University
of Bremen

Online Parameter Identification for Optimal Feedback Control of Nonlinear Dynamical Systems

Dissertation

submitted to the Faculty of Mathematics and Computer Science of University of Bremen for the degree

— Dr. rer. nat. —

by

M.Sc. Margareta Runge

September 28, 2023

1st Reviewer:	Prof. Dr. Christof Büskens
2nd Reviewer:	Prof. Dr.-Ing. Kai Michels
Date of Defense:	January 10, 2024

Abstract

This research aims to enhance current methods for the optimal feedback control of complex nonlinear dynamical systems via online parameter identifications. Accurate knowledge of the system parameters is essential in numerous practical applications to ensure effective control. A considerable number of advanced control algorithms use model-based approaches. However, the model parameters may often be unknown or subject to change over time. This could result in deviations from the feedback control objective, increased expected costs, and even divergence of the controller.

The main objective of this thesis is to develop a combined online parameter identification and model-based controller approach that allows continuously estimating the model parameters of a nonlinear system. The available real-time measurements of the system are used to compute an approximation of the searched parameters. This repeated parameter estimation enables the control algorithm to adapt to the changing system dynamics and maintain optimal control accuracy. This study investigates three approaches. First, a coupled algorithm is developed that employs parameter identifications during operation to adapt a linear quadratic regulator using techniques from parametric sensitivity analysis. Additionally, an approach is presented that also examines the information quality in the data used to predict the probability of success of the parameter estimation. This process enables the suspension of parameter estimates in case of insufficient information. An adaptive control algorithm using nonlinear model predictive control (NMPC) and online parameter identification is proposed as a third alternative. The NMPC method is based on the repeated solution of optimal control problems, including system dynamics in their constraints. As such, the overall performance is significantly influenced by the model quality. The controller can continuously update the utilized model to account for changing system parameters, thus enabling full adaptivity. All proposed techniques rely on highly efficient numerical methods for solving nonlinear optimization problems (NLP) and the potential to transfer related problems from optimal control into an NLP by discretization.

The proposed approaches are extensively evaluated by conducting simulations and comparing them to the existing standard control methods. The findings indicate that the suggested online parameter identification significantly improves control performance compared to traditional static approaches. In particular, it compensates for model uncertainties, disturbances, and changes in system dynamics, leading to a robust control performance.

Zusammenfassung

Diese Forschungsarbeit befasst sich mit dem Ansatz, bestehende Verfahren zur optimalen Regelung von komplexen nichtlinearen dynamischen System durch online durchgeführte Parameteridentifikationen zu erweitern. In vielen praktischen Anwendungen sind genaue Kenntnisse über die Systemparameter von entscheidender Bedeutung, um eine effektive Regelung zu gewährleisten. Bei einer Vielzahl an hochentwickelten Regelungsalgorithmen handelt es sich um modellbasierte Ansätze. Dabei sind allerdings die Modellparameter häufig nicht vollständig bekannt oder können sich im Laufe der Zeit ändern. Dies kann zu Abweichungen vom Regelungsziel, einer Erhöhung der erwarteten Kosten und sogar zu Divergenz führen.

Das Hauptziel dieser Arbeit besteht darin, einen kombinierten Ansatz aus online Parameteridentifikation und modellbasiertem Regler zu entwickeln, der es ermöglicht, die Modellparameter eines nichtlinearen Systems kontinuierlich zu schätzen. Dabei werden die verfügbaren Echtzeitmessungen des Systems genutzt um eine Approximation der gesuchten Parameter zu berechnen. Diese kontinuierliche Parameterschätzung ermöglicht eine Adaption des Regelalgorithmus an die sich ändernde Systemdynamiken und gewährleistet eine optimale Regelungsgenauigkeit. In dieser Arbeit werden drei verschiedene Ansätze verfolgt. Zum einen wird ein gekoppelter Algorithmus entwickelt, der Parameteridentifikationen während des Betriebes nutzt, um einen linearen quadratischen Regulator mit Methoden der parametrischen Sensitivitätsanalyse anzupassen. Desweiteren wird ein Ansatz vorgestellt, der zusätzlich die Informationsgüte in den herangezogenen Daten untersucht, um die Erfolgswahrscheinlichkeit der Parameterschätzung zu prognostizieren. Dieses Vorgehen ermöglicht es bei unzureichender Informationslage die Parameterschätzungen auszusetzen. Als eine dritte Option wird ein adaptiver Regelalgorithmus vorgestellt, der auf einem nichtlinearen modellprädikitiven Regler (NMPC) und online Parameteridentifikationen beruht. Der NMPC Ansatz basiert auf der wiederholten Lösung von Optimalsteuerungsproblemen, die in ihren Nebenbedingungen unter anderem die Systemdynamik beinhalten. Dadurch hängt die Performance stark von der Güte des Modells ab. Die kontinuierlichen Anpassungen dieses Modells im laufenden Betrieb ermöglichen eine Adaption des gesamten Reglers an sich änderende Systemparameter. Alle diskutierten Methoden basieren dabei auf hocheffizienten numerischen Verfahren zur Lösung von nichtlinearen Optimierungsproblemen (NLP) und der Möglichkeit verwandte Probleme aus der optimalen Steuerungstheorie per Diskretisierung in ein solches NLP zu überführen.

Um die Leistungsfähigkeit der Ansätze zu evaluieren, werden umfangreiche Simulationen durchgeführt und mit den existierenden Standardregelungsmethoden verglichen. Die Ergebnisse zeigen, dass die vorgeschlagene online Parameteridentifikation eine signifikante Verbesserung der Regelungsleistung im Vergleich zu herkömmlichen statischen Ansätzen bietet. Insbesondere ermöglicht sie die Kompensation von Modellunsicherheiten, Störungen und Änderungen der Systemdynamik, was zu einer robusten Regelungsfähigkeit führt.

Vorwort

Die vorliegende Arbeit ist während meiner Tätigkeit als wissenschaftliche Mitarbeiterin an der Universität Bremen am Zentrum für Technomathematik entstanden. Ich danke der Deutschen Forschungsgemeinschaft (DFG) für die Förderung im Rahmen des Graduiertenkollegs "Π^3 Parameter Identification - Analysis, Algorithms, Applications" GRK 2224/1.

An dieser Stelle möchten ich mich bei allen Menschen bedanken, die dazu beigetragen haben, dass diese Dissertationsschrift entstehen konnte. An erster Stelle möchte ich mich bei meinem Erstgutachter Prof. Dr. Christof Büskens bedanken, der diese Arbeit ermöglicht hat und jede Richtung, die ich eingeschlagen habe, unterstützt hat. Weiter möchte ich mich bei Herrn Prof. Dr.-Ing. Kai Michels bedanken, der sich bereit erklärt hat als Zweitgutachter für meine Arbeit bereitzustehen. Ich bedanke mich auch ganz herzlich bei Prof. Dr. Kathrin Flaßkamp. In unseren gemeinsamen Diskussionen sind die grundlegenden Ideen für diese Arbeit entstanden und ihre Begeisterung für dynamische Systeme und die Optimalsteuerung sind immer ansteckend gewesen.

Desweiteren möchte ich mich bei allen Kollegen aus der Arbeitsgruppe Optimierung und Optimale Steuerung, Topas und dem Graduiertenkolleg für die gute Zusammenarbeit bedanken. Es hat immer eine Atmosphäre der Offenheit und Hilfsbereitschaft geherrscht. Die vielen kleinen Gespräche, großen Diskussionen, Hilfestellungen bei Programmierungsfragen und auch das Korrekturlesen haben zum Erfolg dieser Arbeit beigetragen. Vielen lieben Dank an Chathura Wanigasekara, Maria Höffmann, Kai Wah Chan, Ruggero Simonelli, Malin Lachmann, Arne Berger und Christian Janson. Ein extra Dankeschön möchte ich an Kai Schäfer aussprechen, der als mein Bürogefährte ein toller Kamerad auf dem Weg zur Doktorarbeit war. Besonders hervorheben möchte ich außerdem Dr. Matthias Knauer, der mich als Studentin mit seiner Begeisterung als Lehrender dazu gebracht hat, immer weiter und tiefer mit Mathematik arbeiten zu wollen und ohne den ich bestimmt nicht so weit gekommen wäre.

Mein größter Dank gilt meiner Familie. Vielen Dank, Mama und Papa, dass ihr mich immer in allen Höhen und Tiefen unterstützt habt und mir, insbesondere in Fragen der Kinderbetreuung, den Rücken freigehalten habt! Vielen Dank Christian, du bist mein Rückgrat. Auch wenn du in schwierigen Phasen der Arbeit einmal zurückstecken musstest, hast du mich in meinem Bestreben immer bekräftigt. Und schlussendlich danke ich meinen Kindern, Emma und Jakob, die vielleicht nicht produktiv zu dieser Arbeit beigetragen haben, die aber mit nur einem Lächeln meine Motivation unendlich steigern können.

Contents

List of Figures

List of Tables

List of Algorithms

1. Introduction

This chapter briefly overviews the importance of parameter identification and its use to improve control algorithms in the presented work. Additionally, the dissertation's key contributions are outlined, and an overview of the entire thesis is given.

1.1. Motivation and Rationale

Parameter identification is the process of estimating unknown quantities within a mathematical model. The numerical problem of *parameter identification (PI)*, also called *parameter estimation*, *nonlinear regression* or *data fitting* is one of the most encountered issues in many practical applications that rely on mathematical modeling. These problems classically originate from almost all major scientific fields, such as engineering [BC12; BLM09], robotics [SVD07; YG18], chemistry [Küp+09], medical technology [Ges+17; Sch08] or financial mathematics [YLP20]. Such a system model is often designed linearly since the analysis and numerical methods are easier to handle and based on a profound theory. Nevertheless, most processes are nonlinear in practice, so a fine-tuned nonlinear model often better reproduces the system's behavior. If this model is time-dependent, then it is called a dynamical system, and the arising problems are so-called *nonlinear dynamic parameter identification problems (NDPIP)*. This type of problem often occurs when a mathematical model is to be generated for an actual system process as close to reality as possible. Nowadays, the term *digital twin* is often used in this context, cf. [Liu+22; Tao+22]. For example, if a robotic system such as an industrial robotic manipulator or an autonomous car is to be controlled, a model that is as accurate as possible is required. With this model, the system behavior can be planned and predicted by describing how the available control variables affect the process under consideration. It enables the user to predict and analyze the process behavior. Additionally, most industrial plants and processes should also operate optimally with regard to a predefined criterion, such as time, costs, or safety. The performance of the optimization methods used for this purpose is often based on the accuracy of the employed model, see Nelles [Nel01].

The general routine of finding a suitable system model is called *system identification.* This term covers the entire workflow, from defining the model design and model structure to data collection, estimation of individual model parameters, and validation of the model with test data. The standard approach to system identification is that all of the above points are satisfactorily addressed prior to system operation. The model is defined and identified before the system is running and is then used for all further actions and tasks with the real process, such as operational planning, development of a control strategy, and optimization. However, in reality, hardly any process can be described precisely in its entirety, and the model found remains error-prone. Furthermore, changes in the real system may occur over time, such as wear or changed operating conditions. Therefore, it is desirable that the mathematical model is also adapted during operation. For this reason, we propose to perform the parameter estimation online. Here, online means that the parameter identification is performed during system execution.

Suppose we want to control a certain system while optimizing it with respect to a specific objective. The theoretical approach to such a problem is *open-loop control*. In this context, an optimal control curve is computed based on the currently available model information and then executed on the system. In general, however, models are prone to inaccuracies and may deviate from the optimal solution. This implies that the previously generated control curves are no longer optimal. Consequently, large model errors can prevent the control objective from being achieved. Furthermore, the predicted costs may not be achieved, which usually corresponds to a cost increase. In addition, constraints may be violated, which is particularly critical in many applications, as they are often technical constraints, system limits, or even security regulations, cf. [Gle23; Brü+21].

Typically, deviations caused by model errors or external disturbances are compensated by *feedback control*. There, current system information, such as state measurements, is used to counteract deviations and stabilize the system. The flow of information from the system output to the controller closes the control loop, so the term *closed-loop control* is often chosen as well. Feedback control usually requires an additional control effort that is added to a reference control to compensate for disturbances. For example, this reference control can be a previously calculated optimal open-loop control. The goal of *optimal feedback control* is to transform an actual state into the desired state by state feedback while optimizing for a specific criterion. One of the most common approaches in engineering is the *linear quadratic regulator (LQR)* or *Riccati controller*. It is part of the class of linear optimal control methods. Modern control theory offers a comprehensive basis for optimal controllers for linear systems, see Anderson et al. [APD71; KS72; HRS06]. The quality of the controller, however, depends on the accuracy of the model used to determine the feedback law. If the system changes, the feedback gain must be recalculated. This task is usually too time-consuming to be performed during system operation. Büskens [Büs09] introduced a solution to this problem by addressing the LQR problem as a nonlinear parametric optimization problem and using results from the theory of parametric sensitivities to adapt the control law to known parametric disturbances. In his work, Tietjen [Tie12] showed the equivalence between nonlinear optimization problems and linear quadratic control problems. In a case where variations in the model parameters are given, this allows the approximation of parameter-perturbed solutions of the LQR problem using methods of parametric sensitivity analysis. Further, these approximation methods are extended to optimal observer and observer-control methods. In both works, it is assumed that the parametric deviations are known.

The disadvantage of linear control approaches for nonlinear processes is that, despite accuracy and adaptation, linearization always introduces an additional error. The transition to using nonlinear control methods enables the direct use of nonlinear system models instead. With the steady increase in computing power, more and more nonlinear approaches in feedback control are being used in real-world applications. A well-known modern method is *nonlinear model predictive control (NMPC)*, classified as an optimal nonlinear closed-loop control method. The idea is to solve an *optimal control problem (OCP)* on a certain time horizon each time new measurements are taken. Then, only the first part of the optimized control up to the next sample is executed on the system. Next, the time interval is shifted forward, and an optimal control problem is solved anew. During this process, new measurement information can be included, such as initial values for the system states in the optimal control problem. That is why NMPC can be interpreted as a feedback control method that compensates for disturbances in the system. A detailed description of the method can be found in Rawlings et al. [RMD17] and Grüne and Pannek [GP17]. The accuracy of the system model utilized has a significant impact on the controller performance. Schomakers [Sch14] observed for an NMPC controller for trajectory optimization that while state errors were adequately compensated for, the correction of model errors proved difficult. Therefore, Seelbinder [See17] suggests that a combination of the NMPC algorithm and adaptive model would be preferable.

Feedback control offers a good possibility to react to general unpredictable disturbances or system internal errors. In general, however, no adjustments are made to the models used in the control algorithm during the execution of the process. Existing approaches to provide an online adaptation of the system models for the linear as well as for the nonlinear case can be found in *adaptive control methods*, see Åström and Wittenmark [ÅW09]. They are designed to adapt the controller to unknown or changing parameters in the system. In the linear domain, *self-tuning regulators (STR)* [ÅW73; ÅW09], *model-reference adaptive systems (MRAS)* [Lan74], *gain scheduling* [RS00; AA00] and *stochastic adaptive control* [Wit75] are well-known. It is distinguished between indirect and direct adaptive controllers. Indirect adaptive controllers use methods based on online parameter estimation of model parameters and then use this information to compute a new feedback gain matrix. Direct adaptive control methods directly estimate the controller parameters, i.e., the gains. In this context, the control algorithms developed in this thesis can be considered as indirect adaptive controllers. For the presented control strategies, different established methods of optimal feedback control, such as Riccati control and NMPC, are combined with online nonlinear parameter identification. In this case, the estimation problem provides current parameter estimates for the model used in the controller so that it can be adapted to changes in the system.

1.2. Thesis Goals and Contributions

This thesis aims to design and develop control algorithms for nonlinear dynamical systems that can be adapted to unknown parametric perturbations in the system. For this purpose, nonlinear optimization methods for the online identification of model parameters and techniques of linear and nonlinear optimal control are combined. Two approaches are followed here. On the one hand, an adaptive linear-quadratic controller based on parametric sensitivities is extended by an online parameter identification method. This provides current estimates of parametric disturbances during system execution. On the other hand, a nonlinear model predictive controller is combined with online parameter estimation. This advanced nonlinear control method allows the direct use of the nonlinear system model, which is expected to result in higher prediction quality and better control performance.

In all approaches, nonlinear parameter identification determines unknown or time-varying model parameters from online collected measurement data. These parameters can be integrated into the considered system in an arbitrary nonlinear way, i.e., there is no restriction to a specific type of model. To identify the parameters, methods of nonlinear optimization are applied. For this, the error between the measurements and the output of the nonlinear model is minimized. A modern SQP-based solver for nonlinear optimization problems is utilized. This approach allows high flexibility in the objective function of the PI problem as well as in desired constraints. Numerical investigations of two application examples demonstrate the real-time capability of the applied parameter estimation methods.

The first algorithm combines nonlinear parameter identification with an adapted Riccati controller. The conditions that must be imposed on the PI to further fulfill the requirements to apply the sensitivity-adapted LQ controller are addressed. Furthermore, the problem of having enough informative data for online identification is discussed. In this context, the advantages and disadvantages of an additional inspection of the level of information in the data in the form of an approximation of the Fisher information matrix are discussed. The resulting second adaptive control and identification algorithm is numerically verified using the example of an inverted pendulum on a controlled cart.

Another goal of the thesis is to extend a nonlinear model predictive controller with real-time nonlinear parameter estimation to improve its performance. The efficiency and accuracy of NMPC depend strongly on the quality of the predictions made within the algorithm. By regularly adapting the system model, improved control results can be achieved. Furthermore, this third presented algorithm can tackle parametric perturbations in the model, which lead to divergence in the standard NMPC approach. Compared to the classical method, these advantages are shown in a comprehensive numerical analysis using the example of the pendulum-cart system. Furthermore, the performance of the presented adaptive model predictive controller is validated on the highly nonlinear example of an industrial robotic manipulator.

Contributions

This thesis investigates the connection between nonlinear online parameter identification and optimal control algorithms for dynamical systems. The scientific contribution of this thesis can be divided into three central aspects.

- The first aspect deals with the formal requirements for nonlinear online parameter identification to enable real-time sensitivity-based corrections for a linear quadratic regulator using the estimated parameters. To the author's knowledge, there is no other research in this direction. Previous works by Büskens [Büs98] and Tietjen [Tie12] assume known parametric perturbations. This thesis also presents an extension with identifiability checks to facilitate practical applications.
- The second major focus is on the numerical investigation of the real-time applicability of the nonlinear parameter identification method in the context of nonlinear model predictive control. The numerical investigations show that online model adaptations positively influence the stability behavior of the control algorithm.
- As a third contribution, it is demonstrated that all treated nonlinear problems, parameter identification, LQR, and optimal control, can be efficiently solved using the same approaches from optimal control and thus the same software TransWORHP. It is demonstrated that the full discretization approach provides a universal real-time solution method for these problems.

1.3. Thesis Structure

The thesis is divided into seven chapters. This introduction chapter is followed by five main chapters and a concluding one. In Chapter 2, we describe the problem of parameter identification. We discuss what parameter identification is commonly used for and how we consider it here in particular. In Section 2.2, we introduce the basic nonlinear optimization theory. The formulation of a general nonlinear optimization problem (NLP) and the necessary and sufficient optimality conditions for the solution are presented. Furthermore, the numerical solution method of sequential quadratic programming (SQP) is outlined, which will be used as a standard solution method in this thesis. The parameter-dependent nonlinear optimization problem is formulated, and the main features of the parametric sensitivity analysis are presented. With the main theorem of parametric sensitivity analysis, real-time strategies for approximating perturbed solutions can be formulated. This is the basis for later formulating a perturbation-adaptive linear quadratic regulator.

Based on the theory of nonlinear programming, the nonlinear dynamic parameter identification problem is formulated in 2.3. Next, Section 2.4 describes our preferred solution approach using full discretization of the dynamics and subsequent solution of the problem as NLP. The advantages and shortcomings of this approach compared to other commonly used approaches are discussed. Further, special aspects of solving parameter identification problems are discussed, such as the choice of the initial guess, the scaling of system states, and the sufficient information content of trajectories for identification.

