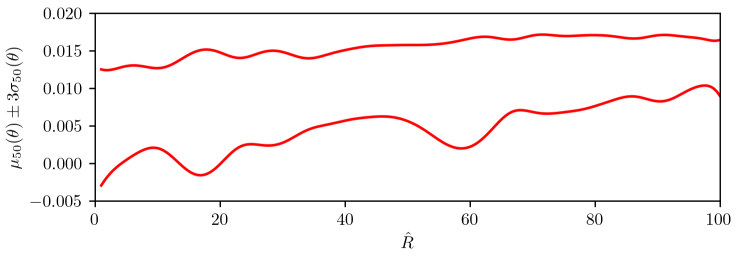

Figure 5.6: Confidence bounds for the GP posterior model of performance as a function of tuning parameter $\hat{R}$ after 50 episodes. The MPC scheme used a terminal cost function and a horizon of $N = 10$.

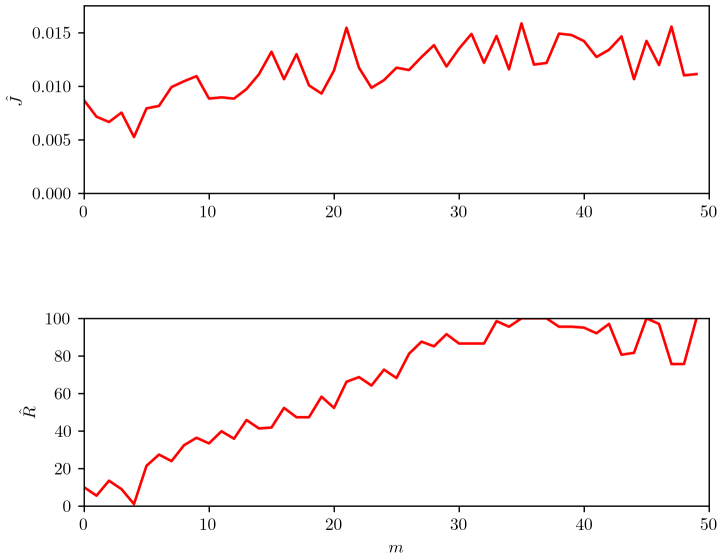

Figure 5.7: Selected values of the tuning parameter $\hat{R}$ and corresponding performance values observed while learning $\hat{R}$. The MPC scheme used a terminal cost function and a horizon of $N = 10$.

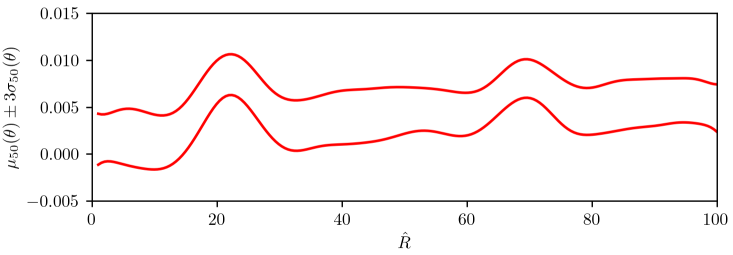

Figure 5.8: Confidence bounds for the GP posterior model of performance as a function of tuning parameter $\hat{R}$ after 50 episodes. The MPC scheme used no terminal conditions and used a horizon of $N = 30$.

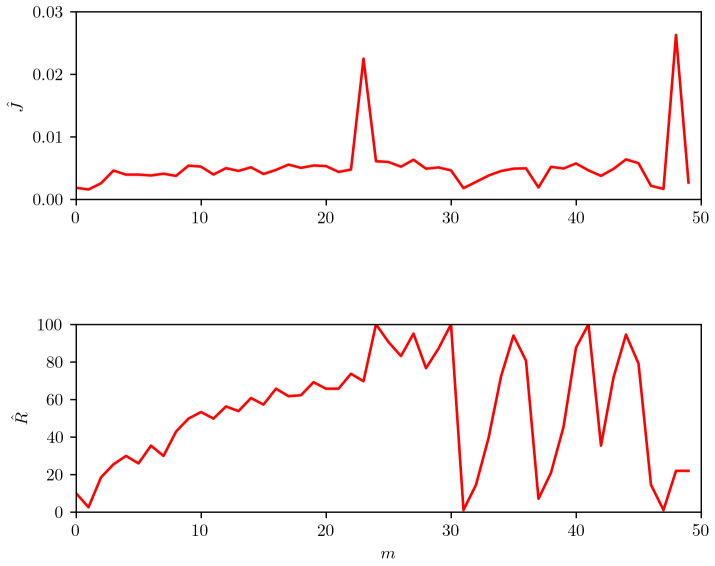

Figure 5.9: Selected values of the tuning parameter $\hat{R}$ and corresponding performance values observed while learning $\hat{R}$. The MPC scheme used no terminal conditions and used a horizon of $N = 30$.

6 Conclusion

6.1 Summary

In this thesis, we propose a method of learning the cost function for an MPC scheme based on the concept of improving a performance value. We then interpret the problem of learning the stage cost as one of controller tuning.

An important outcome of this thesis was the design of the performance measure to be less dependent on the initial state of an episode. We demonstrated that this property is crucial by comparison with another performance function which does not satisfy this property.

We observed in simulation experiments that it is possible for the performance of a closed loop system to improve using the proposed technique. However, certain drawbacks were also observed in the experiment without terminal conditions. In particular, that the observed performance data might not easily admit to modelling by an already chosen GP model.