Chapter 3 focuses on the basics of the linear quadratic regulator. It starts in Section 3.1 by introducing the basic definitions of linear and nonlinear control systems. Based on this, Section 3.2 presents the linear quadratic regulator problem (LQR). Besides the classical solution of this problem via the algebraic Riccati equation, an alternative way via formulating and solving it as an optimization problem is provided. As a next step, parametric dependencies are introduced to the LQR problem, and it is shown that by transformation to an NLP, it is possible to use methods from parametric sensitivity analysis. In particular, this allows a real-time approximation of the solution of the perturbed LQR problem from the nominal solution of the unperturbed problem using sensitivity differentials.

The crossover to nonlinear control is made in the following Chapter 4. First, the basic idea of nonlinear model predictive control (NMPC) is explained, followed by the necessary basics of optimal control and the solution of such problems by direct methods in Section 4.2. Therein, the same approach of full discretization of the dynamical system is pursued as described in Chapter 2. Then, the algorithm for classical NMPC is presented in more detail. Section 4.4 introduces different types of NMPC formulations and provides the associated stability theory. Characteristics that are crucial in the design of a model predictive controller, such as the length of the prediction horizon, the computation times for solving the individual OCPs, or the quality of the prediction model, are highlighted in 4.5. In Section 4.6 the chapter concludes by introducing existing approaches to handle uncertainties in the NMPC setting.

In Chapter 5 an adaptive control approach is developed that uses the previously presented approaches for nonlinear parameter identification and the LQ regulator with real-time approximations from Section 3.3. First, we discuss the formulation options for the involved PI problem. Then, in Section 5.2, the first adaptive control algorithm is presented in detail. Additionally, it is described in Section 5.3 how computations of the Fisher information matrix can help to assess the success probability of the PIs. Finally, the functionality of the proposed control methods is demonstrated by numerical investigations using the benchmark problem of an inverted pendulum on a cart system in 5.4. In particular, the influence of time intervals with insufficient data for PI is discussed.

An equivalent approach is followed in Chapter 6 to develop the nonlinear adaptive controller. Here, the classical nonlinear model predictive control approach is extended by online parameter identification. First, the exact formulation of the parameter estimation problem is discussed. Then Section 6.2 presents the combined algorithm. This is followed by a numerical analysis of the functionality and efficiency of the presented controller on the pendulum-cart system in 6.4. To further prove the strong closed-loop performance of the proposed controller, a two-joint industrial robot is used in numerical simulations comparing the proposed algorithm to standard NMPC.

This thesis concludes with a summary of the results and a discussion of possible directions for future research in Chapter 7. Two sections in the Appendix provide additional information on the numerical evaluations in Chapters 5 and 6. These are the computation of the Fisher information matrix and additional figures on the numerical evaluations for different scenarios for the inverted pendulum on a cart.

2. Nonlinear Parameter Identification

As the introduction indicates, parameter estimation in dynamical systems is relevant in many application domains. In the same way, many different mathematics disciplines are involved or closely related to it. The most relevant ones concerning this work are shown in Figure 2.1. The areas of Modeling and System Identification can be understood here more as encompassing terms, of which PI is an essential component. Including data knowledge in predefined models and identifying specific model parameters make these models valuable for practitioners. Nonlinear optimization methods and numerical integration techniques for solving nonlinear differential equations are fundamental mathematical concepts needed to solve optimization problems with differential equations as constraints. Furthermore, statistical approaches for the determination of model parameters can allow further statements about the solution, e.g., the calculation of so-called confidence intervals, in which the solution lies with a certain probability. In addition, there is the theoretical application-specific knowledge, for example, with a mechanical or electrical engineering background, associated with the problem. All this makes the parameter identification process a highly complex problem. It should be noted that it is beyond the scope of this dissertation to address all of these issues to their full extent. In this thesis, the focus is on the methods of numerical optimization. The necessary basics are presented for the other fields, and references to further literature are given.

Within this chapter, we first highlight the relevance and significance of *nonlinear dynamical parameter identification problems (NDPIP)* in research and society. Then, the mathematical fundamentals of nonlinear constrained optimization are discussed, which are used as a basis for the further theory of the numerical solution of NDPIP. Finally, some difficulties and pitfalls in this context are discussed. The structure of this chapter and the theoretical background to the numerical solution of NDPIP are based on a book by Schittkowski [Sch02]. The notation and the introduction to optimization theory are oriented towards the work of Büskens [Büs98] and Echim [Ech14].

2.1. Introduction and Motivation

Parameter identification methods can build a connection between data and models. The challenge in implementing a theoretically developed model is that not all important parameters involved in the model are known. These unknown parameter values can be

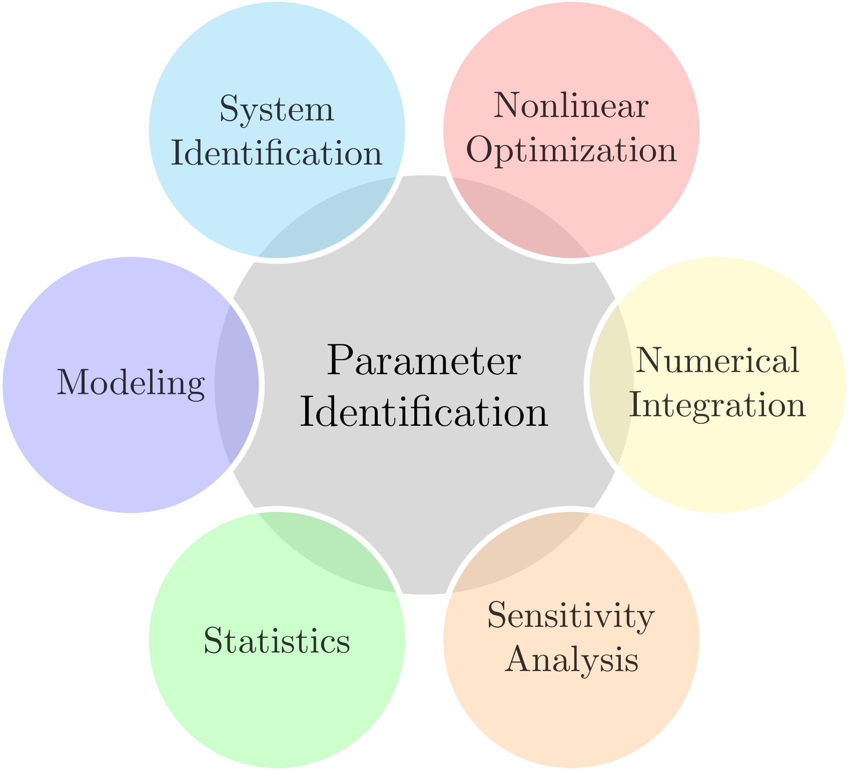

Figure 2.1.: *Mathematical disciplines involved in nonlinear parameter identification with respect to dynamical systems.*

determined by measuring data at the system input and output and evaluating them comparatively.

Why do we need mathematical models for real-world problems? – The answer is usually an economic motive. Models can be realized at low cost and offer the possibility to simulate and analyze a system, even before the construction or the final design. This can avoid costly physical tests and construction errors. Furthermore, situations that would not be possible or even dangerous in reality can be simulated, such as overload tests or other extreme situations. Furthermore, they are used for advanced control design and optimization procedures. Nelles [Nel01] states that the performance of the final solution of such an optimization or optimal control problem usually has the quality of the underlying model as an upper bound. This motivates the great need for sophisticated identification techniques.

In the following, we assume that we have a model output function of our process that gives us the output for some input variables. However, this model is supposed to contain unknown parameters. If measurements of the system outputs and the corresponding inputs are available, then the parameters can be identified using a data-driven method.

In this work, we formulate the parameter identification problem as a *least squares problem*. We utilize this special form of an optimization problem, where the sum of squares of distances of a system model from some collected measurements is minimized to estimate unknown model parameters. Within the context of parameter estimation, these distances are also called the *residuals* of the problem. If the model perfectly represented the data, the residuals would be zero.

Definition 2.1.1 (Least Squares Estimation Problem)
Let $y_i \in \mathbb{R}^{n_y}$ be measurements and $x_i \in \mathbb{R}^{n_x}$ inputs at discrete time points with $M \in \mathbb{N}$ and $i = 0, \dots, M$. Further it is $h\colon \mathbb{R}^{n_x} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_y}$ the system model function with the unknown parameter vector $p \in \mathbb{R}^{n_p}$. Then the problem

$$\min_p F(p) := \frac{1}{2} \sum_{i=0}^{M} \|h(x_i, p) - y_i\|_q^2 \tag{2.1}$$

is called **least squares estimation problem**.

The definition of the distance depends on the used norm $\|\cdot\|_q$. In the application example in Chapter 5, we will assume a normally distributed error in the measurement data and, therefore, use the Euclidean norm. Nevertheless, in Subsection 2.4.4, alternative norms for specific user-oriented cases are discussed.

The least-squares problem is well-known in mathematical programming. In the special case of a static linear model, where the output depends linearly on the parameters, the problem (2.1) above simplifies solving a system of linear equations. This happens when black-box modeling is performed using polynomials, and it can then be resolved with QR decomposition, as demonstrated in Jung [Jun19]. Such linear optimization methods, however, can only be applied if the model output is linear in the parameters. If the unknown parameters enter the model output function nonlinearly, then nonlinear optimization methods must be considered, e.g., gradient-based methods, see Schröder [Sch17].

The parameter identification problem becomes more complex because we want to consider dynamical systems. That is, the h introduced above in (2.1) is describing the solution of a dynamical system, for example, of a system of ordinary differential equations, i.e., $\dot{x}(t, p) = f(x(t), t, p)$, $t \in \mathbb{R}$. This can be considered by adding the dynamical system equations as constraints to the optimization problem. Also, for this kind of nonlinear and dynamic data fitting problem, many mathematical algorithms exist to solve problems without additional general constraints, compare Schittkowski [Sch02]. Standard approaches are the Gauss-Newton method and the Levenberg-Marquardt method as described in [GMW81].

An alternative to parameter estimation using optimization problems is the use of state observers. The well-known *Kalman Filter (KF)* introduced by Kalman [Kal60] and Kalman and Bucy [KB60] is usually applied in case of incomplete feedback to estimate the unmeasured state variables in linear problems. In the case of (weakly) nonlinear systems, the *Extended Kalman Filter (EKF)*, see Sorenson [Sor70], Anderson and Moore [AM12], or the *Unscented Kalman filter (UKF)*, Julier et al. [JUD95], can be used. Classically, these methods are known as state estimators, where a predictor-corrector method estimates the system behavior recursively. From the current state at a time, the state at the next time is predicted, and then this estimate is corrected by the new measurement. In addition, parameters can also be estimated by extending the state vector of the dynamical system by introducing a parameter of the model as a new constant system state. Knowledge of the process noise and the measurement noise is crucial for the success of the filter methods.

In the context of this work, we will further extend the above problem (2.1) to include nonlinear equality and inequality constraints. In other words, we consider the solution of general nonlinear dynamical parameter identification problems with constraints. We will employ solution approaches that are mainly known in the field of optimal control. By using direct solution methods, we remain flexible in choosing the norm in the objective function. Moreover, we may include user-defined constraints. In these direct solution methods, we first discretize the dynamics of the underlying problem by creating additional constraints and then solve the resulting *nonlinear optimization (NLP)* problem using constrained optimization methods. Therefore, in the following Section 2.2, we introduce the basics of nonlinear constrained optimization and the numerical solution method *sequential quadratic programming (SQP).*

2.2. Nonlinear Optimization

As already mentioned, we will transfer the parameter identification problem to a nonlinear optimization problem (NLP), which then has to be solved primarily. In Section 3.2, the linear quadratic regulator problem will also be reduced to a finite optimization problem. Furthermore, NLPs will appear when we reduce the optimal control problem within the NMPC algorithm in Chapter 4 to a finite NLP by time discretization and integration of the differential equations. Therefore, Section 2.2.1 gives an introduction to nonlinear optimization. Introductions to the theory can also be found in Büskens [Büs98], Jungnickel [Jun15], Durea and Strugariu [DS14] among many others. Highly recommended textbooks for a deeper introduction to the theory of optimization and detailed explanations are, for example, Geiger and Kanzow [GK02] and Nocedal [NW06].

2.2.1. Nonlinear Programming

We start with the formulation of the standard problem of nonlinear optimization, also called a *nonlinear programming problem.*

Definition 2.2.1 (Nonlinear Program)
Let $x \in \mathbb{R}^{n_x}$ and $F\colon \mathbb{R}^{n_x} \to \mathbb{R}$ and $g\colon \mathbb{R}^{n_x} \to \mathbb{R}^m$ be functions. Then the problem

$$\begin{aligned} \min_x \quad & F(x), \\ \text{w.r.t.} \quad & g_i(x) = 0, \quad i = 1, \dots, m_e, \\ & g_i(x) \leq 0, \quad i = m_e + 1, \dots, m \end{aligned} \tag{2.2}$$

is called **nonlinear program (NLP)**. The function F is called **objective or cost function** with **optimization variables** x. The vector $x = (x_1, \dots, x_{n_x})^T \in \mathbb{R}^{n_x}$ of independent variables $x_1, \dots, x_{n_x}$ is often referred to as vector of **decision variables**. Further is g the function of equality and inequality **constraints**, where m_e is the number of equality constraints.

(2.2) is formulated as a minimization problem. If a maximization is desired, this can be achieved by negating the objective function. To develop solution strategies for the presented problem (2.2), terms like *feasible set* and *local minimum* are needed. These are specified in the following.

Definition 2.2.2 (Feasible Set)
Let g be the function from Definition 2.2.1, then we call

$$\mathcal{S} = \{x \in \mathbb{R}^{n_x} | g_i(x) = 0,\ i = 1, \dots, m_e,\ g_i(x) \leq 0,\ i = m_e + 1, \dots, m\}$$

the **feasible set**. Any point $x \in \mathcal{S}$ is called **feasible point**.

Now we can define the term of a *local minimum* of the problem (2.2). It is often also referred to as *local solution* or *minimizer* of the function F.

Definition 2.2.3 (Minimum/Minimizer)
Assuming the preconditions from Definitions 2.2.1 and 2.2.2, then a vector $x^* \in \mathcal{S}$ is called a **local minimizer** of the function F on the set $\mathcal{S}$, if there exists a neighborhood $\mathcal{V}(x^*) \subseteq \mathbb{R}^{n_x}$ of x^* such that

$$F(x^*) \leq F(x), \quad \forall x \in \mathcal{S} \cap \mathcal{V}.$$

The value $F(x^*)$ is called a **local minimum** of F. If $\mathcal{V} = \mathbb{R}^{n_x}$ applies, one says that x^* is the **global minimizer** of F on $\mathcal{S}$ and $F(x^*)$ is the **global minimum** of F.

Next, we want to discuss the role of constraints. A constraint is called active if $g_i(x) = 0$, $i \in \{1, 2, \dots, m\}$ holds. Note that within the feasible set, the equality constraints are always active. When introducing necessary and sufficient optimality conditions for a local solution of (2.2) in Section 2.2.1, we will see that only active constraints influence the search for a minimum. Providing a set of active constraints is reasonable since this set can change during the numerical process. We need a special assumption on the constraints to formulate the following optimality conditions. It is called *constraint qualification* or *regularity* in a more general form.

Remark. In the following, the continuous differentiability of F and g is always assumed.

Definition 2.2.4 (Active Set, Regularity, Normality)
Assume the problem (2.2) and let $x \in \mathcal{S}$ be a feasible point of (2.2).

(i) We call the index set $\bar{I}(x) = \{i \in \{m_e+1, \dots, m\} | \ g_i(x) = 0\}$ the **set of active inequality constraints**. The entire active constraints are characterized by the index set called **active set** $I(x) = \bar{I}(x) \cup \{1, \dots, m_e\}$.

(ii) Then x is **regular**, if the gradients $\nabla_x g_i(x)$ for $i \in \{1, \dots, m_e\}$ are linearly independent and there exists a $v \in \mathbb{R} \setminus \{0\}$ with

$$\begin{aligned} \nabla_x g_i(x) v &< 0, \ \ i \in \bar{I}(x), \\ \nabla_x g_i(x) v &= 0, \ \ i = 1, \dots, m_e. \end{aligned}$$

(iii) x is **normal**, if the gradients of active constraints $\nabla_x g_i(x)$, $i \in \ I(x)$ are linearly independent.

Several formulations of these *regularity conditions* exist. The conditions (ii) in Definition 2.2.4 are also known as *Mangasarian-Fromowitz constraint qualifications (MFCQ)*. If a point x fulfills condition (iii), it satisfies the *linear independence constraint qualifications (LICQ)*. These are more restrictive than the MFCQ.

Necessary and Sufficient Optimality Criteria

The *Lagrangian function* is the most important tool for characterizing optimality conditions. In literature, different formulations can be found. For example, Schittkowski [Sch02] subtracts the constraints from the objective instead of adding them. Nevertheless, this eventually results in an equivalent formulation. We use the following definition.

Definition 2.2.5 (Lagrangian Function, Lagrangian Multiplier)
For $\lambda \in \mathbb{R}^m$, $\lambda = (\lambda_1, \dots, \lambda_m)^T$ the **Lagrangian function** $L : \mathbb{R}^{n_x} \times \mathbb{R}^m \to \mathbb{R}$ of problem (2.2) is defined by

$$L(x, \lambda) := F(x) + \lambda^T g(x) = F(x) + \sum_{i=1}^{m} \lambda_i g_i(x).$$

The variables λ_i are called the **Lagrangian multipliers** of (2.2).

In this context, x is also called the *primal* and λ the *dual variable* of the nonlinear program (2.2). Using the Lagrangian function, we now introduce the well-known optimality conditions, the *Karush-Kuhn-Tucker conditions (KKT)*, for problem (2.2).

Definition 2.2.6 (Karush-Kuhn-Tucker Conditions)
Consider problem (2.2) with continuously differentiable functions F and g.

(i) Then the conditions

(a) $\nabla_x L(x, \lambda) = \nabla_x F(x) + \lambda^T \nabla_x g(x) = 0$,
(b) $\lambda_i \geq 0, \ i \in \bar{I}(x)$,
(c) $\lambda_i = 0, \ i \notin I(x)$,

are called **Karush-Kuhn-Tucker conditions** of (2.2).

(ii) A vector $(x, \lambda) \in \mathbb{R}^{n_x} \times \mathbb{R}^m$ that satisfies the KKT-conditions is a **Karush-Kuhn-Tucker point**, **KKT-point** or also **critical point**.

The following theorem gives necessary first-order optimality conditions. It states that at a local solution, the gradient of the objective function can be represented by a linear combination of gradients of active constraints. The Lagrangian multipliers specify the influence of the constraints on the objective function.

Theorem 2.2.7 (Karush-Kuhn-Tucker Condition with MFCQ, [GK02])
Let $x^* \in \mathbb{R}^{n_x}$ be a local minimizer of (2.2) and x^* fulfills the MFCQ. Then there exist Lagrangian multipliers $\lambda^* \in \mathbb{R}^m$, such that the tuple (x^*, λ^*) is a KKT-point of (2.2).

If we strengthen the constraint qualification to LICQ, the Lagrangian multipliers are even unique.

Theorem 2.2.8 (Karush-Kuhn-Tucker Condition with LICQ, [GK02])
Let x^* be a local minimizer of (2.2) and x^* fulfills the LICQ. Then there exist unique Lagrangian multipliers $\lambda^* \in \mathbb{R}^m$, such that the tuple (x^*, λ^*) is a KKT-point of (2.2).

The KKT conditions extend the general condition that in the optimum, the derivative of the objective function must be equal to zero, $\nabla_x F(x^*) = 0$, to constrained problems. Theorem 2.2.8 shows that inactive inequality constraints do not affect the optimality conditions because their corresponding Lagrangian multipliers equal zero.

It is important to note that every local minimizer is a critical point, but not every critical point is also directly a local solution to the problem. Properties of the Hessian matrix of the Lagrangian function are used to formulate the necessary and sufficient second-order conditions in order to identify the local minimizer from the set of critical points.

Theorem 2.2.9 (Necessary and Sufficient 2nd Order Conditions, [GK02]) Let $x^* \in \mathbb{R}^{n_x}$ be a local minimum of the problem (2.2) and x^* be normal (x^* satisfies the LICQ). Further, let F be two times differentiable and g_i be two times continuously differentiable for $i = 1, \ldots, m$ in a neighborhood of x^*. Then for the **critical cone**

$$\begin{aligned}\mathcal{C}(x^*) \coloneqq \{v \in \mathbb{R}^{n_x} | \nabla_x g_i(x^*)v = 0, \text{ for } i \in \bar{I}(x^*) \text{ and } \lambda_i > 0,\\ \nabla_x g_i(x^*)v \leq 0, \text{ for } i \in \bar{I}(x^*) \text{ and } \lambda_i = 0,\\ \nabla_x g_i(x^*)v = 0, \text{ for } i = 1, \ldots, m_e\},\end{aligned}$$

the following holds.

(i) **(Necessary 2nd-Order Conditions)**
There exist uniquely defined Lagrangian multipliers $\lambda^* \in \mathbb{R}^m$ and the Hessian matrix of the Lagrangian function is positive semidefinite on the critical cone:

$$v^T \nabla^2_{xx} L(x^*, \lambda^*) v \geq 0, \ \forall \ v \in \mathcal{C}(x^*).$$

(ii) **(Sufficient 2nd Order Conditions)**
Let the Hessian matrix satisfy the condition

$$v^T \nabla^2_{xx} L(x^*, \lambda^*) v > 0, \ \forall \ v \in \mathcal{C}(x^*) \setminus \{0\},$$

then there exist a neighborhood $U(x^*)$ of x^* and a constant $c > 0$ such that x^* is a strong local minimizer of (2.2):

$$F(x) \geq F(x^*) + c\|x - x^*\|^2, \ \forall \ x \in \mathcal{S} \cap U(x^*).$$

The necessary and sufficient conditions can thus be used to set up an algorithm to search for local solutions of (2.2). Usually, one is initially satisfied with the search for KKT-points of (2.2), even if this might lead to saddle points or even maximizers. If a candidate is found, optimality can be checked using the criterion from Theorem 2.2.9.

At this point, we want to point out that non-active constraints do not influence the evaluation of necessary or sufficient conditions. This is due to the relation $\lambda_i = 0 \ \forall i \notin I(x)$ from Definition 2.2.6. For this reason, a so-called vector of active constraints is often used in numerical solution methods. We define the *vector of active constraints* and the corresponding Lagrangian multiplier as

$$g^a \coloneqq (g_i)_{i \in I(x^*)}, \quad \lambda^a \in \mathbb{R}^{n_a}, \quad \text{where } n_a \coloneqq |I(x^*)|. \tag{2.3}$$

This notation will simplify the following advanced formulations.

Sequential Quadratic Programming

Numerous approaches are available for numerically solving the presented nonlinear optimization problems. We focus here on *sequential quadratic programming (SQP)* since this method is used to compute the numerical results presented in the following chapters. Further solution methods and the associated theory are described in more detail in Geiger and Kanzow [GK02] and Nocedal and Wright [NW06].

Sequential quadratic programming is among the best-known and most powerful optimization algorithms to solve problems of the form (2.2). It is based on iteratively solving quadratic subproblems to determine a descent method's search direction to get further iterations. For this purpose, the nonlinear program (2.2) is locally approximated by a convex subproblem. Such a subproblem is usually obtained by a quadratic approximation of the Lagrangian function and a linearization of the constraints. In the next step of the algorithm, the solution of the subproblem is used as a search direction of the descent method to solve the original problem.

In general, a descent method is designed to find a minimum by iteratively moving in a search direction $d^{[k]} \in \mathbb{R}^n$ by step sizes $\alpha^{[k]} \in \mathbb{R}$ starting from an initial estimate $x^{[0]}$ with the iteration counter $k \in \mathbb{N} \setminus \{0\}$. The k-th iteration is given by

$$x^{[k+1]} = x^{[k]} + \alpha^{[k]} d^{[k]}. \tag{2.4}$$

The search direction should be a descent direction of the Lagrangian function.

Definition 2.2.10 (Descent Direction)
Let $f\colon \mathbb{R}^{n_x} \to \mathbb{R}$ be a mapping. The vector $d \in \mathbb{R}^{n_x}$, $d \neq 0$ is a **descent direction** of f in x, if there exists an $a \in \mathbb{R}$, $a > 0$ with

$$f(x + \alpha d) < f(x), \quad \text{for all } \alpha \in (0, a).$$

A sufficient condition for a descent direction is given in Jungnickel [Jun15] with the following theorem.

Theorem 2.2.11 ([Jun15])
Let $f\colon \mathbb{R}^{n_x} \to \mathbb{R}$ be a function that is differentiable in $x \in \mathbb{R}^{n_x}$. Then every vector $d \in \mathbb{R}^{n_x}$ with

$$\nabla f(x)^T d < 0$$

is a descent direction of f in x.

In every main iteration of the SQP method, the problem (2.2) is locally approximated by a quadratic subproblem. This problem has the optimization variables $d^{[k]} = x^{[k+1]} - x^{[k]}$ to find a descent direction of the Lagrangian.

Definition 2.2.12 (Quadratic Subproblem)
The quadratic optimization problem

$$
\begin{aligned}
&\min_{d^{[k]} \in \mathbb{R}^{n_x}} \frac{1}{2} {d^{[k]}}^T \nabla^2_{xx} L\left(x^{[k]}, \lambda^{[k]}\right) d^{[k]} + \nabla_x F\left(x^{[k]}\right)^T d^{[k]} \\
&\text{w.r.t. } g_i\left(x^{[k]}\right) + \nabla g_i\left(x^{[k]}\right)^T d^{[k]} = 0, \quad i = 1, \ldots, m_e, \\
&\qquad\quad g_i\left(x^{[k]}\right) + \nabla g_i\left(x^{[k]}\right)^T d^{[k]} \leq 0, \quad i = m_e + 1, \ldots, m,
\end{aligned}
\tag{2.5}
$$

is called **quadratic subproblem of** (2.2).

Very efficient solution methods exist for this quadratic problem to solve the subproblems more efficiently and reliably, [Gef17]. Common solution methods are the *active-set strategy* and the so-called *interior-point methods*. This thesis uses the NLP-solver WORHP and the implemented SQP method applies the interior point method.

If the Hessian of the Lagrangian function $\nabla^2_{xx} L(x^{[k]}, \lambda^{[k]})$ is positive definite, then the solution $d^{[k]}$ of (2.5) is a descent direction of the Lagrangian function in $x^{[k]}$ and can be used as search direction for (2.2). In many applications, however, the exact computation of the Hessian is impossible or would mean a disproportionate effort. A possible approach to solve the quadratic subproblems is the approximation of the Hessian by finite differences. Increasingly, automatic differentiation methods, such as the automatic differentiation library ADOL-C, are used, see Walther and Griewank [WG09]. In many implementations of SQP methods, the Hessian is approximated by a Quasi-Newton approximation, where the Hessian is replaced by a positively definite matrix $H^{[k]}$ in each iteration step. This is computed from the previous iteration using an update formula. Well known approaches are the *BFGS methods* from Broyden [Bro70], Fletcher [Fle70], Goldfarb [Gol70] and Shanno [Sha70]. The update of the Hessian then results in:

$$
\begin{aligned}
H^{[k+1]} &= H^{[k]} + \frac{y^{[k]} {y^{[k]}}^T}{{y^{[k]}}^T d} - \frac{H^{[k]} d^{[k]} {d^{[k]}}^T H^{[k]}}{{d^{[k]}}^T H^{[k]} d^{[k]}}, \\
\text{with } d^{[k]} &= x^{[k+1]} - x^{[k]} \text{ and } y^{[k]} = \nabla F\left(x^{[k+1]}\right) - \nabla F\left(x^{[k]}\right).
\end{aligned}
\tag{2.6}
$$

The positive definiteness of the sequence $\{H^{[k]}\}_{k \in \mathbb{N}}$ can only be guaranteed in combination with the Wolfe-Powell rule for the step sizes $\alpha^{[k]}$ in equation (2.4). If the Armijo rule is to be applied as a step size strategy, the *modified BFGS formula* is recommended by Powell [Pow78]. If $H^{[k]}$ is symmetric and positive definite, then also $H^{[k+1]}$ is symmetric and positive definite, see Geiger and Kanzow [GK02].

By ensuring that the Hessian or its approximation is positive definite, we establish that $d^{[k]}$ is a descent direction of the Lagrangian function. This search direction can then be used to update the state by $x^{[k+1]} = x^{[k]} + \alpha^{[k]} d^{[k]}$. It can be shown that this local SQP method converges superlinear for initial values $(x^{[0]}, \lambda^{[0]})$ in a certain neighborhood of a local minimum x^*. With further assumptions, even quadratic convergence can be

Algorithm 2.1 Sequential Quadratic Programming (SQP)

```
Choose initial guesses $(x^{[0]}, \lambda^{[0]}) \in \mathbb{R}^{n_x} \times \mathbb{R}^m$ and $H^{[0]} \in \mathbb{R}^{n_x} \times \mathbb{R}^{n_x}$ symmetric and positive definite.
$k = 0$
if $(x^{[k]}, \lambda^{[k]})$ is a KKT-point of (2.2) then
    STOP
else
    Compute a solution $d^{[k]}$ of (2.5) in $x^{[k]}$ with corresponding Lagrangian multipliers $\lambda^{[k+1]}$, where $\nabla^2_{xx} L(x^{[k]}, \lambda^{[k]})$ is replaced by $H^{[k]}$.
    Determine the step size $\alpha^{[k]}$.
    Compute new estimate $x^{[k+1]} = x^{[k]} + \alpha d^{[k]}$.
    Compute $H^{[k+1]}$.
    $k = k + 1$
end if
```

proven. In general, however, this neighborhood of initial values around x^* for that the algorithm converges is unknown. To achieve convergence for arbitrary initial values, *globalization strategies* are used. For this purpose, the step size α is considered when calculating the new iterated values in equation (2.4). To determine a suitable step size, a line search is done in the direction of d_k, where d_k is the solution of the quadratic subproblem. A common method is the one-dimensional line search according to the Armijo method, [Arm66].

At this point, we can formulate the described SQP method in Algorithm 2.1. In the best case, the globalized SQP method converges to the local one after a finite number of steps and then shows at least a superlinear convergence.

2.2.2. Parametric Nonlinear Programming

In practical applications, the objective function to be minimized often depends not only on the optimization variables $x \in \mathbb{R}^{n_x}$, but also on parameters $p \in P \subset \mathbb{R}^{n_p}$. Therefore, parametric nonlinear optimization problems are of special interest in this work. For this reason, the formulation of (2.2) in Definition 2.2.1 is extended by nonlinear parameters to a *parametric nonlinear program*, short NLP(p). The parameters p are also called *perturbations*. In the following, the concept of parametric sensitivity analysis is introduced, and its fundamental theorem is presented. These approaches will be used in the further steps of this thesis to enable real-time updates for the linear quadratic regulator.

According to Definition 2.2.1 we define the *parametric nonlinear program*, also called *perturbed nonlinear program*, as follows.

Definition 2.2.13 (Parametric Nonlinear Program)
Let $x \in \mathbb{R}^{n_x}$, $p \in P \subset \mathbb{R}^{n_p}$ and $f\colon \mathbb{R}^{n_x} \times \mathbb{R}^{n_p} \to \mathbb{R}$ and $g\colon \mathbb{R}^{n_x} \times \mathbb{R}^{n_p} \to \mathbb{R}^m$ be functions. The problem

$$\begin{aligned} \min_{x} \quad & F(x,p), \\ \text{w.r.t.} \quad & g_i(x,p) = 0, \quad i = 1,\dots,m_e, \\ & g_i(x,p) \leq 0, \quad i = m_e+1,\dots,m. \end{aligned} \tag{2.7}$$

is called **parametric nonlinear program (NLP(p)).**

For a fixed reference value $p = p_0 \in P$ and $F(x) \coloneqq F(x,p_0)$ the definition of a minimum point of (2.7) is analogous to Definition 2.2.3. In the same way, the statements for the necessary and sufficient optimality conditions in Propositions 2.2.7, 2.2.8, and 2.2.9 can be formulated for a fixed $p_0 \in P$. For the *nominal perturbation* $p_0 \in P$, the problem $NLP(p_0)$ is denoted as *unperturbed or nominal problem.*

2.2.3. Parametric Sensitivity Analysis

The parametric sensitivity analysis of nonlinear optimization problems is a technique to systematically analyze the influence of parameters on the optimal solution of the optimization problem. Fiacco et al. [Fia+83] provide a comprehensive topic overview. Their work covers the basic results of parametric sensitivity analysis of nonlinear optimization problems and lays the foundation for the following work in this area. We will only discuss the essential results for the further chapters and follow the arguments of Büskens [Büs02].

Considering the nominal problem NLP(p_0) for a fixed nominal parameter $p_0 \in P$, we can study the differentiability properties of the optimal solution in a neighborhood of this nominal parameter p_0 with respect to perturbations p. An important consequence of the sufficient second-order optimality conditions of Subsection 2.2.1 is that the optimal solution $x(p)$ and the corresponding Lagrangian multipliers $\lambda(p)$ of NLP(p_0) become differentiable functions of the parameter p. The following theorem ensures the existence of such functions and their feasibility.

Theorem 2.2.14 (Parametric Sensitivity Theorem on the Differentiability of Optimal Solutions)
Let F and g be defined as in Definition 2.2.13 and be twice continuously differentiable with respect to x. Further, let $\nabla_x F$, $\nabla_x g$ and g be once continuously differentiable with respect to p. Further assume that $x^* \in \mathbb{R}^{n_x}$ is a strictly regular local solution of (2.7), i.e. $(x^*, \lambda^*) \in \mathbb{R}^{n_x} \times \mathbb{R}^m$ fulfills the necessary and sufficient optimality conditions for a fixed $p_0 \in P$. Then there exists a neighborhood $P_0 \subset P$ of p_0

and two uniquely defined, continuously differentiable functions $x\colon P_0 \to \mathbb{R}^n$ and $\lambda\colon P_0 \to \mathbb{R}^m$ with the following properties:

(i) It is $x(p_0) = x^*$ and $\lambda(p_0) = \lambda^*$.

(ii) The active set is unchanged: $I(x(p), p) \equiv I(x(p_0), p_0)$, $\forall p \in P_0$.[a]

(iii) The gradients of the active constraints are linearly independent, i.e.

$$\operatorname{rank}(\nabla_x g^a(x(p), p)) = n_a, \quad \forall p \in P_0.$$

(iv) $x^*(p)$ and $\lambda^*(p)$ fulfill the strict sufficient second-order optimality conditions from Theorem 2.2.9 for all $p \in P_0$ and particularly $(x^*(p), \lambda^*(p))$ is a strict local minimum of (2.7).

[a] For (2.7) the active set is defined as $I\,(x(p), p) := \bar{I}\,(x(p), p) \cup \{1, \ldots, m_e\}$ with the set of active inequality constraints $\bar{I}\,(x(p_0), p_0) := \{i \in \{m_e + 1, \ldots, m\} |\; g_i(x(p), p) = 0\}$.

Proof. A detailed proof can be found in Fiacco et al. [Fia+83]. □

In Fiacco's proof, all constraints are considered. [Büs98] shortens the proof considerably by focusing on the active constraints. Reviewing some of the main elements of the argumentation guides to an explicit representation of the sensitivity derivatives using only the nominal solution. Corresponding to the definitions in (2.3) we use the notation $g^a(p) \in \mathbb{R}^{n_a}$ and $\lambda^a(p) \in \mathbb{R}^{n_a}$ for the vector of active constraints. The necessary KKT conditions from Definition 2.2.6 for the variables sought $(x(p), \lambda^a(p))$ can be written as

$$K(x^*, \lambda^{*,a}, p) := \begin{pmatrix} \nabla_x L(x^*, \lambda^{*,a}, p) \\ g^a(x^*, p) \end{pmatrix} = \begin{pmatrix} \nabla_x F(x^*, p) + (\lambda^{*,a})^T \nabla_x g^a(x^*, p) \\ g^a(x^*, p) \end{pmatrix} = 0.$$

Due to the differentiability assumptions of the Sensitivity Theorem 2.2.14 the Jacobian of $K(x^*, \lambda^{*,a}, p)$ with respect to (x, λ^a) can be calculated as

$$\nabla_{(x,\lambda^a)} K(x^*, \lambda^{*,a}, p) = \begin{pmatrix} \nabla_x^2 L(x^*, \lambda^{*,a}, p) & \nabla_x g^a(x^*, p)^T \\ \nabla_x g^a(x^*, p) & 0 \end{pmatrix}. \tag{2.8}$$

The matrix in equation (2.8) is called *Kuhn-Tucker matrix* or *KKT-matrix*. Since the sufficient conditions are assumed to be satisfied, the invertibility of the KKT matrix can be proven by some algebraic considerations [Büs98]. Thus, all preconditions for applying the classical theorem about implicit functions are given. It follows that $x^* = x^*(p)$ and $\lambda^{*,a} = \lambda^{*,a}(p)$ are differentiable functions in a neighborhood P_0 of the nominal parameter p_0 and that they satisfy the equation $K(x^*(p), \lambda^{*,a}(p), p) = 0$, for all $p \in P_0$. Next, we derive the total differential from K with respect to p by using the chain rule and evaluate it in the nominal parameter $p = p_0$. We use the abbreviating notations $x_0^* := x^*(p_0)$ and $\lambda_0^{*,a} := \lambda^{*,a}(p_0)$. This leads to a system of linear equations for the sensitivity differentials of the optimal solution and multipliers:

$$\begin{pmatrix} \nabla_x^2 L(x_0^*, \lambda_0^{*,a}, p_0) & \nabla_x g^a(x_0^*, p_0)^T \\ \nabla_x g^a(x_0^*, p_0) & 0 \end{pmatrix} \begin{pmatrix} \frac{dx}{dp}(p_0) \\ \frac{d\lambda^a}{dp}(p_0) \end{pmatrix} + \begin{pmatrix} \nabla_{x\lambda}^2 L(x_0^*, \lambda_0^{*,a}, p_0) \\ \nabla_p g^a(x_0^*, p_0) \end{pmatrix} = 0. \tag{2.9}$$

The invertibility of the KKT matrix leads to an explicit representation of the sensitivity differentials by transforming (2.9) to

$$\begin{pmatrix} \frac{dx}{dp}(p_0) \\ \frac{d\lambda^a}{dp}(p_0) \end{pmatrix} = - \begin{pmatrix} \nabla_x^2 L(x_0^*, \lambda_0^{*,a}, p_0) & \nabla_x g^a(x_0^*, p_0)^T \\ \nabla_x g^a(x_0^*, p_0) & 0 \end{pmatrix}^{-1} \begin{pmatrix} \nabla_{x\lambda}^2 L(x_0^*, \lambda_0^{*,a}, p_0) \\ \nabla_p g^a(x_0^*, p_0) \end{pmatrix}. \tag{2.10}$$

The KKT-matrix and sensitivity differentials can be obtained by post-optimality analysis, e.g., finite differences or automatic differentiation to compute the first and second-order derivatives. In the following, we use the sensitivity differentials (2.10) to compute approximations of a perturbed solution for a linear quadratic regulator, see Section 3.3.2. This is done by approximating the perturbed solution by the nominal solution for the unperturbed problem and the sensitivity differentials.

Sensitivity-based Approximation of the Perturbed Solution

The presented differentiability properties of the optimal solution of the parametric optimization problem provide the basis for strategies for real-time approximation of the perturbed solutions. The idea is to approximate the solution of problem (2.7) by a first-order Taylor expansion. This requires the nominal solution $x(p_0)$ and the sensitivity differentials $\frac{dx}{dp}(p_0)$ from Theorem 2.2.14:

$$x^*(p) \approx \tilde{x}(p) \coloneqq x(p_0) + \frac{dx}{dp}(p_0)\Delta p, \quad \Delta p \coloneqq p - p_0. \tag{2.11}$$

As the nominal solution and sensitivity derivatives can be computed offline, this estimation can be done with minimal numerical effort. If a deviation Δp from the nominal parameter value is detected, an approximation of the exact solution $x(p)$ can be achieved online by a simple matrix-vector multiplication and a vector addition. In contrast to recalculating the solution, this is numerically much more efficient. In [Büs02], an error analysis as a function of the deviation Δp is presented for the differences between the approximations by sensitivity-based updates and the exact solution.