6.2 Future work

The experiments in this thesis were focused on learning only one parameter in the stage cost. A clear direction for further work in this regard would be to update multiple parameters characterising the stage cost. Similarly, another extension would be to also learn a terminal cost function simultaneously with the stage cost.

An important consideration while designing a GP model is the choice of hyperparameters. Badly chosen hyperparameters can easily cause posterior models to entirely deviate from reality. One possible direction of future research is to address this issue, probably by choosing and updating the hyperparameters based on the observed data. This is not a new idea and has already been discussed in [4]. Possible methods include finding the *maximum likelihood estimate* (MLE) of the hyperparam-

eters, given the observed data. For more techniques on choosing hyperparameters, see [4].

Acronyms

BO	Bayesian optimisation
CPI	Controlled positively invariant
DARE	Discrete algebraic Riccati equation
EI	Expected improvement
ES	Entropy search
GP	Gaussian process
LQR	Linear quadratic regulator
MLE	Maximum likelihood estimate
MPC	Model predictive control
OCP	Optimal control problem
SE	Squared exponential

List of Mathematical Symbols

n_x	Dimension of state vector
n_u	Dimension of control vector
$\mathbb{X}$	State vector space
$\mathbb{U}$	Control vector space
X	State constraint set
U	Control constraint set
X_F	Terminal constraint set
x	State vector
u	Control vector
x_t	State vector at time t
u_t	Control vector at time t
w_t	Random noise vector at time t
Σ_w	Covariance of process noise w_t
$x_{t,i}$	Component i of state vector at time t
$u_{t,i}$	Component i of control vector at time t
$\hat{x}$	Predicted state vector
$\hat{u}$	Predicted control vector
$\hat{x}_n$	Predicted state vector after n time steps
$\hat{u}_n$	Predicted control vector after n time steps
x_e	Closed loop equilibrium state vector
x_{in}	Initial state vector
x_{in}^m	Initial state vector for episode m
n_θ	Dimension of tuning parameter vector
$\mathbb{P}$	Tuning parameter vector space
θ	Tuning parameter vector
θ_m	Tuning parameter vector for episode m

J	Performance function
$\hat{J}$	Performance estimation function
$\hat{J}_m$	Performance estimate for episode m
v_m	Random performance estimation error for episode m
Σ_v	Covariance of performance estimation error v_m
T	Length of episode
N	Length of prediction horizon
f	Plant dynamics
$\hat{f}$	Prediction model of plant dynamics
L	Stage cost function
F	Terminal cost function
V	Total cost function
L_θ	Stage cost function with tuning parameters θ
F_θ	Terminal cost function with tuning parameters θ
V_θ	Total cost function with tuning parameters θ
π	Control law
π_θ	Control law with tuning parameters θ
μ_m	Posterior mean function after m evaluations
σ_m	Posterior standard deviation function after m evaluations
k	Gaussian process kernel function
a	Acquisition function for Bayesian optimisation

List of Figures

List of Tables

Bibliography

[1] A. Aswani, H. Gonzalez, S. S. Sastry, and C. Tomlin, "Provably safe and robust learning-based model predictive control," *Automatica*, vol. 49, no. 5, pp. 1216–1226, 2013.

[2] S. Bouabdallah and R. Siegwart, "Backstepping and sliding-mode techniques applied to an indoor micro quadrotor," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005, pp. 2247–2252.

[3] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control," *IEEE Control Systems Magazine*, vol. 26, no. 3, pp. 96–114, 2006.

[4] P. I. Frazier, "A tutorial on bayesian optimisation," *arXiv preprint arXiv:1807.02811*, 2018.

[5] H. Goldstein, C. P. Poole, and J. L. Safko, *Classical mechanics*, 3rd ed. Addison Wesley, 2001.

[6] S. Gros and M. Zanon, "Data-driven economic nmpc using reinforcement learning," *IEEE Transactions on Automatic Control*, vol. 65, no. 2, pp. 636–648, 2020.

[7] L. Grüne and J. Pannek, *Nonlinear model predictive control*, 2nd ed. Springer International Publishing, 2017.

[8] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-based model predictive control: Toward safe learning in control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 269–296, 2020.

[9] B. Kouvaritakis and M. Cannon, *Model predictive control*, 1st ed. Springer International Publishing, 2016.

[10] A. Marco, P. Hennig, J. Bohg, S. Schaal, and S. Trimpe, "Automatic lqr tuning based on gaussian process global optimisation," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 270–277.

[11] D. Q. Mayne, "Model predictive control: Recent developments and future promise," *Automatica*, vol. 50, no. 12, pp. 2967–2986, 2014.

[12] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[13] M. A. Müller and K. Worthmann, "Quadratic costs do not always work in mpc," *Automatica*, vol. 82, pp. 269–277, 2017.

[14] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control Engineering Practice*, vol. 11, no. 7, pp. 733–764, 2003.

[15] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. MIT Press, 2006.

[16] U. Rosolia and F. Borrelli, "Learning model predictive control for iterative tasks: A data-driven control framework," *IEEE Transactions on Automatic Control*, vol. 63, no. 7, pp. 1883–1896, 2018.

[17] J. J. E. Slotine and W. Li, *Applied nonlinear control*. Prentice Hall, 1991.

[18] M. Tanaskovic, L. Fagiano, R. Smith, and M. Morari, "Adaptive receding horizon control for constrained mimo systems," *Automatica*, vol. 50, no. 12, pp. 3019–3029, 2014.