Theorem 2.2.15 (Error Analysis, [Büs02])
Let the assumptions of the Sensitivity Theorem 2.2.14 hold and let the functions F and g be three times continuously differentiable with respect to their arguments. Then there exists a neighborhood $P_0 \subset P$ of p_0, so that for all $\Delta p \coloneqq p - p_0 \in P_0$ the error estimates hold:

$$\begin{aligned} \|x(p) - \tilde{x}(p)\| &= \mathcal{O}(\|\Delta p\|^2), \\ \|F(x(p), p) - F(\tilde{x}(p), p)\| &= \mathcal{O}(\|\Delta p\|^2), \\ \|g^a(\tilde{x}(p), p)\| &= \mathcal{O}(\|\Delta p\|^2), \\ \|\lambda^a(p) - \tilde{\lambda}^a(p)\| &= \mathcal{O}(\|\Delta p\|^2), \\ \|\nabla_x L(\tilde{x}(p), \tilde{\lambda}^a(p), p)\| &= \mathcal{O}(\|\Delta p\|^2). \end{aligned}$$

The linear approximation of the solution can be expected due to the linear Taylor approximation. In the case of unconstrained NLP problems, an additional order in the objective function can be obtained, [Büs02]. An analytical statement about the size of the set P_0 from Theorem 2.2.14 cannot be made in general, so this has to be investigated numerically in advance.

2.2.4. The Optimization Software WORHP

In the context of the presented work, different nonlinear optimization problems of the form (2.2) appear several times. They arise during the formulation of the parameter identification problems and the solution of optimal control tasks with direct methods. For the solution of such problems, the previously presented SQP method is used. An efficient implementation is given with the optimization software Worhp, see Büskens and Wassel [BW13]. It implements the SQP method so that the quadratic subproblem is solved with an interior point method.

The solver's name is an acronym for "We Optimize Really Huge Problems" and indicates that it was developed to solve very large problems, i.e., those with many optimization variables and constraints. The numerical solution of such large problems generally requires very high computational and memory resources, since especially the gradients and the Hessian matrix of the Lagrangian function are needed. For specific problem classes, sparsely populated structures often appear in the derivative information, i.e., many of the entries in the Jacobian and Hessian matrix are zero. Worhp takes advantage of the fact that only the entries that do not disappear must be stored, see Wassel [Was13]. By specifying the structure of the non-zero entries within the derivative matrices, the computational effort and, thus, the computation time can be reduced. Examples of large and sparse optimization problems are the discretized dynamical parameter identification problems, which appear in Subsection 2.4.3, and general optimal control problems, as in Section 4.2.

The software was developed in 2008 under the direction of the University of Bremen in cooperation with the University of the German Federal Armed Forces Munich in projects of the German Aerospace Center DLR and the European Space Agency ESA. Since then, new extensions have continuously enhanced and supplemented the implementations. In 2012 the transcription tool TransWorhp was added, see Subsection 4.2.3 and Knauer and Büskens [KB12]. Moreover, the possibility of parametric sensitivity analysis was added with the WorhpZen module, see Kuhlmann et al. [KGB18]. This makes sensitivity information accessible that is already obtained during the SQP process, which includes sensitivity derivatives of the optimal solution, the constraints, the objective function, and second-order derivatives of the objective function within the NLP. Thus, post-optimality analyses can be performed. We use this information to apply parametric real-time corrections, see Subsection 3.3.2.

2.3. The Nonlinear Dynamical Parameter Identification Problem

In the following, we return to the question of how parameters in a dynamical model can be determined to optimally represent a given real system with respect to a specific objective. Furthermore, we will deal with the numerical solution of the resulting problems and discuss two different solution approaches. First, we need the definition of a parameter-dependent dynamical model.

Definition 2.3.1 (Parametric Dynamical Model)
For a time interval $[t_0, t_f] \subset \mathbb{R}$ let $f\colon D \to \mathbb{R}^{n_x}$ be a twice continuously differentiable function with domain $D \subseteq \mathbb{R}^{n_x} \times \mathbb{R}^{n_p} \times [t_0, t_f]$. The differential equation

$$\dot{x}(t) = f(x(t), p, t)$$

is then called **parametric dynamical model**.

In the following, the term *dynamics of the system* is used synonymously to *dynamical model*. Given an initial condition $x(t_0) = x_0$, the dynamical model in Definition 2.3.1 has exactly one solution, see Theorem 2.4.2. The solution of the system at time $t \in [t_0, t_f]$ is denoted by $x(t; x_0, p)$ to express that the solution implicitly depends on the initial state and the parameters. For the sake of simplicity, the explicit dependence of each state x on p is not written $x(t) = x(p, t)$.

The problem of parameter identification can now be formulated as an infinite-dimensional nonlinear optimization problem with the system dynamics in the constraints.

Definition 2.3.2 (Nonlinear Dynamical Parameter Identification Problem)
Let the function f be a parametric dynamical model from Definition 2.3.1 and $g(x, p)\colon \mathbb{R}^{n_x} \times \mathbb{R}^{n_p} \to \mathbb{R}^m$ be a function, that is twice continuously differentiable with respect to both arguments. Let $t_i \in [t_0, t_f]$ for $i = 0, \dots, M$ be discrete time points with $t_0 < t_1 < \dots < t_M$ and $y_i \in \mathbb{R}^{n_x}$ corresponding measurements. Then $y = (y_0, \dots, y_M)^T$ is called a set of measurement data.

The optimization problem

$$\begin{aligned} \min_{x_0, p} \quad F(x_0, p) &\coloneqq \frac{1}{2M} \sum_{i=0}^{M} \|x(t_i; x_0, p) - y_i\|_2^2, \\ \text{w.r.t. } \dot{x}(t) &= f(x(t), p, t), \text{ for } t \in [t_0, t_f], \\ x(t_0) &= x_0, \\ g(x(t), p) &\leq 0, \end{aligned} \tag{2.12}$$

is called **nonlinear dynamical parameter identification problem (NDPIP)**.

Here, the term $x(t_i; x_0, p)$ denotes the solution of the initial value problem in the constraints for fixed p and x_0 evaluated at time t_i. The basic idea in (2.12) is to minimize the distances between the measured values and the model output at the respective points in time. Instead of directly minimizing the residuals $\|x(t_i) - y_i\|_2$, a least-squares approach is chosen using the square of the errors. This provides the numerical advantage that the objective function is everywhere differentiable. It can be advantageous to weigh individual summands of the objective function. Then, the objective function is said to be of *weighted least-squares* type:

$$F(x_0, p) \coloneqq \frac{1}{2M} \sum_{i=0}^{M} w_i \|x(t_i; x_0, p) - y_i\|_2^2.$$

The weights $w_i \in \mathbb{R}$ for $i = 0, \ldots, M$ can be particularly interesting if, for example, the variance of the observations is known, see Subsection 5.3.1 on maximum likelihood estimation. If additional information about the parameters is given, it may be useful to restrict the search space. Therefore, any nonlinear constraints on the searched parameters are allowed as general equality or inequality constraints with a function $g(\cdot, \cdot)$. This includes simple *box constraints*, where upper and lower bounds are given for the parameters to be optimized., i.e., $p_l \leq p \leq p_u$ with p_l, $p_u \in \mathbb{R}^{n_p}$. This can be important if the parameters have physical meanings, e.g., they cannot be negative.

It should be noted that in the context of this thesis, only ordinary differential equations are considered for the dynamical system. For example, an extension to partial differential equations is described in Schittkowski [Sch02] and Wernsing [Wer18]. Furthermore, it is noted that in addition to the searched parameters, the initial system states x_0 are optimized since it is assumed that these are only available as error-prone measurements. Due to the dynamics in the constraints, the problem (2.12) shows analogies to optimal control problems. We will look at these problems later in the context of nonlinear model predictive control in Chapter 4. In optimal control problems, the states and controls are infinite-dimensional optimization variables that have to be approximated in time. In the same way, the system states $x(t; p)$ are to be optimized in (2.12), which are the solutions of the dynamical system in the constraints. The system model can also include controls, which are not yet explicitly mentioned in this chapter because they are fixed within the parameter identification process and enter the optimization problem as input data. To obtain a problem in the standard form (2.2), methods from the theory of optimal control such as *full discretization* are applied to the parameter identification problem.

2.4. Numerical Solution of NDPIP

We distinguish two solution approaches to solve nonlinear optimization problems with a dynamical system as a constraint. The general procedure of the two methods is shown schematically in Figure 2.2. The classical approach to solving the (2.12) is shown on the

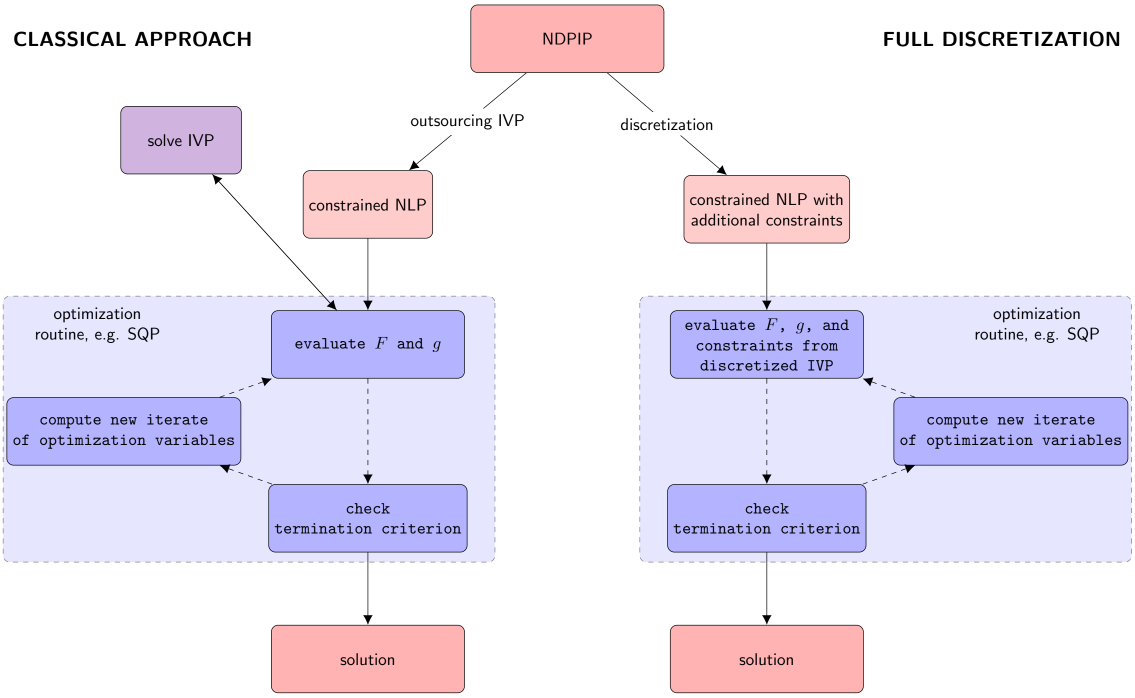
CLASSICAL APPROACH
NDPIP
FULL DISCRETIZATION
outsourcing IVP
discretization
solve IVP
constrained NLP
constrained NLP with additional constraints
optimization routine, e.g. SQP
evaluate F and g
compute new iterate of optimization variables
check termination criterion
evaluate F, g, and constraints from discretized IVP
optimization routine, e.g. SQP
compute new iterate of optimization variables
check termination criterion
solution
solution

Figure 2.2.: *Comparison of numerical solution approaches for nonlinear dynamical parameter identification problems. Classical approach with external IVP solver on the left and full discretization on the right.*

left side. The strategy is to outsource the solution of the so-called *initial value problem (IVP)* in the constraints. This is the problem of finding the solution to the differential equations with the given initial values in the constraints of (2.12). So, whenever the optimization procedure needs to evaluate the state $x(t)$ at a time t, e.g., when evaluating the objective function, the IVP is solved by an external integration procedure. This external solver can be, for example, a *shooting method.*

On the right side, the full discretization method is illustrated. It embeds the numerical solution of the IVP into the optimization problem. For this purpose, additional equality constraints are introduced within the NLP. In the next subsections, the definitions for former terms are given, and the different solution approaches are formulated in more detail. Further, we discuss their advantages and disadvantages.

2.4.1. Initial Value Problems

The differential equations with initial values in the constraints of (2.12) are called *initial value problems.* To solve them numerically, we first need some theoretical background.

Definition 2.4.1 (Parametric Initial Value Problem)
Let $f\colon D \to \mathbb{R}^{n_x}$ be a parameter-dependent dynamical model according to Definition 2.3.1, i.e., f is a twice continuously differentiable function with domain $D \subseteq \mathbb{R}^{n_x} \times \mathbb{R}^{n_p} \times [t_0, t_f]$. Further, let $(x_0(p), t_0) \in \mathbb{R}^{n_x} \times [t_0, t_f]$ be a given initial condition. The problem

$$\begin{aligned} \dot{x}(t) &= f(x(t), p, t), \\ x(t_0) &= x_0(p), \end{aligned} \tag{2.13}$$

is called **parametric initial value problem (IVP(p))** for the initial value $x_0(p)$.

We neglect again to directly write out the dependence of x on p. Under the given conditions, the following theorem ensures the existence of a continuously differentiable unique solution of (2.13).

Theorem 2.4.2 (Picard-Lindelöf)
Given is (2.13) from Definition 2.4.1 with fixed $p \in \mathbb{R}^{n_p}$. Let the system function f be continuous on D in t and Lipschitz-continuous with respect to the state x, i.e. there exists a positive Lipschitz constant L with $\|f(x_1, p, t) - f(x_2, p, t)\| \leq L\|x_1 - x_2\|$ for all $(x_1, p, t), (x_2, p, t) \in D$. Then for each initial state $x_0(p) \in \mathbb{R}^{n_x}$ and initial time $t_0 \in [t_0, t_f]$ there exists exactly one solution $x(t; x_0, p, t_0)$ of (2.13). The solution is defined for all times t in an open maximum existence interval $I := I(x_0, p, t_0) \subset \mathbb{R}$ with $t_0 \in I$.

Proof. A proof of the theorem is shown in Grüne and Junge [GJ16] or Heuser [Heu95]. □

Due to the continuous differentiability of f on a compact support, the Lipschitz condition is fulfilled in Theorem 2.4.2, and a unique solution of the initial value problem (2.13) is ensured. In his book, Heuser [Heu95] provides different examples for the analytical solution of (linear) differential equations. In general, however, solving more complex nonlinear differential equation systems in a closed form is impossible. This makes numerical methods that can provide sufficiently accurate approximations all the more important. Therefore, numerical methods are necessary, which can at least provide sufficiently accurate approximated solutions.

A standard approach to numerically solve the initial value problem (2.13) is to perform a suitable discretization of the time interval and approximate the exact solution at the grid points. Time discretization reduces the continuous system to a finite, discrete system. This can then be solved numerically in a simplified form. The idea of discretization is to replace a region with a grid. Instead of a solution that is defined over the whole region, the discretized solution is obtained only at the grid points.

Definition 2.4.3 ((Equidistant) Grid)
Let $G := \{\tau_i \mid i = 0, \ldots, N\}$ be a **discretization**, or **grid**, of the time interval $[t_0, t_f] \subset \mathbb{R}$, i.e. $\tau_i \in [t_0, t_f] \subset \mathbb{R}$ for $i = 0, \ldots, N$ with $N \in \mathbb{N}$ and $\tau_i < \tau_{i+1}$, for all $i = 0, \ldots, N-1$ and $\tau_0 := t_0$ and $\tau_N := t_f$. We call G_h an **equidistant discretization** or **equidistant grid** respectively with step size $h \in \mathbb{R}$, if additionally $\tau_{i+1} - \tau_i = h$ is valid for all $i = 0, \ldots, N-1$.

We search for approximations $\tilde{x}_i \approx x(\tau_i)$ at discrete time points τ_i on the equidistant grid G_h. All discretized states are summarized in

$$\tilde{x} := (\tilde{x}_0^T, \ldots, \tilde{x}_N^T)^T \in \mathbb{R}^{n_x \cdot (N+1)}.$$

One of the most familiar methods for solving initial value problems is *explicit Euler's method*. It is a method of order one and is based on a Taylor approximation of $x(t)$. For an equidistant discretization G_h of the time interval $[t_0, t_f]$ with the fixed step size $h \in \mathbb{R}$ an approximation of the solution of (2.13) can be obtained by the recursive formula

$$\tilde{x}_{i+1} := \tilde{x}_i + hf(\tilde{x}_i, p, \tau_i), \text{ for } i = 0, \ldots, N-1, \text{ and } \tilde{x}_0 := x_0.$$

This procedure motivates the use of *general one-step methods* of the form

$$\tilde{x}_{i+1} := \tilde{x}_i + h\Phi(\tilde{x}_i, p; h; f), \text{ for } i = 0, \ldots, N-1, \text{ and } \tilde{x}_0 := x_0. \tag{2.14}$$

Here, the *incremental function* $\Phi(\tilde{x}_i, p; h; f)$ serves as a placeholder to describe with (2.14) any one-step method. It should be chosen so that the approximated solution matches the exact solution as closely as possible. For the Euler method it applies that $\Phi(\tilde{x}_i, p; h; f) := f(\tilde{x}_i, p, \tau_i)$. With this idea, other methods of higher order can be developed to solve (2.13), such as the class of *(s-staged) explicit Runge-Kutta methods*.

Definition 2.4.4 ((s-staged) Explicit Runge-Kutta Method)
For an $s \in \mathbb{N}$ with constant coefficients $a_{ij}, b_i, c_i \in \mathbb{R}$, $k_i \in \mathbb{R}^{n_x}$ for $i = 1, \ldots, s$ and $j = 1, \ldots, s-1$, we call the one-step method (2.14) specified by

$$\begin{aligned} \Phi(\tilde{x}, p; h; f) &:= \sum_{i=1}^{s} b_i \cdot k_i, \\ \text{with } k_i &:= f\Big(\tilde{x} + h \sum_{j=1}^{i-1} a_{ij} \cdot k_j, p, \tau_i + c_i h\Big), \end{aligned} \tag{2.15}$$

an **(s-staged) explicit Runge-Kutta method**.

In this work, the preferred methods are the *trapezoidal method*

$$\tilde{x}_{i+1} := \tilde{x}_i + \frac{h}{2}\big(f(\tilde{x}_i, p, \tau_i) + f(\tilde{x}_{i+1}, p, \tau_{i+1})\big), \tag{2.16}$$

and the *Hermite-Simpson method*

$$\begin{aligned}\tilde{x}_{i+\frac{1}{2}} &:= \frac{1}{2}(\tilde{x}_{i+1} + \tilde{x}_i) + \frac{h}{8}\big(f(\tilde{x}_i, p, \tau_i) - f(\tilde{x}_{i+1}, p, \tau_{i+1})\big),\\ \tilde{x}_{i+1} &:= \tilde{x}_i + \frac{h}{6}\big(f(\tilde{x}_i, p, \tau_i) + f(\tilde{x}_{i+\frac{1}{2}}, p, \tau_{i+\frac{1}{2}}) + f(\tilde{x}_{i+1}, p, \tau_{i+1})\big),\end{aligned} \tag{2.17}$$

with an additional function evaluation at the intermediate point $\tau_{i+\frac{1}{2}} = \frac{1}{2}(\tau_i + \tau_{i+1})$. These two implicit methods have order of $\mathcal{O}(h^2)$ and $\mathcal{O}(h^4)$, respectively. Since the computational effort of these methods is proportional to the number of single steps, one will try to choose the step size h as large as possible while keeping the discretization error as small as possible. Therefore, one-step methods are often extended by *step size strategies*. More detailed information on step size strategies and the construction of general higher-order methods is provided by Bulirsch and Stör [BS05].

2.4.2. Classical Approach

The classical solution approach of (2.12) is to combines an NLP solver with an IVP solver. The procedure is depicted on the left side in Figure 2.2. Whenever the optimizer has to evaluate the state $x(t;p)$ in the objective function at a time t, the IVP is solved by an external integration method. Common approaches are the *single-shooting* or *multiple-shooting method*, Schittkowski [Sch02].

Single Shooting

When applying a single-shooting method to solve the IVP, the given dynamics are integrated forward in time to provide the states at a specific time point. This can be achieved by any one-step method (2.14). The resulting NLP reduces to:

$$\begin{aligned}\min_{\tilde{x}_0, p} \quad & F(\tilde{x}, p) := \frac{1}{2M}\sum_{i=0}^{M} \|\tilde{x}(t_i; p, \tilde{x}_0) - y_i\|_2^2,\\ \text{w.r.t. } & g(\tilde{x}(t_i), p) \le 0, \qquad \text{for } i = 0, \dots, N.\end{aligned}$$

Since for fixed parameters p and given initial values $\tilde{x}_0$, the solution of the IVP is uniquely determined due to Theorem 2.4.2, only these values are defined as optimization variables. Thus $\tilde{x}(t_0;p) = \tilde{x}_0$ and $\tilde{x}(t_i;p)$ is the solution of IVP at time t_i, $1 \le i \le M$. These states at discrete time points are only needed internally to evaluate the cost function. In this manner, only the searched parameters are optimized and the problem remains small compared to the full discretization approach. Due to the process of sequential integration and optimization, the dynamics of the underlying system are fulfilled in every iteration step to an accuracy of a numerical discretization error, Flaßkamp [Fla13]. One disadvantage is that the objective function is highly nonlinear due to the numerical integration and can, therefore, have many local minima, see Schittkowski [Sch02]. Furthermore, the method is relatively sensitive to disturbances in the initial values. This is especially relevant when considering data over a longer time horizon. To compensate for this, a good initial guess for the parameters is needed.

Multiple Shooting

The multiple-shooting approach is proposed, for example, in [VTK04; Boc+13] as a way to overcome the lack of robustness of the single-shooting approach. For this method, so-called *shooting nodes* $\xi_j \in \mathbb{R}$, for $j = 1, \ldots, k$ and $k \in \mathbb{N}$, are introduced at which the integration is restarted. The time interval $[t_0, t_f]$ is thus divided into subintervals. The IVP is solved on the subintervals $[\xi_j, \xi_{j+1}]$ by an external integration procedure as in the single-shooting approach. For the additional initial values $\tilde{x}(\xi_j)$, new optimization variables $\tilde{x}_{\xi_j}$ are introduced for $j = 1, \ldots, k$. In order to obtain a continuous solution of the IVP on the whole time interval at the end of the optimization, *continuity conditions* are implemented at the shooting nodes. They ensure that the endpoints of each subinterval coincide with the initial values of the next interval. This is realized with additional nonlinear equality constraints of the form

$$\tilde{x}_{\xi_{j+1}} - \tilde{x}(\xi_{j+1}; p, \tilde{x}_{\xi_j}) = 0, \quad \text{for } j = 1, \ldots, k-1.$$

The resulting optimization problem is

$$\begin{aligned} \min_{\tilde{x}_\xi, p} \quad & F(\tilde{x}_\xi, p) := \frac{1}{2M} \sum_{i=0}^{M} \|\tilde{x}(t_i; p, \tilde{x}_{\xi_{\kappa(i)}}) - y_i\|_2^2, \\ \text{w.r.t.} \quad & \tilde{x}_{\xi_{j+1}} - \tilde{x}(\xi_{j+1}; p, \tilde{x}_{\xi_j}) = 0, \qquad \text{for } j = 1, \ldots, k-1, \\ & g(\tilde{x}(t_i), p) \leq 0, \qquad\qquad\qquad\quad \text{for } i = 0, \ldots, M, \end{aligned}$$

with $\tilde{x}_\xi = (\tilde{x}_{\xi_1}, \ldots, \tilde{x}_{\xi_k})$. Here, the expression $\tilde{x}(t_i; p, \tilde{x}_{\xi_{\kappa(i)}})$ describes the solution $\tilde{x}$ obtained by integrating the dynamics from the initial value $x_{\xi_{\kappa(i)}}$ to time t_i. In this context, $\kappa(i)$ is the index of the nearest smaller shooting node to the time stamp of the current measurement y_i, i.e. $\kappa(i) := \max_{1 \leq j \leq k}\{j \colon \xi_j \leq t_i\}$, cf. Schäfer et al. [Sch+18]. The number of optimization variables in the NLP increases by the number of introduced shooting nodes. If only one node is used, we again get the well-known single-shooting method since the ODE is integrated over the entire time interval. If the number of shooting nodes coincides with the number of time discretization steps, we get the *full discretization method*, which we present in the following section.

2.4.3. Full Discretization

In the case of full discretization, the constraints of the given (2.13) are discretized before optimization. The numerical integration methods explained in the previous Subsection 2.4.1 are used for this. In this way, the states at each discretization point are transformed into optimization variables, and the dynamical system is turned into individual constraints. For the sake of simplicity, we will discretize the dynamics on the time grid of measurements in the following representation. Thus, the number of discretization grid points equals the number of measurements, $N = M$. This is not necessarily the case. Since the number of discretization points strongly influences the size of the resulting optimization problem and, thus, the required computational effort, reducing the number of grid points may be of interest. In particular, if there is a large amount of closely sampled data, a larger discretization step size can be selected. If only a few or non-equidistant measurements are available, it may be necessary to interpolate them at the discretization points of the dynamics. This can be done, for example, with linear interpolation. Note that in this case, an interpolation error is introduced. If we choose as many discretization points as measurements, we obtain the following problem formulation:

Definition 2.4.5 (Discretized Nonlinear Dynamical Parameter Identification Problem)
Let f be the parametric dynamical model from Definition 2.3.1. Let $y = (y_0, \dots, y_M)^T$ be given measurements $y_i \in \mathbb{R}^{n_x}$ for $i = 0, \dots, M$ at discrete time points $t_i \in \mathbb{R}$ with $t_0 < t_1 < \cdots < t_M$. Furthermore, let $\Phi(\tilde{x}_j, p; h; f)$ be any explicit or implicit one-step method with step size h. The optimization problem

$$\begin{aligned}
\min_{\tilde{x}_0, p} \quad & F(\tilde{x}_0, p) := \frac{1}{2M} \sum_{i=1}^{M} \|\tilde{x}(t_i; p, \tilde{x}_0) - y_i\|_2^2, \\
\text{w.r.t.} \quad & \tilde{x}_{j+1} = \tilde{x}_j + h \cdot \Phi(\tilde{x}_j, \tilde{x}_{j+1}, p; h; f), \text{ for } j = 0, \dots, M-1, \\
& \tilde{x}(t_0) = \tilde{x}_0, \\
& g(\tilde{x}(t_j), p) \leq 0, \text{ for } j = 0, \dots, M,
\end{aligned} \tag{2.18}$$

with $\tilde{x} = (\tilde{x}_0, \dots, \tilde{x}_{N=M})$ is called **discretized nonlinear dynamical parameter identification problem (dNDPIP)**.

By including the initial states as optimization variables, measurement inaccuracies in the initial values are taken into account. Thus, the dynamics of the system can be adapted to the measurement data in the best possible way. The discretization introduces new optimization variables so that the resulting optimization problem (2.18) has a high number of constraints. As a result, it appears much more complex than the original (2.12). However, often the derivatives of the objective function, the constraints, the ODE system, and the Hessian do not depend on all the optimization variables, leading to a sparse structure of these matrices. This also applies to optimization problems resulting from discretized dynamic problems with a one-step method, [Ech14]. The software library WORHP uses the SQP method outlined in Subsection 2.2.1 and can exploit sparse structures in the derivative matrices, as explained in Section 2.2.4. Thus, such problems can be solved with high efficiency.

The main advantage of the full discretization approach is that the initial value problem formulated in the constraints of (2.12) does not have to be solved in every iteration step of the optimizer to evaluate the objective function. Instead, the IVP is solved implicitly during the iteration process. Furthermore, the feasibility of the optimization variables has to be guaranteed only at the end of the iteration process. Therefore, the dynamic equations have to be valid only at the end of the optimization when all constraints are fulfilled with a certain accuracy. In particular, this means that a feasible initial guess p_0 for the parameters is unnecessary, as is the case with classical approaches. Here, *feasible* means that the initial value problem from Definition 2.4.1 has a solution at an initial value $p_0 \in \mathbb{R}^{n_p}$. If such a feasible initial guess cannot be provided, this usually has the effect of divergence of the numerical method. For the full discretization approach, such initial values do not pose a problem, even though the choice of the initial guess may influence the result of the optimization, compare Subsection 2.4.4.

2.4.4. Special Aspects of the Numerical Solution Process

Here, we discuss some special aspects of solving nonlinear dynamical parameter identification problems by numerical optimization methods. The consideration of these additional aspects can contribute significantly to a better and more efficient numerical solution to the identification problem.

Dependence on Initial Guess

The presence of many local minima is a typical problem in nonlinear dynamical parameter identification problems. In general, the choice of the initial guess highly affects which minimum is found by a nonlinear optimization algorithm. Echim [Ech14] motivates to use the full discretization approach. There, numerical examples show that it is more robust to the initial guess than the classical approach with an external IVP solver, especially when many local minima occur. The convergence of the local SQP method depends on the choice of initial estimates for the parameters since a specific convergence rate can only be shown for a neighborhood around the optimal solution, see Subsection 2.2.1. In general, however, the size of this neighborhood is unknown. Therefore, special attention should be paid to an excellent choice of initial guesses. Within the transcription tool TransWORHP we also provide initial estimates for the discretized solution trajectory $\tilde{x}$, see Subsection 4.2.3. In this work, the measured data at the respective discretization points or a value interpolated from neighboring measured points are used for this purpose. Thus, a very good initial guess for the values can be provided.

Furthermore, it may be beneficial to first optimize problems with a small number of discretization points and then use the solution as a ‘good’ initial guess for a finer resolution optimization problem. We apply this approach, for example, in the offline computation of the feedback gain for the linear quadratic regulator in Subsection 5.4.4.

Scaling of System States

Let $x(t) \in \mathbb{R}^{n_x}$ be the time-dependent state vector of a dynamical system as described in Definition 2.3.1:

$$\dot{x}(t) = f(x(t), p, t),$$

with dynamics $f \colon \mathbb{R}^{n_x} \times \mathbb{R}^{n_p} \times \mathbb{R} \to \mathbb{R}^{n_x}$. The objective function for parameter identification problems consists of the squared errors of the distances of the system states $x(t)$ to the measurement values $y(t) \in \mathbb{R}^{n_x}$. If the domains of the different system states are of different orders of magnitude, the error terms have unbalanced influences on the objective function. The effect of certain variables on the minimization of the objective function would thus be larger than that of others. Furthermore, large size imbalances can also lead to ill-conditioned matrices and thus to numerical problems within the optimization, [Büs98]. Measurements and system states are scaled before solving the optimization problem to avoid such problems. In the context of this thesis, a linear scaling approach is chosen to adjust the quantities:

$$\begin{aligned} \bar{x}_i(t) &= c_i x_i(t), \quad t \in [t_0, t_f] \subset \mathbb{R}, \quad i = 1, \dots, n_x, \\ \bar{y}_i(t) &= c_i y_i(t), \quad t \in [t_0, t_f] \subset \mathbb{R}, \quad i = 1, \dots, n_x, \end{aligned} \tag{2.19}$$

for a fixed $c \in \mathbb{R}^{n_x}$, where $\bar{x}(t) \in [-1, 1]^{n_x}$ is the scaled version of $x(t)$ to the interval $[-1, 1]^{n_x}$ and $\bar{y}(t) \in [-1, 1]^{n_x}$ of $y(t)$, respectively. Note that the scaling is applied to the vector-valued variables and dynamics component-wise.

Given some measured data, the scaling factor is computed from the largest absolute value to

$$c_i = \frac{1}{\max\limits_t \mid y_i(t) \mid}, \quad i = 1, \dots, n_x.$$

In the case that $\max\limits_t y_i(t) = 0$ holds for some $i \in \{1, \dots, n_x\}$, we set $c_i = 1$. The scaling of the states also needs to be considered in the system dynamics. Let $\bar{f} \colon [-1, 1]^{n_x} \times \mathbb{R}^{n_p} \times \mathbb{R} \to \mathbb{R}^{n_x}$ be the scaled dynamics and $\bar{x}(t)$ be the solution of the differential equation

$$\dot{\bar{x}}(t) = \bar{f}(\bar{x}(t), p, t), \quad t \in \mathbb{R}.$$

Then the differentiation of the first equation in (2.19) with respect to the time t yields

$$\dot{\bar{x}}(t) = c\dot{x}(t).$$

Thus we get

$$\dot{\bar{x}}(t) = cf(x(t), p, t).$$

The optimization results can then be easily scaled back using the scaling factors and rearranging the equations (2.19) to $x(t)$.

Different Norms for the Objective Function

The standard choice for the norm in the objective function is the Euclidean or 2-norm, i.e., the sum of the squares of the errors. However, sometimes, using a different norm to estimate parameters can be desirable. For example, one can use the L_∞-norm to minimize the maximum of the absolute values of the distance between the model and the measured data. Another alternative to fit the model to the measured data is the L_1-norm. Here, the sum of the absolute values of the residuals is minimized. Both cases lead to non-differentiable objective functions. This requires a transformation into a smooth nonlinear program, which can then be solved with a standard method, e.g., an SQP algorithm [Sch02].

We have decided to use the 2-norm here since it has certain advantages. In particular, this way generates a quadratic objective function in the full discretization approach. This is numerically advantageous, although we would like to emphasize that the approaches used in this thesis are not committed to this choice.

Practical Identifiability and Persistent Excitation

Another question that arises with respect to parameter identification concerns the identifiability of the parameters. On the one hand, the so-called *structural identifiability* can be analyzed. The focus here is on whether the used model is well-defined. That is, the dynamical system under consideration can be solved uniquely and thus, the parameters can be identified, [BÅ70]. We will not discuss this aspect further here and implicitly assume it since we consider the model in our context to be given and not free to choose.

The identifiability of parameters in practical applications is very interesting for this work. This is known as *practical identifiability*, which refers to whether parameters in a dynamical model can be effectively identified in concrete applications. The aim is to determine whether the available model input and output information is sufficient for parameter identification. For example, identifying the parameter that determines the steering ratio between the steering wheel and tire angles is crucial in a car model. However, if arbitrary accelerations, but only constant zero angular velocity, is provided, the parameter is practically not identifiable.

In offline (system) identification, the term *persistent excitation* is often encountered. This involves analyzing how much the input signal excites the system so that model outputs can be used to determine system parameters. Aström and Wittenmark [ÅW09] give an excitation condition for linear time-invariant dynamical models. It can be used to determine the order of persistent excitation (PE) of the input signal, which specifies the number of parameters that can be determined from this input. For example, the unit step function is persistently exciting of order one and a random signal would be PE of any degree. Miranda-Colorado and Moreno-Valenzuela [MM17] analyze various input signals on PE to identify a nonlinear servomechanism in closed-loop identification. The system is nonlinear but linear in the parameters. Further, Adetola and Guay [AG07; ADG09] examine conditions for parameter convergence that can be checked for some specific classes of nonlinear systems.

The field of *optimal experimental design* (OED) deals with the question of how to design experiments for parameter identification. Essentially, it involves strategically searching for inputs that allow accurate identification when combined with the corresponding outputs they produce. Additionally, they also optimize a chosen objective function. An overview of relevant work in OED for dynamical systems is given, for example, in Franceschini and Macchietto [FM08]. Experiments are often optimized for the shortest possible duration or maximum information content. There, the concept of persistent excitation of the system is also fundamental. For this purpose, formulations based on the so-called *Fisher information matrix* (FIM) are often used in the objective functions to be optimized, Bar-Shalom et al. [BLK01]. Thus, for a given parameter estimator, a probability statement can be made about the quality of the obtained parameters under certain assumptions on the model and measurements. In this thesis, the FIM is used in Chapter 5 to make statements about the information content of the used data within the proposed algorithm and to decide if a parameter identification should be made.

Evaluation of the Identification Result

The evaluation of parameter identification results can be approached in different ways. The chosen strategy usually depends on the application. For example, suppose a high-quality model based on physical assumptions exists for a process to be modeled. In that case, parameter identification often aims to estimate individual unknown or imprecise physical parameters. The result of the identification process $p \in \mathbb{R}^{n_p}$ can then be evaluated by the distance to the true parameter values $p^* \in \mathbb{R}^{n_p}$.

$$\varepsilon_p = \|p^* - p\|_2$$

is used. This metric is particularly valuable for evaluating algorithm performance when simulated data is used and the true parameters are known. If a sequence of parameter estimates $\{p_i\}$, $i = 1, \dots, M$ at $M \in \mathbb{N}$ discrete points in time is available, the *root mean squared error* (RMSE) can be considered as a measure:

$$\text{RMSE}_p := \sqrt{\frac{\sum_{i=1}^{M} (p_i^* - p_i)^2}{M}}.$$

This will also be the used approach in the following chapters. Note, that for $p \in \mathbb{R}^{n_p}$ it applies also that $\text{RMSE}_p \in \mathbb{R}^{n_p}$.

A different approach is required in real-world applications where the true value is unknown. In this case, model validation is performed to evaluate the identified model. This is done by splitting the measurements into two sets. With the so-called *training* *data* $\{y_i\}_T$ for $i \in J \subset \mathbb{N}$, the model is trained on only a subset of the measured data. The remainder is the *validation data* $\{y_i\}_V$ for $i \in \bar{J} \subset \mathbb{N}$. The sets J and $\bar{J}$ are chosen so the intersection is empty, i.e., $J \cap \bar{J} = \emptyset$. Once the identification result is available, the model is tested with the validation data to determine the model quality. For that, the error for the validation data is computed by

$$\varepsilon_V = \frac{1}{\mid \bar{J} \mid} \sum_{i \in \bar{J}} \|x(t_i; p, x_0) - y_i\|_2^2,$$

where $x(t_i; p, x_0)$ is the solution of $\dot{x}(t) = f(t, x(t), p)$, $x(t_0) = x_0$ corresponding to the measurements y_i for $i \in \bar{J}$. This approach is widely used in offline identification and is especially common in machine learning and neural network training.

3. Optimal Linear Feedback Control

This chapter presents the theoretical foundations of an adaptive linear quadratic regulator. The basic definitions of linear and nonlinear control systems and the fundamentals of system theory for linear systems are introduced. Based on this, the *linear quadratic regulator* (LQR) problem and its classical solution via the algebraic Riccati equation are presented. Further, an alternative solution method is demonstrated by solving the LQR problem as an optimization problem. This approach allows the usage of methods of parametric sensitivity analysis in the case of parametric perturbations. Thus, a real-time approximation of the solution of the parametric LQR problem can be applied.

3.1. Fundamentals of Systems Theory

Mathematical systems theory is concerned with finding, formulating, and analyzing theoretical statements about the behavior of dynamical control systems, i.e., time-dependent control systems. These can be formulated in different ways, in continuous-time as well as in discrete-time, such as *transfer functions*, *ordinary*, or *partial differential equations*. This chapter will focus on continuous-time models with ordinary differential equations. An fundamental property of the systems under consideration in the following is the dependence on the control variable u, used to influence the system. Practical examples of controls are the acceleration and steering angle of a car or the thrusters of a rocket.

When considering a real control process from a system's theoretical point of view, it is essential to have a suitable mathematical model available. Next, the definitions of nonlinear and linear control systems are given. Furthermore, we present how to linearize nonlinear systems to make the linear theory also applicable to them.

We formulate the definitions for control systems as parameter-dependent systems since we will refer to them in the following Chapters 4, 5, and 6. For the sake of simplicity, the following theoretical considerations on controllability and stability in the Subsections 3.1.2, 3.1.3, and 3.1.4 are made under the assumption of a fixed parameter. The chapter is based on the statements from Grüne [Grü18] and Büskens [Büs19], where also more detailed explanations of systems theory can be found.

3.1.1. Control Systems

As a basis for the further discussions, we define for a parameter vector $p \in \mathbb{R}^{n_p}$ a linear parametric control system as follows.

Definition 3.1.1 ((Autonomous) Linear Parametric Control System)
Let $A\colon \mathbb{R}^{n_p} \times [t_0, t_f] \to \mathbb{R}^{n_x \times n_x}$, $B\colon \mathbb{R}^{n_p} \times [t_0, t_f] \to \mathbb{R}^{n_x \times n_u}$, $C\colon \mathbb{R}^{n_p} \times [t_0, t_f] \to \mathbb{R}^{n_y \times n_x}$, $D\colon \mathbb{R}^{n_p} \times [t_0, t_f] \to \mathbb{R}^{n_y \times n_u}$, $x\colon [t_0, t_f] \to \mathbb{R}^{n_x}$ and $u\colon [t_0, t_f] \to \mathbb{R}^{n_u}$ be sufficiently smooth matrix- or vector-valued functions. For $t \in [t_0, t_f] \subset \mathbb{R}$, $x_0 \in \mathbb{R}^{n_x}$ and $p \in \mathbb{R}^{n_p}$ a **linear parametric control system** is given by

$$\begin{aligned} \dot{x}(t) &= A(p,t)x(t) + B(p,t)u(t), \\ x(t_0) &= x_0, \\ y(t) &= C(p,t)x(t) + D(p,t)u(t). \end{aligned} \tag{3.1}$$

The vector $x(t)$ and the vector $u(t)$ are called the **state** and the **control** of the system. The vector $y(t)$ represents the **output variables** of the system, where $n_y \leq n_x$ applies. Moreover, t_0 indicates the **initial time** with corresponding **initial state** x_0. If the system matrices in (3.1) do not depend on the time, i.e.,

$$A(p,t) \equiv A(p), \quad B(p,t) \equiv B(p), \quad C(p,t) \equiv C(p), \quad D(p,t) \equiv D(p),$$

the system is called **autonomous** or **time-invariant**.

For $p \in \mathbb{R}^{n_p}$ the linear map $A(p, \cdot)$ is acting on the *state space* and is referred to as the *state transition matrix*, $B(p, \cdot)$ the *input* or *control matrix*, $C(p, \cdot)$ the *output matrix* and $D(p, \cdot)$ the *feedthrough matrix* of the linear parametric control system (3.1). For later use, we define nonlinear parametric control systems:

Definition 3.1.2 ((Autonomous) Nonlinear Parametric Control System)
Let $f\colon \mathcal{D} \to \mathbb{R}^{n_x}$ and $g\colon \mathcal{D} \to \mathbb{R}^{n_y}$ be sufficiently smooth functions with $\mathcal{D} \subset \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_p} \times [t_0, t_f]$ and $x_0 \in \mathbb{R}^{n_x}$. The problem

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t), p, t), \\ x(t_0) &= x_0, \\ y(t) &= g(x(t), u(t), p, t), \end{aligned} \tag{3.2}$$

is called a **parametric nonlinear control system**. The terms for states and controls apply accordingly to Definition 3.1.1. If the state and output equations in (3.2) do not depend on the time, i.e.,

$$f(x(t), u(t), p, t) \equiv f(x(t), u(t), p), \quad g(x(t), u(t), p, t) \equiv g(x(t), u(t), p),$$

the system is called **autonomous** or **time-invariant**.

A time-dependent system can be transformed into an autonomous system by introducing an additional state variable representing the time. If the state of the system remains unchanged without the influence of controls or disturbances, it is at rest. Such a point is called an *equilibrium*.

Definition 3.1.3 (Equilibrium Point)
Assume the autonomous systems from Definition 3.1.1 and 3.1.2 for a fixed $p \in \mathbb{R}^{n_p}$ are given. Then the solution $(\hat{x}, \hat{u}) \in \mathbb{R}^{n_x} \times \mathbb{R}^{n_u}$ of the equation

$$A(p)\hat{x} + B(p)\hat{u} = 0$$

or respectively

$$f(\hat{x}, \hat{u}, p) = 0$$

is called **equilibrium (point)** or **stationary point** for p.

It can be shown, that any equilibrium $(\hat{x}, \hat{u}) \neq (0, 0)$ can be transformed to the stationary point $(\hat{x}, \hat{u}) = (0, 0)$, [Büs19]. Thus, it is assumed in the following, without loss of generality, that the origin $(\hat{x}, \hat{u}) = (0, 0)$ is the desired equilibrium point.

Linear controllers appear in many practical applications, even if the original control system is nonlinear. For this purpose, the nonlinear system equations are linearized at an equilibrium $(\hat{x}, \hat{u})$ of the system. Assume $(\hat{x}, \hat{u}) = (0, 0)$ is an equilibrium for a fixed $p \in \mathbb{R}^{n_p}$. Then the system matrices are given by

$$\begin{aligned} A &= \nabla_x f(\hat{x}, \hat{u}, p), & B &= \nabla_u f(\hat{x}, \hat{u}, p), \\ C &= \nabla_x g(\hat{x}, \hat{u}, p), & D &= \nabla_u g(\hat{x}, \hat{u}, p). \end{aligned}$$

The linearization is based on the approximation of the nonlinear functions $f(\cdot)$ and $g(\cdot)$ by a Taylor series expansion. It is, therefore, sufficiently exact only in a neighborhood around the operating point. The accuracy of the linearization, therefore, depends on the distance of the current state $x(t)$ and the control $u(t)$ to the chosen equilibrium point $(\hat{x}, \hat{u})$. In the following, the considerations are restricted to autonomous linear control systems. This restriction to linear controlled systems facilitates the following advanced theoretical investigation of the dynamical behavior of the systems. For simplicity, we will neglect the parameter dependency in the following theoretical considerations of this chapter. However, all statements remain valid for a fixed $p \in \mathbb{R}^{n_p}$.

3.1.2. Controllability

Consider a linear control system with an initial state $x(t_0) = x_0 \in \mathbb{R}^{n_x}$ of the form

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) + Du(t). \end{aligned} \tag{3.3}$$

A frequently occurring question in the analysis of a system described by (3.3) is controllability, i.e., the possibility to influence the states by manipulating the inputs of the system. This is examined by searching for points $x_0, x_1 \in \mathbb{R}^{n_x}$ and a time t_1 for which a control function $u\colon [t_0, t_f] \to \mathbb{R}^{n_u}$ can be found, so that $x(t_1; x_0, u) = x_1$ holds. In this case, we denote the solution trajectory of the dynamical system (3.3) with initial state x_0 and given control u as $x(\cdot; x_0, u)$. In other words, we are searching for points, that could be connected by a solution trajectory.

Definition 3.1.4 (Controllable/Reachable Point)
Consider the linear control system (3.3).

(i) A state $x_0 \in \mathbb{R}^{n_x}$ is called **controllable** to a state $x_1 \in \mathbb{R}^{n_x}$ at time $t_1 > 0$, if and only if there exists a control function $u \in \mathcal{U}([t_0, t_f], \mathbb{R}^{n_u})$ with $x_1 = x(t_1; x_0, u)$.[1]

(ii) The point x_1 is then called **reachable** from x_0 at time t_1.

If any point $x_0 \in \mathbb{R}^{n_x}$ is controllable to any other state $x_1 \in \mathbb{R}^{n_x}$, then we call the underlying system (3.3) controllable. By a translation the case of the controllability of any $x_0 \in \mathbb{R}^{n_x}$ to any state $x_1 \in \mathbb{R}^{n_x}$ can be reduced to the controllability of $\bar{x}_0 = 0$ to any state $\bar{x}_1 = x_1 - x(t_1; x_0, 0) \in \mathbb{R}^{n_x}$. For the controllability analysis, this characterization also motivates to consider the set $\mathcal{R}(t)$ of states that can be reached at time t with an appropriate control u starting from the origin $x_0 = 0$.

Definition 3.1.5 (Controllable/Reachable Set)
Consider the linear control system (3.3).

(i) The **reachable set** of $x_0 = 0$ at time $t \geq 0$ is given by

$$\mathcal{R}(t) = \{x(t) \in \mathbb{R}^{n_x} | \text{ there exists } u\colon [t_0, t_f] \to \mathbb{R}^{n_u} \text{ such that } x(t) = x(t; 0, u)\}.$$

(ii) The **controllable set** of $x_1 = 0$ at time $t \geq 0$ is given by

$$\mathcal{C}(t) = \{x_0 \in \mathbb{R}^{n_x} | \text{ there exists } u\colon [t_0, t_f] \to \mathbb{R}^{n_u} \text{ with } x(t; x_0, u) = 0\}.$$

For the continuous formulation of system (3.3) the relation $\mathcal{R}(t) = \mathcal{C}(t)$ applies for all $t > 0$, cf. [Grü18]. If the system (3.3) is controllable, it is $\mathcal{R}(t) = \mathbb{R}^{n_x}$. Note that the controllability of system (3.3) is independent of the matrices C and D. This is why, instead of saying that the system (A, B, C, D) is controllable, it is often said that the pair (A, B) is controllable. Now, we can formulate a constructive tool commonly used to prove controllability in practice.

[1] Here $\mathcal{U}(D, I)$ denotes a function space that maps from D to I. When considering controls we often utilize the space of piecewise continuous functions from $[t_0, t_f]$ to $\mathbb{R}^{n_u}$, denoted by $C^{0,p}([t_0, t_f], \mathbb{R}^{n_u})$.

Theorem 3.1.6 (Controllability Criterion of Kalman, [Grü18])
Consider a linear control system of the form (3.3) with state space dimension n_x. Then the following statements are equivalent.

(i) (A, B) is controllable.
(ii) The controllability matrix of (A, B) has full rank, i.e.,

$$rank\left(\begin{bmatrix} B & AB & \dots & A^{n-1}B \end{bmatrix}\right) = n_x.$$

The size of the controllability matrix can grow significantly, and the computation of the rank often leads to numerical difficulties. The *Hautus criterion* provides an alternative to check the controllability of linear systems without the need to compute the controllability matrix. For that all eigenvalues λ_i and corresponding eigenvectors $v_i \in \mathbb{R}^{n_x}$, $i = 1, \dots, n_x$ of the matrix A are computed and it is tested whether for all $i = 1, \dots, n_x$ it holds that $v_i^T B = 0$, see Grüne [Grü18].

3.1.3. Observability

In many real applications, it is impossible to measure a system's complete state vector $x(t)$ to compute the control $u(t)$. In most cases, it can only be assumed that certain values $y(t) \in \mathbb{R}^{n_y}$ are known. Particularly, the dimension of the output can be smaller than the number of system states, i.e., $n_y \leq n_x$. In practice, it may be necessary to reconstruct the states from the measured output variables with the help of the system dynamics. The concept of state *observability* specifies whether this possibility exists at all time points.

Definition 3.1.7 (Observability)
Consider the linear control system (3.3).

(i) A state $x \in \mathbb{R}^{n_x}$ is called **observable** if the initial state x_0 can be uniquely determined from the known input function $u(t)$ and the output $y(t)$ on a finite time interval $[t_0, t]$.
(ii) The linear dynamical system (3.3) is called fully observable if every state $x \in \mathbb{R}^{n_x}$ is observable.

We formulate a criterion for linear autonomous systems of the form

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t). \end{aligned} \tag{3.4}$$

The following criterion can verify the observability of such a system.

Theorem 3.1.8 (Observability Criterion of Kalman, [Grü18])
Consider a linear control system of the form (3.4) with state space dimension n_x. Then the system (A, B, C) is fully observable if and only if the observability matrix has full rank, i.e.,

$$rank \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{pmatrix} = n_x.$$

It is said that (A, C) is observable if the system (A, B, C) is observable. Furthermore, the similarity of the criteria for controllability and observability is remarkable. There is a duality between observability and controllability, as (A, C) is observable if and only if (A^T, C^T) is controllable, see Heij et al. [HRS06]. This motivates the definition of the *dual problem* to system (3.4) by

$$\begin{aligned} \dot{x}(t) &= A^T x(t) + C^T u(t) \\ y(t) &= B^T x(t). \end{aligned}$$

The dual system (A^T, B^T) is then fully observable if (A, B) is fully controllable. Furthermore, the dual system (A^T, C^T) is controllable if (A, C) is observable.

For the rest of this thesis, we will assume full-state feedback for simplicity so that the observability of the considered system is assumed. The construction of *observer methods*, e.g., Luenberger observers, to compute the full system state in case of incomplete state feedback, can be found in [Tie12].

3.1.4. Stability

In the following, we will introduce the concept of stability of dynamical systems. In this context, stability means that small perturbations should have a limited effect on the temporal behavior of the system states. This means that if a system at rest is brought out of balance, the dynamics automatically counteract the perturbation, and the system is returned to its original position. Stability is often an important objective when selecting control functions to achieve satisfactory system performance. Furthermore, we will introduce how an appropriate choice of the control input variables could stabilize an unstable system.

When defining stability, we first consider autonomous linear differential systems without controls of the form

$$\dot{x}(t) = Ax(t), \quad x(t_0) = x_0 \in \mathbb{R}^{n_x}. \tag{3.5}$$

Obviously, the null vector is an equilibrium of the system (3.5), i.e., $\dot{\hat{x}} = A\hat{x} = 0$. As mentioned before, any rest position $\hat{x} \neq 0$ can be transformed into a rest position with

$\hat{x} = 0$. Thus the question is whether the state vector converges to zero starting from an initial value $x(0) = x_0 = 0$.

Definition 3.1.9 (Stability)
Let $\hat{x} = 0$ be a rest position of the system (3.5).

(i) The rest position $\hat{x} = 0$ is **stable**, if for all $\varepsilon \in \mathbb{R}$, $\varepsilon > 0$ there exists a $\delta \in \mathbb{R}$, $\delta > 0$, such that

$$\|x(t; x_0)\| \leq \varepsilon$$

is fulfilled for all $t \leq t_0$ and for all initial values x_0 with $\|x_0\| \leq \delta$.

(ii) The rest position $\hat{x} = 0$ is called **asymptotically stable**, if it is stable and there additionally exists a neighborhood $U \subseteq \mathbb{R}^{n_x}$ of $\hat{x}$, such that for all $x_0 \in U$:

$$\lim_{t \to \infty} \|x(t; x_0)\| = 0.$$

Next, we give a criterion that uses the eigenvalues of the state transition matrix to check for the stability of the system (3.5).

Theorem 3.1.10 ([Grü18])
Consider the linear time-invariant differential system (3.5) with matrix $A \in \mathbb{R}^{n_x \times n_x}$ with eigenvalues $\lambda_j \in \mathbb{C}$, $\lambda_j = a_j + ib_j$ and $j = 1, \dots, n_x$. The equilibrium $\hat{x} = 0$ is asymptotically stable if and only if all eigenvalues have a negative real part, i.e., $a_j < 0$ for all $j = 1, \dots, n$.

In this case, where every eigenvalue of the matrix A has a strictly negative real part, A is called a *Hurwitz matrix* or *Hurwitz* for short. Another instrument for the analysis of stability properties are *Lyapunov functions*, for which a comprehensive theory also exists for nonlinear model functions, see Grüne [Grü18].

So far, we have considered the stability of systems without control input. For a system with control inputs, i.e of the form

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \\ x(0) &= x_0, \end{aligned} \tag{3.6}$$

the concept of stability can then be transferred as follows.

Definition 3.1.11 (Stabilizability)
A linear system of the form (3.6) is called **stabilizable**, if for any $x_0 \in \mathbb{R}^{n_x}$ there exists a control function $u(t)$, such that $\lim_{t \to \infty} \|x(t; x_0)\| = 0$ is fulfilled.

If we want to stabilize an unstable system, this can be done by the use of *(state) feedback control*. The idea is to use the current state to determine the new input variables. The control function $u(t)$ is then defined by the state of the system $x(t)$:

$$u(t) = \mu(x(t)). \tag{3.7}$$

The function $\mu\colon \mathbb{R}^{n_x} \to \mathbb{R}^{n_u}$ is called a *control law*. According to our linear setting, it is reasonable to assume the control law also as linear, i.e., $u(t) = -Kx(t)$. The matrix $K \in \mathbb{R}^{n_u \times n_x}$ is also referred to as *feedback gain*. Inserting this control law into the system (3.6) results in a *closed-loop system*

$$\begin{aligned} \dot{x}(t) &= (A - BK)x(t), \\ x(0) &= x_0. \end{aligned} \tag{3.8}$$

Note, that the closed-loop feedback system no longer depends on the control $u(t)$. With the Definition 3.1.11 and the control $u(t) = -Kx(t)$ we can directly formulate the following lemma for the stability of the closed-loop system (3.8).

Lemma 3.1.12
A linear system of type (3.6) can be stabilized if and only if there exists a matrix $K \in \mathbb{R}^{n_u \times n_x}$, so that the system (3.8) is asymptotically stable.

This means that we can use the state feedback to manipulate the system's stability. By a sophisticated design of the feedback gain K, the eigenvalues of the system are shifted into the stable range, i.e., the real parts of the eigenvalues of $(A - BK)$ are negative.

3.2. Linear Quadratic Regulator

The general task of control algorithms is to transfer the current state $x(t)$ of a system into a specified reference state x_{ref} by a suitable selection of the control $u(t)$. There are numerous approaches to realizing this task. These include the classic P, I, and D controllers and their combination to PID controllers, Lunze [Lun16]. For decades, optimal controllers have been particularly important in this field. The classical approaches aim to achieve a reference state using linear state feedback. Optimal controllers additionally seek to minimize a certain target criterion at the same time, e.g., the energy required to transfer the system into the target state. The *linear quadratic regulator* (LQR) is a well-known linear optimal controller. Another important class of optimal controllers is the so-called H_∞-control, which is particularly robust to model uncertainties and disturbances, cf. Zhou and Doyle [ZD98].

In the following section, we will discuss the LQR in more detail. First, the definition of the LQR problem is formulated, followed by a variant form using the LQR for state tracking. Then, the classical solution using the algebraic Riccati equation and an alternative based on solving finite-dimensional nonlinear optimization problems are explained. The section is based on the theoretical work of Hespanha [Hes09] and is structured according to Tietjen [Tie12].

3.2.1. Problem Formulations and the Algebraic Riccati Equation

The linear quadratic regulator problem is formulated as an optimal control problem with an infinite time horizon. A quadratic objective function and linear constraints characterize it.

Definition 3.2.1 (Linear Quadratic Regulator Problem)
Let $x(t) \in \mathbb{R}^{n_x}$ be the state of a linear dynamical system with initial value $x(0) = x_0 \in \mathbb{R}^{n_x}$ and $u(t) \in \mathbb{R}^{n_u}$ the control for all $t \in [0, \infty]$. With the matrices $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$, $Q \in \mathbb{R}^{n_x \times n_x}$, and $R \in \mathbb{R}^{n_u \times n_u}$ we can formulate the **linear quadratic regulator problem** or short **LQR-problem** as

$$\begin{aligned} \min_{x,u} \int_0^\infty & x(t)^T Q x(t) + u(t)^T R u(t) \,\mathrm{d}t \\ \text{w.r.t. } & \dot{x}(t) = Ax(t) + Bu(t), \\ & x(0) = x_0. \end{aligned} \tag{3.9}$$

The LQR approach allows setting the relative influence of the quantities $u(t)$ and $x(t)$ on the controller design by choosing the entries of the matrices R and Q. These weighting matrices adjust the behavior of the controller. With higher values for the components of the matrix Q, the focus is put on the stabilization of the state. If the control effort should be kept low, this can be realized by higher weights in R. A sophisticated choice of Q and R is not always intuitive and is often solved as an independent problem with high numerical effort, Kemper [Kem15].

In Figure 3.1 it is shown how the current system state $x(t)$ is used to determine the control to

$$u(t) = -Kx(t).$$

If the feedback gain K is found by solving the linear quadratic controller problem from Definition 3.2.1, then an optimal controller is obtained.

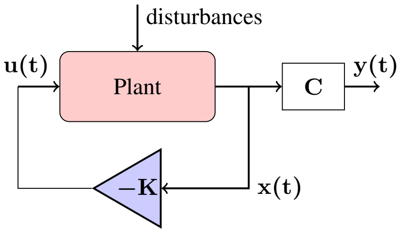

Figure 3.1.: *The control loop with LQR.*

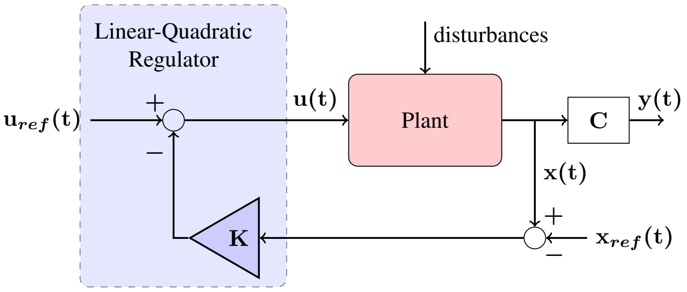

Figure 3.2.: *The control loop with LQR for tracking a reference.*

The classical linear quadratic regulator is designed to regulate the system to the final state $x_{ref} \equiv 0$. However, it can also be modified by a coordinate transformation to track different states. This is illustrated in Figure 3.2. For this purpose, the desired trajectory $x_{ref}(t)$ to be tracked can be selected beforehand. Furthermore, a reference control $u_{ref} \neq 0$ can be chosen, to which the feedback control is then added. As a result, we obtain the following problem formulation:

Definition 3.2.2 (Linear Quadratic Tracking Problem)
Assume the linear dynamical system from Definition 3.2.1, a reference trajectory $x_{ref}(t) \in \mathbb{R}^{n_x}$ and a reference control $u_{ref} = (0) \in \mathbb{R}^{n_u}$. Then we call

$$\begin{aligned} \min_{x,u} \int_0^\infty & (x(t) - x_{ref}(t))^T Q(x(t) - x_{ref}(t)) + (u(t) - u_{ref})^T R(u(t) - u_{ref}) \, \mathrm{d}t \\ \text{w.r.t. } & \dot{x}(t) = Ax(t) + Bu(t), \\ & x(0) = x_0, \end{aligned}$$

a **linear quadratic tracking problem**.

In order to apply the linear control law in this case, the transformation must be considered here as well, and it is

$$u(t) = -K\left(x(t) - x_{ref}(t)\right) + u_{ref}. \tag{3.10}$$

This transfer has the advantage that the feedback control strategy can include feedforward control strategies. In [Hes09], this problem is also treated under the term *optimal set-point control*. A formulation for the discrete case can be found in [Cat89]. We assume to have complete state space information. In the case of incomplete state space information, an additional output $y(t) = Cx(t)$ with $y(t) \in \mathbb{R}^{n_y}$ and $C \in \mathbb{R}^{n_y \times n_x}$ can be considered, compare Definition 3.1.1. In the following, it is shown that the solution of LQR problems is obtained from the *algebraic Riccati equation* (ARE). For this purpose, the ARE is defined, and some special properties of this matrix equation are given, which are required to solve LQR problems.

Definition 3.2.3 (Algebraic Riccati Equation)
Let $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$, $Q \in \mathbb{R}^{n_x \times n_x}$ and $R \in \mathbb{R}^{n_u \times n_u}$ be matrices. If R is invertible, then the **algebraic Riccati equation (ARE)** for the linear control system

$$\dot{x}(t) = Ax(t) + Bu(t), \quad x(0) = x_0,$$

is defined as

$$A^T S + SA - SBR^{-1}B^T S + Q = 0, \tag{3.11}$$

with a matrix $S \in \mathbb{R}^{n_x \times n_x}$.

The following theorem indicates under which assumptions on the matrices A, B, Q, and R the algebraic Riccati equation has a solution. Furthermore, a special property of the solution is shown.

Theorem 3.2.4 ([Hes09])
Let $Q \in \mathbb{R}^{n_x \times n_x}$ be positive semi-definite and $R \in \mathbb{R}^{n_u \times n_u}$ be positive definite. Furthermore let the system (A, B) be stabilizable and the pair $(A, \sqrt{Q})$ be observable. Then the algebraic Riccati equation (3.11) has exactly one positive definite solution S^* and this matrix S^* is symmetric.

The remarkable result is that the positive definite solution S^* of the algebraic Riccati equation (3.11) guarantees an asymptotically stable behavior of the closed-loop system without the direct consideration of the linear quadratic controller problem. With the prior information under which assumptions a solution of (3.11) exists, it is possible to use various numerical solution methods, for example, Newton's method. A good overview of different approaches to solve (3.11) is given by Bunse-Gerstner [Bun89]. In the following, we take an alternative approach to obtain an optimal solution for the (3.9) problem.

3.2.2. Optimal Solution of LQR

With the former statements, the theorem about the optimal solution of LQR problems can be formulated as follows.

Theorem 3.2.5 (Optimal Solution of LQR)
Let the system (A, B) be stabilizable and $(A, \sqrt{Q})$ observable, $Q \in \mathbb{R}^{n_x \times n_x}$ positive semi-definite and $R \in \mathbb{R}^{n_u \times n_u}$ positive definite, then the following statements hold:

- The algebraic Riccati equation
$$A^T S + SA - SBR^{-1}B^T S + Q = 0$$
has a unique positive definite solution S^*.
- The feedback law
$$u(t) = -Kx(t),\, t \in [0, \infty], \quad K := R^{-1}B^T S^*$$
minimizes the (3.9)-problem.
- The optimal objective function value of (3.9) is given by $x(0)^T S^* x(0)$.

Proof. See Benner [Ben09]. □

For this reason, the LQ regulator is often referred to as the *Riccati regulator or Riccati controller*. Tietjen [Tie12] shows that the solution of a (3.9) problem is equivalent to the solution of a corresponding (2.2). For this, the infinite-dimensional optimization problem is transformed into a finite-dimensional unconstrained optimization problem, which can take advantage of the well-studied results of nonlinear unconstrained optimization theory and the existing optimality criteria for optimization problems.

Suppose we already have such an optimal feedback law. Then we insert this into (3.9). This leads to

$$\begin{aligned} \min_{K} \quad & \int_0^\infty x(t)^T Q x(t) + x(t)^T K^T R K x(t) \,\mathrm{d}t \\ \text{w.r.t.} \quad & \dot{x}(t) = Ax(t) - BKx(t), \\ & x(0) = x_0. \end{aligned} \tag{3.12}$$

This represents the transformation from an infinite-dimensional optimization problem in x and u to a constrained finite-dimensional optimization problem in $K \in \mathbb{R}^{n_u \times n_x}$. We can further simplify this problem by inserting the solution of the differential equation in the constraints directly into the objective function and thus obtain an unconstrained nonlinear optimization problem. Since the differential equation in the constraint is linear, the exact solution is given as a function of the initial value $x_0 \in \mathbb{R}^{n_x}$ and gain K by

$$x(t; K, x_0) = e^{(A-BK)t} x_0 \ , \qquad t \in [0, \infty). \tag{3.13}$$

Inserting the solution (3.13) into the objective function of the problem in (3.12) results in the unconstrained finite-dimensional nonlinear optimization problem

$$\min_{K} \ J(K) = \int_0^\infty x(t; K, x_0)^T Q x(t; K, x_0) + x(t; K, x_0)^T K^T R K x(t; K, x_0) \,\mathrm{d}t. \tag{3.14}$$

Tietjen [Tie12] shows that transforming the LQR control process into the NLP problem (3.12) represents an equivalence. This also clarifies the origin of the gain K for the formulation of the feedback control law $u(t) = -Kx(t)$. The feedback gain is given by the uniquely-defined positive definite solution of (3.11). We can now consider the (3.9) problem without any restrictions as a finite-dimensional nonlinear optimization problem (2.2). This transfer allows the use of advanced methods from optimization, such as parametric sensitivity analysis. Furthermore, the idea of real-time approximations of disturbed solutions can be extended to the class of LQR problems.

3.3. Real-Time Adaption of the Linear Quadratic Regulator

As we have seen before, solving an LQR problem is equivalent to solving a corresponding NLP. This solution approach enables the application of parametric sensitivity analysis methods to parametric LQR problems. In particular, sensitivity information is used to estimate changes within the control law if disturbances in the parameter values of the system matrices occur. The idea of updating the linear feedback matrices by parametric sensitivities was first presented for LQR problems with full state feedback by Büskens [Büs09]. Furthermore, Tietjen [Tie12] extended this approach to systems with incomplete state feedback and transferred the approximation algorithms to optimal observer methods. The basics of sensitivity analysis for parametric nonlinear programs have already been shown in Subsection 2.2.2. For further steps, we first need the theoretical background on how these methods can adapt the LQR gain in the presence of parametric disturbances. We base our argumentation on the approach of Tietjen [Tie12].

3.3.1. Parameter-Dependent LQR

The linear quadratic regulator problem (3.9) is expanded to nonlinear disturbances p in the system matrices.

Definition 3.3.1 (Parametric Linear Quadratic Regulator Problem)
Let $p \in P \subset \mathbb{R}^{n_p}$ be the vector of constant parameters and $x(t) \in \mathbb{R}^{n_x}$ the state of a system with initial value $x_0(\cdot)\colon P \to \mathbb{R}^{n_x}$ and $u(t) \in \mathbb{R}^{n_u}$ the control for all $t \in [0, \infty]$. With the mappings $A(\cdot)\colon P \to \mathbb{R}^{n_x \times n_x}$, $B(\cdot)\colon P \to \mathbb{R}^{n_x \times n_u}$, $Q(\cdot)\colon P \to \mathbb{R}^{n_x \times n_x}$ and $R(\cdot)\colon P \to \mathbb{R}^{n_u \times n_u}$ we can formulate the **parametric linear quadratic regulator problem (LQR(p))** as

$$\begin{aligned} \min_{x,u} \quad & \int_0^\infty x(t)^T Q(p) x(t) + u(t)^T R(p) u(t) \, \mathrm{d}t \\ \text{w.r.t.} \quad & \dot{x}(t) = A(p)x(t) + B(p)u(t), \\ & x(0) = x_0(p). \end{aligned} \tag{3.15}$$

The problem (3.15) corresponds to (3.9) for a fixed $p \in P$. We make the assumption that for an initial perturbation $p_0 \in P \subset \mathbb{R}^{n_p}$ all required assumptions for $A(p_0)$, $B(p_0)$, $Q(p_0)$ and $R(p_0)$ in Theorem 3.2.5 are fulfilled. Then there exists an optimal feedback law

$$u(t) = -K(p_0)x(t). \tag{3.16}$$

The notation $K(p_0)$ explicitly states the dependence of the gain K on p_0. We can now replace the control in (3.15) with the feedback law (3.16) and transform the infinite-dimensional problem (3.15) to a parametric finite-dimensional nonlinear optimization problem with constraints. In general, this can be formulated in (2.7) by

$$\begin{aligned} \min_{x,K,p} \quad & J(K,p) = \int_0^\infty x(t)^T Q(p) x(t) + x(t)^T K(p)^T R(p) K(p) x(t) \, \mathrm{d}t \\ \text{w.r.t.} \quad & \dot{x}(t) = A(p)x(t) - B(p)K(p)x(t), \\ & x(0) = x_0(p). \end{aligned} \tag{3.17}$$

Thanks to the existence and uniqueness of initial value problems with linear differential equations, the constraint's differential equation possesses a uniquely defined solution denoted as $x(t) = x(t; K(p), p)$. This solution depends on the perturbation p and the coefficients of the feedback matrix $K(p)$. The solution can be inserted directly into the objective function analogous to Subsection 3.2.2. This transforms (3.17) to an unconstrained nonlinear parametric optimization problem.

$$\min_{K,p} J(K,p) = \int_0^\infty x(t;K,p)^T Q(p) x(t;K,p) + (Kx(t;K,p))^T R(p) K x(t;K,p) \mathrm{d}t \quad (3.18)$$

When determining a feedback matrix $K(p)$ for a given perturbation p, it is possible to employ conventional nonlinear optimization techniques, such as the sequential quadratic programming methods (SQP). This is an alternative to the solution approaches used for the algebraic Riccati equation.

3.3.2. Adaptive LQR

In the following, we study situations where online parametric perturbations occur. More precisely, this corresponds to a situation where, during a concrete realization of the Riccati control on a specific application system, deviations Δp from a given nominal value p_0 with $p = p_0 + \Delta p$ are expected. In this case, the feedback law given by $u(t) = -K(p_0)x(t)$ is no longer optimal. The control law can even become unstable, which means that the objective function value is no longer limited, and a new computation of the feedback gain $K(p_0 + \Delta p)$ is required. Even in less dramatic cases, a new computed gain is desirable, but usually too time-consuming. We have already shown in Subsection 2.2.3 how the solutions of nonlinear parametric optimization problems can be updated with respect to the perturbation parameters through a Taylor approximation. This is done by adapting the feedback law with sensitivity-based updates. Thus, a real-time alternative to the recalculation of the feedback gain is provided.

Next, we show under which conditions the optimal gain $K(p)$ is a continuously differentiable function with respect to p, and we present a formula for the computation of the sensitivity derivatives.

Theorem 3.3.2 (Approximation Theorem for Parametric Linear Quadratic Regulator Problems)
Let the mappings $A(\cdot)\colon P \to \mathbb{R}^{n_x \times n_x}$, $B(\cdot)\colon P \to \mathbb{R}^{n_x \times n_u}$, $Q(\cdot)\colon P \to \mathbb{R}^{n_x \times n_x}$ and $R(\cdot)\colon P \to \mathbb{R}^{n_u \times n_u}$ and $x_0(\cdot)\colon P \to \mathbb{R}^{n_x}$ be three times continuously differentiable with respect to p. For $p_0 \in \mathbb{R}^{n_p}$ let $(A(p_0), B(p_0))$ be stabilizable, $(A(p_0), \sqrt{Q(p_0)})$ observable, $Q(p_0) \neq 0$ positive semi-definite and $R(p_0)$ positive definite. Then there exists an environment $U(p_0) \subset \mathbb{R}^{n_p}$ of p_0 and a uniquely defined continuously differentiable function $K(\cdot)\colon U(p_0) \to \mathbb{R}^{n_u \times n_x}$ with the following properties:

- For all $p \in U(p_0)$ the feedback gain $K(p)$ satisfies the necessary and sufficient optimality conditions of the disturbed (3.15).
- There is exactly one positive definite solution $S^*(p)$ of the algebraic Riccati equation

  $$A(p)^T S + SA(p) + Q(p) - SB(p)R(p)^{-1}B(p)^T S = 0,$$

  with

  $$K(p) = R(p)^{-1}B(p)^T S^*(p).$$
- It is

  $$\frac{dK(p_0)}{dp} = -\nabla_K^2 J(K(p_0), p_0)^{-1} \bigtriangledown_{K,p} J(K(p_0), p_0).$$
- The feedback law

  $$u(t) = -K(p)x(t), \quad t \in [0, \infty],$$

  is optimal with respect to the perturbed objective function $J(K(p), p)$ and defines an asymptotic stable feedback system with the objective function value $J(K(p), p) = x_0^T S^*(p) x_0$.
- For the approximation of the perturbed feedback matrix

  $$\tilde{K}(p) = K(p_0) + \frac{dK}{dp}(p_0)\Delta p,$$

  the following estimates apply for all $p \in U(p_0)$:

  $$\|K(p) - \tilde{K}(p)\| = \mathcal{O}(\|\Delta p\|^2),$$
  $$\|J(K(p), p) - J(\tilde{K}(p), p)\| = \mathcal{O}(\|\Delta p\|^3).$$

Proof. The proof can be found in [Tie12]. □

According to Theorem 3.3.2 a solution of the problem (3.15) exists as a function $K(p)$ which is continuously differentiable with respect to p. To estimate this function in a point $p \neq p_0$, we use a first-order Taylor series approximation of the form

$$K(p) \approx \tilde{K}(p) = K(p_0) + \frac{dK}{dp}(p_0)\Delta p. \tag{3.19}$$

This method of sensitivity updates for the feedback law has several advantages. First of all, the only additional numerical effort to achieve this online method is the multiplication of the sensitivity derivative $\frac{dK}{dp}(p_0)$ with the perturbation Δp which generally corresponds to the multiplication of a tensor of the dimension $n_u \times n_x \times n_p$ with a vector of the dimension n_p.

The feedback law of the parametric LQ regulator becomes

$$u(t) = -K(p)x(t) \approx -\tilde{K}(p)x(t) = -\Big(K(p_0) + \frac{dK}{dp}(p_0)\Delta p\Big)x(t). \tag{3.20}$$

The nominal solution $K(p_0)$ and the sensitivity derivatives $\frac{dK}{dp}(p_0)$ can be computed offline. Therefore, this method is particularly applicable to real-time operations. The linear approximation of the perturbed feedback law leads to a quadratic approximation of the objective function and, thus, to a significant improvement of the solution's optimality. This also means a remarkable enhancement in the stability of the solution because the objective function of an unstable problem would tend to infinity. A disadvantage of the presented method is that no statements about the size of the environment $U(p_0)$ can be made. Therefore, it is necessary to check for each application which orders of magnitude are feasible for the parameter perturbations to obtain satisfactory results by parametric sensitivity analysis. This can be done by numerical simulations.

4. Nonlinear Model Predictive Control

4.1. Introduction

We now turn to nonlinear feedback control after introducing linear optimal feedback controllers in the form of LQR. For this purpose, the idea of *model predictive control (MPC)* is explained in this chapter. This method allows, in particular, for the control of constrained processes. Linear model predictive control is theoretically well understood and is widely used in industrial applications, cf. [ML99; Lee11]. In general, most real processes are nonlinear, and the necessary linearization of the system introduces inaccuracies in the control process. *Nonlinear model predictive control (NMPC)* avoids this by directly including the nonlinear system model. Figure 4.1 shows the control loop for model predictive control with full state feedback. It is based on solving nonlinear *optimal control problems (OCP)*. In an OCP, an optimal control is computed, which transfers a system from an initial state to a final state. In this process, an objective functional is minimized, and the differential equations of the system and additional constraints must be satisfied. Instead of completely executing the found solution on the system, NMPC applies only a first sequence of the optimal control on the considered system. As soon as measurements y of the system's current state are available, the OCP is adapted, i.e., the new measurements are used as initial values and the time horizon is shifted forward. Then, the new OCP is solved again. This results in the optimal open-loop control becoming an optimal closed-loop control strategy. By using a nonlinear model, the process behavior can usually be predicted better than with controls computed from a linearized model. Another advantage of NMPC is the straightforward integration of

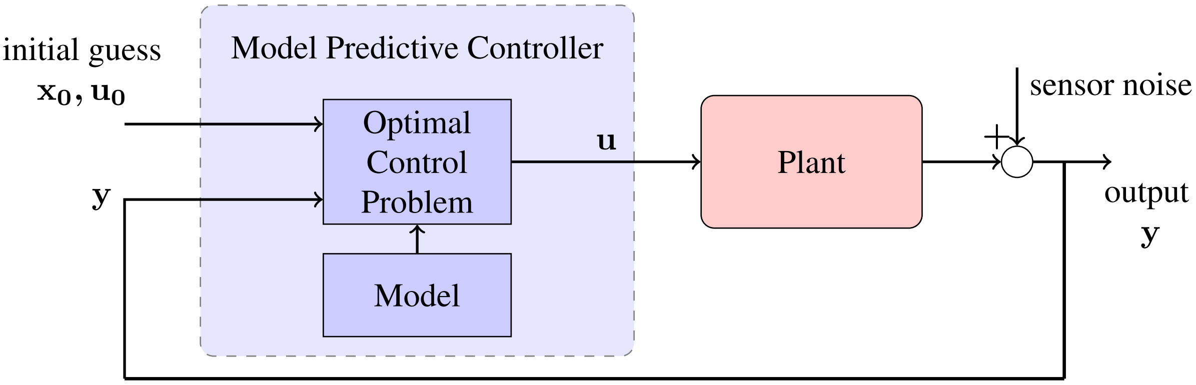

Figure 4.1.: *The basic control loop for nonlinear model predictive control (NMPC).*

equality and inequality constraints in the OCP. However, the good predictive properties of a nonlinear model predictive controller are also accompanied by difficulties. Among other things, the computational effort is significantly increased, and the proof of a stabilization property of the controller is more challenging than in the linear case. However, with certain assumptions on the problem, there are also stability results for NMPC. For this purpose, terminal constraints are often added as a state constraint to ensure stability, see Graichen [Gra12]. Comprehensive information on nonlinear MPC can be found in [GP17; RMD17; RL19].

Section 2.2 provided all preconditions to cope with static optimization problems and to solve them with numerical methods, like the SQP approach. Furthermore, for the solution of nonlinear parameter identification problems, we have included constraints that involve dynamics in time, compare Definition 2.3.2. To solve these problems, we have used the full discretization of the dynamics. This approach has allowed us to offload the need for our optimization variables to satisfy the system dynamics at each time point into the constraints, thus avoiding the costly solution of the differential equation each time the objective function is evaluated. The original idea of this approach comes from the solution of optimal control processes. The basis of nonlinear optimal control problems are usually nonlinear control systems, as we already encountered in Definition 3.1.2. In this context, the trajectories of the control and state variables that are optimal with respect to certain selected criteria while preserving the system behavior and constraints are sought. Moreover, these control problems are the basic modules for nonlinear model predictive control.

This chapter starts by explaining the basic notions of optimal control theory and derives Pontryagin's theoretical conditions for their solvability in Subsection 4.2.1. Then Subsection 4.2.2 briefly discusses different approaches for the numerical solution of OCPs, particularly focusing on direct methods. Based on this, the transcription tool TransWORHP that belongs to the NLP solver WORHP is briefly introduced and explained. The second part of this chapter is then devoted to the formulation of the NMPC algorithm and presents the stability statements for different formulation possibilities. Finally, important techniques for efficient application are explained. The first part of this chapter on the theoretical foundations of OCPs is based on Liberzon [Lib11] and Büskens [Büs98]. The second part on NMPC follows the arguments of Graichen [Gra12] and Grüne and Pannek [GP17].

4.2. Nonlinear Optimal Control

Henceforth, we will consider autonomous parametric nonlinear control systems of the same form as in Definition 3.1.2. For $t \in [t_0, t_f]$ we consider the system:

$$\dot{x}(t) = f(x(t), u(t), p), \quad x(t_0) = x_0 \in \mathbb{R}^{n_x}. \tag{4.1}$$

The goal of optimal control is to find control maneuvers $u(t) \in U \subseteq \mathbb{R}^{n_u}$ that control a system (4.1) optimally, i.e., minimizing a given cost functional, for a given control task. We assume without loss of generality that it is $t_0 = 0$. Further, let the components in the state vector be piecewise continuous differentiable and the controls be piecewise continuous[1]:

$$x \in C^1_\mathrm{p}([t_0, t_f], \mathbb{R}^{n_x}) \text{ and } u \in C^0_\mathrm{p}([t_0, t_f], U).$$

Moreover, we assume that for a fixed parameter p, the autonomous function $f\colon \mathbb{R}^{n_x} \times U \to \mathbb{R}^{n_x}$ is locally Lipschitz in x to guarantee the existence and uniqueness of solutions, see Liberzon [Lib11]. We assume the parameter dependence of the dynamical system in the following but will skip explicitly writing out the parameter notation to keep the expressions concise.

4.2.1. Standard Nonlinear Optimal Control Problem

Before formulating an optimal control problem, we first need a functional that we want to optimize. We define this *cost functional* as follows.

Definition 4.2.1 (Cost Functional)
Let the functions $\varphi\colon \mathbb{R}^{n_x} \to \mathbb{R}$ and $l\colon \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ be sufficiently often continuously differentiable. Then

$$J(x, u, t_f) = \varphi(x(t_f)) + \int_{t_0}^{t_f} \ell(x(t), u(t))\,\mathrm{d}t$$

is called the **objective functional** or **cost functional**.

Here, t_f denotes the final time, and $x(t_f)$ is the corresponding final state of the controlled system. The mapping ℓ is called the *running cost* and φ the *final cost.* The optimal control process is called a *Bolza problem* using this type of objective functionals. If the final cost is zero, it is called a *Lagrangian problem.* The final cost is also called the *Mayer term.* If $\varphi \neq 0$ holds and at the same time the running cost vanishes, the OCP is

[1] Here, $C^j(I, \mathbb{R}^n)$ denotes the class of j-times continuously differentiable functions from a domain $I \subset \mathbb{R}^k$ to $\mathbb{R}^n$, and $C^j_\mathrm{p}(I, \mathbb{R}^n) \subset C^{j-1}(I, \mathbb{R}^n)$ the class of j-times piecewise continuously differentiable functions. This is a standard notation where C_p refers to *piecewise* and is not related to the parameters at this point.

called *Mayer problem*. The different representations of an optimal control problem are equivalent and can be transformed into each other, compare Liberzon [Lib11].

A distinction is made between problems with fixed final time t_f and those with free final time. If the objective of a control problem is to minimize the final time, i.e., to drive the system to the desired state as quickly as possible, the cost functional is of the form

$$J(x, u, t_f) = \int_{t_0}^{t_f} 1 dt.$$

In this case, t_f is introduced as an additional optimization parameter. Thus, a problem with free final time can be transformed into a problem with fixed final time. Therefore, in the following, we assume problems with fixed final time.

The optimal control problem can now be formulated as the minimization problem of the cost functional $J(x, u)$ while satisfying the dynamical control system (4.1). Often, additional constraints are imposed on the initial state $x(t_0)$ and final state $x(t_f)$. For this purpose, we consider *boundary constraints* $\psi(x(t_0), x(t_f)) = 0$ depending on the initial and final states of the system. In many applications, we also need to impose more general constraints on the states and the controls in the form of *inequality constraints* $c(x(t), u(t)) \leq 0$ for almost all time points $t \in [t_0, t_f]$. These additional constraints are often referred to as *path constraints*. If the constraints c depend only on the states x, we speak of *state constraints*. If the function c, in turn, depends only on the controls u, the constraints are called *control constraints*. Often these are simple *box constraints* $u_{min} \leq u(t) \leq u_{max}$ for $t \in [t_0, t_f]$ and $u_{min}, u_{max} \in \mathbb{R}^{n_u}$.

Definition 4.2.2 (Optimal Control Problem (OCP))
Let $J(x, u)$ be a nonlinear functional according to Definition 4.2.1. Then the problem

$$\begin{aligned} \min_{x,u} \quad & J = \varphi(x(t_f)) + \int_{t_0}^{t_f} \ell(x(t), u(t)) \, \mathrm{d}t \\ \text{w.r.t.} \quad & \dot{x}(t) = f(x(t), u(t)), \quad t_0 \leq t \leq t_f, \\ & \psi(x(t_0), x(t_f)) = 0, \\ & c(x(t), u(t)) \leq 0, \qquad t_0 \leq t \leq t_f, \end{aligned} \tag{4.2}$$

is called an **optimal control problem (OCP)**. The constraints are defined by sufficiently often continuously differentiable functions $\psi\colon \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \to \mathbb{R}^{n_\psi}$ and $c\colon \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_c}$ with $n_\psi, n_c \in \mathbb{R}$, $0 \leq n_\psi \leq 2n_x$.

How can the solutions of such an optimal control problem with fixed final time be characterized? First, this can be done by their admissibility and, in the second step, by their optimality.

Definition 4.2.3 (Admissible and Optimal Solution)
A solution (x^*, u^*) of the system (4.1) that satisfies the constraints of (4.2) on the time interval $[t_0, t_f]$ is called an **admissible solution**. Moreover, if the admissible solution (x^*, u^*) satisfies

$$J(x^*, u^*) \leq J(x, u, p),$$

for all admissible solutions (x, u) then it is called **optimal solution**. In this case, the vector x^* denotes the **optimal trajectory** and u^* the **optimal control** of the problem (4.2).

Necessary Optimality Conditions

For the formulation of necessary conditions and the famous *Pontryagin minimum principle*, we consider an 4.2 without path constraints. The dynamical system is assumed to be autonomous, and the cost function is assumed to be time-independent. The Pontryagin minimum principle was established by Pontryagin around 1955 as a maximum principle and proved in the following years by Pontryagin et al. [Pon+62]. Together with the Hamilton-Jacobi-Bellman equation, it represents the central theory of optimal control for characterizing optimal solutions of optimal control processes. We start with defining the *Hamiltonian function* and introduce *adjoint variables.*

Definition 4.2.4 (Hamiltonian and Adjoint Variable)
Let $\lambda_0 \in \mathbb{R}$ be a constant and $\lambda \in \mathbb{R}^{n_x}$ a column vector. Then the function $\mathcal{H}\colon \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \times \mathbb{R} \to \mathbb{R}$ defined as

$$\mathcal{H}(x, u, \lambda, \lambda_0) \coloneqq \lambda_0 \ell(x, u) + \lambda^T f(x, u)$$

is called the **Hamiltonian** to the control process (4.2). The vector λ is called **adjoint variable** or **co-state** to x.

The main result of the theory of optimal control processes can now be formulated.

Theorem 4.2.5 (Pontryagin Minimum Principle)
Let the functions φ, ℓ, f, and ψ be continuous, and continuous differentiable with respect to x and u. Further, let (x^*, u^*) be an optimal solution of the problem (4.2) without state-control or state constraints.
Then, there exist multipliers $\lambda_0 \in \mathbb{R}$, $\rho \in \mathbb{R}^{n_\psi}$ and a continuous and a piecewise continuously differentiable function $\lambda(t)\colon [t_0, t_f] \to \mathbb{R}^n$, such that the following hold:

(i) It is $l_0 \geq 0$, $(l_0, \rho, \lambda^T(t)) \neq 0$, $\forall\, t \in [t_0, t_f]$.
(ii) For all points $t \in [t_0, t_f]$ in which u^* is continuous, it holds

(a) the optimality condition

$$\mathcal{H}(x^*(t), u^*(t), \lambda(t), l_0) = \min_{u \in U} \mathcal{H}(x^*(t), u(t), \lambda(t), l_0),$$

(b) and the adjoint differential equation

$$\dot{\lambda}^T(t) = -\nabla_x \mathcal{H}(x^*(t), u^*(t), \lambda(t), \lambda_0),$$

(iii) The transversality conditions apply:

$$\begin{aligned} \lambda(t_0) &= -\nabla_{x(t_0)}(\lambda_0 \varphi(x^*(t_f)) + \rho^T \psi(x^*(t_0), x^*(t_f)))^T, \\ \lambda(t_f) &= \ \ \nabla_{x(t_f)}(\lambda_0 \varphi(x^*(t_f)) + \rho^T \psi(x^*(t_0), x^*(t_f)))^T. \end{aligned}$$

(iv) In the special case of autonomous processes it is

$$\mathcal{H}(x^*(t), u^*(t), \lambda(t), l_0) = \textit{constant}, \quad \forall\, t \in [t_0, t_f].$$

(v) In the case of a free final time t_f it holds

$$\mathcal{H}(x^*(t_f^*), u^*(t_f^*), \lambda(t_f^*), l_0) = 0.$$

Proof. A proof can be found in [Lib11]. □

The minimum principle leads to a two-point boundary value problem that has to be solved. However, finding analytical solutions is often difficult, even for small problems with simple dynamics. An analytical solution is rarely possible in more complex applications, so numerical algorithms are needed to approximate the solutions.

4.2.2. Numerical Solution Methods

The solution methods for optimal control problems can be divided into two approaches: the *direct* and the *indirect methods*. In this thesis, direct methods are applied. For this reason, the indirect approaches are discussed very briefly before the implementation of the direct methods is described in more detail.

Indirect Methods

Indirect methods use the Pontryagin minimum principle and reduce the problem (4.2) to a two-point boundary value problem. The necessary optimality conditions of the optimal control theory are used to eliminate the control u from (4.2). For this purpose, a Hamiltonian function is introduced as described in Section 4.2.1, and Pontryagin's minimum principle 4.2.5 is used. The obtained boundary value problem can be solved by shooting methods [BS05]. Indirect methods are known for their high accuracy but require a deep understanding of the control theory by the user. Additionally, providing accurate initial estimates for the adjoint variables can be challenging [Büs98; Lib11].

Direct Methods

The direct solution approach is based on the direct discretization of control and state variables, which approximates the infinite-dimensional problem (4.2) by a finite-dimensional nonlinear optimization problem (2.2). The resulting optimization problem can then be solved with appropriate methods, such as the SQP method presented in Section 2.2.1. This also has the advantage that comparatively little theoretical knowledge from the field of control theory is required for its application. In addition, the solution of the adjoint equations is avoided, but if necessary, they can be computed afterward from the NLP solution. In the past, one of the main disadvantages of direct methods was the low accuracy of the solution. Today's high computing capacities allow for fast computation times and large memory capacities. So, a fine discretization of the problems can be used, and highly accurate solutions are obtained. Today's high computing power allows for fast computation times and large memory capacities so that a fine discretization of the problems can be used and highly accurate solutions can be obtained.

For the numerical solution, the integral in the objective function and the right-hand side of the differential equations of (4.2) have to be evaluated. For this, we need approximations of the states and controls at selected grid points. To get them, the time interval $[t_0, t_f]$ is replaced by an equidistant grid G_h with discrete-time points τ_i for $i = 0, \ldots, N$ and $N \in \mathbb{N}$, compare Subsection 2.4.1. Then the states are discretized at these time points,

$$\tilde{x}_i \approx x(\tau_i), \qquad \text{for } i = 0, \ldots, N.$$

The discretization of the controls is done analogously by

$$\tilde{u}_i \approx u(\tau_i), \qquad \text{for } i = 0, \ldots, N.$$

All discretized states and controls are then summarized in a state and control vector

$$\tilde{x} := (\tilde{x}_0^T, \ldots, \tilde{x}_N^T)^T \in \mathbb{R}^{n_x \cdot (N+1)}$$

and

$$\tilde{u} := (\tilde{u}_0^T, \ldots, \tilde{u}_N^T)^T \in \mathbb{R}^{n_u \cdot (N+1)}.$$

Then the differential equations in (4.1) can be discretized with any integration scheme as described in Subsection 2.4.1, e.g., an explicit Runge-Kutta method, cf. (2.15). If the integral term, i.e., the running cost, (4.2) is nonzero, then the integral is numerically approximated between two time points τ_i and τ_{i+1} by

$$\int_{\tau_i}^{\tau_{i+1}} \ell(x(t), u(t)) \, \mathrm{d}t \approx (\tau_{i+1} - \tau_i) \ell(\tilde{x}_i, \tilde{u}_i).$$

A more general integration rule is obtained using Euler's method. With

$$\int_{\tau_0}^{\tau_{i+1}} \ell(x(t), u(t))\ dt \approx \int_{\tau_0}^{\tau_i} \ell(x(t), u(t))\ \mathrm{d}t + (\tau_{i+1} - \tau_i)\ell(\tilde{x}_i, \tilde{u}_i), \quad \text{for } i = 0, \dots, N-1,$$

we get

$$\int_{\tau_0}^{\tau_f} \ell(x(t), u(t))\ \mathrm{d}t \approx \sum_{i=0}^{N-1} \underbrace{(\tau_{i+1} - \tau_i)}_{=h} \ell(\tilde{x}_i, \tilde{u}_i).$$

Further approximation methods for integrals, also called *quadrature formulas*, can be found in Schwarz and Köckler [SK11]. We obtain the discretized version of (4.2) after full discretization.

Definition 4.2.6 (Discretized Optimal Control Problem (dOCP))
Let $\Phi(\tilde{x}, \tilde{u}; h; f)$ be a one-step method with step size h and G_h an equidistant grid of the time interval $[t_0, t_f]$ with the discrete time points $\tau_i \in G_h$, $\tau_0 < \tau_1 < \dots < \tau_N$. Then

$$\begin{aligned} \min_{\tilde{x}, \tilde{u}} \quad & J = \varphi(\tilde{x}_0, \tilde{x}_N) + \sum_{i=0}^{N-1} h\ell(\tilde{x}_i, \tilde{u}_i) \\ \text{w.r.t.} \quad & \tilde{x}_{i+1} - \tilde{x}_i - h \cdot \Phi(\tilde{x}, \tilde{u}; h; f) = 0, \qquad \text{for } i = 0, \dots, N-1, \\ & \psi(\tilde{x}_0, \tilde{x}_N) = 0, \\ & c(\tilde{x}, \tilde{u}) \leq 0, \end{aligned} \tag{4.3}$$

describes the **discretized optimal control problem** to Definition 4.2.2.

The states $\tilde{x}$ and control $\tilde{u}$ can be combined into a vector $z \in \mathbb{R}^{(n_x+n_u)\cdot(N+1)}$ of optimization variables and also the constraints can be grouped together, thus obtaining a finite-dimensional optimization problem of the form (2.2)

$$\begin{aligned} \min_{z} \quad & F(z), \\ \text{w.r.t.} \quad & g_i(z) \leq 0, \quad i = 0, \dots, m. \end{aligned}$$

This problem can now be solved with known methods and solvers for NLPs, such as WORHP.

4.2.3. TransWORHP

The software library TransWORHP ('Transcription for WORHP') extends the NLP solver WORHP from Subsection 2.2.4 with methods for solving optimal control problems [KB12]. Different direct methods for transcribing the optimal control problem into an optimization problem are provided, such as full discretization, multiple shooting, and pseudospectral methods. The resulting nonlinear optimization problem is then solved using WORHP.

In the context of this work, we use the full discretization approach to solve both the dynamic parameter identification problems, as described earlier in Section 2.4, and the optimization problems that arise in the context of nonlinear model predictive control. TransWORHP currently provides Euler's method, the trapezoidal rule, and the Hermite-Simpson method as integration methods for solving the differential equation system. This work primarily uses the trapezoidal method presented in (2.16). It is applied to integrate system dynamics in both the parameter identification problems in Chapter 5 and Chapter 6 and the OCPs that arise in the context of NMPC in Chapter 6. Only for the offline computation of the feedback gain $K(p_0)$ in the `adLQR+PI`-Algorithm 5.1, the Hermite-Simpson method described in (2.17) is applied. In full discretization, all control variables and all state variables are used for optimization, which can lead to very large NLP problems depending on the level of refinement of the discretization grid. However, these problems have sparse structures in the derivative matrices, so they can be solved particularly efficiently with WORHP, cf. Section 2.4.3. For WORHP and TransWORHPthe user can choose from different. For example, they allow for a choice of transcription methods.

The transcription tool has been successfully used in many different areas of industry and research. For example, TransWORHP has been used in the context of optimal maneuvering of ships [RWB17]. Further, Rick et al. [Ric+19] apply the transcription tool to OCPs in a model predictive controller for an autonomous driving project. Other applications include optimal trajectory planning for robots [Rau14], space applications [KB19], and parameter identification of dynamical systems [Ech14; Sch+18]. Next, we address the role of optimal control problems in nonlinear model predictive control.

4.3. The Basic Principles of NMPC

An NMPC algorithm aims to determine a nonlinear control law for a nonlinear constrained system. Common applications are stabilization tasks or state-tracking tasks. The general NMPC method is based on the repeated solution of dynamic optimal control problems on a moving finite time horizon. System dynamics and possible state and control constraints need to be considered. A distinction is made between the general *process time* $t \in [t_0, t_f] \subset \mathbb{R}$ and the *internal time* $\tau \in [0, T]$. Here T denotes the *prediction horizon* on which the state trajectory of the system is predicted.

Figure 4.2 shows schematically how the NMPC algorithm works. At the current process time t_k, $k \in \mathbb{N}_0^+$, the internal time is $\tau = \tau_0 := 0$. The current measured value of the system at this time $x_k := x(t_k)$ is used as the initial value for the current optimal control problem. Thus, $\bar{x}_0^{[k]} := \bar{x}^{[k]}(\tau = 0) = x_k$ applies to the internal trajectory. Using the system dynamics, the future behavior on the prediction interval $[0, T]$ can be predicted.

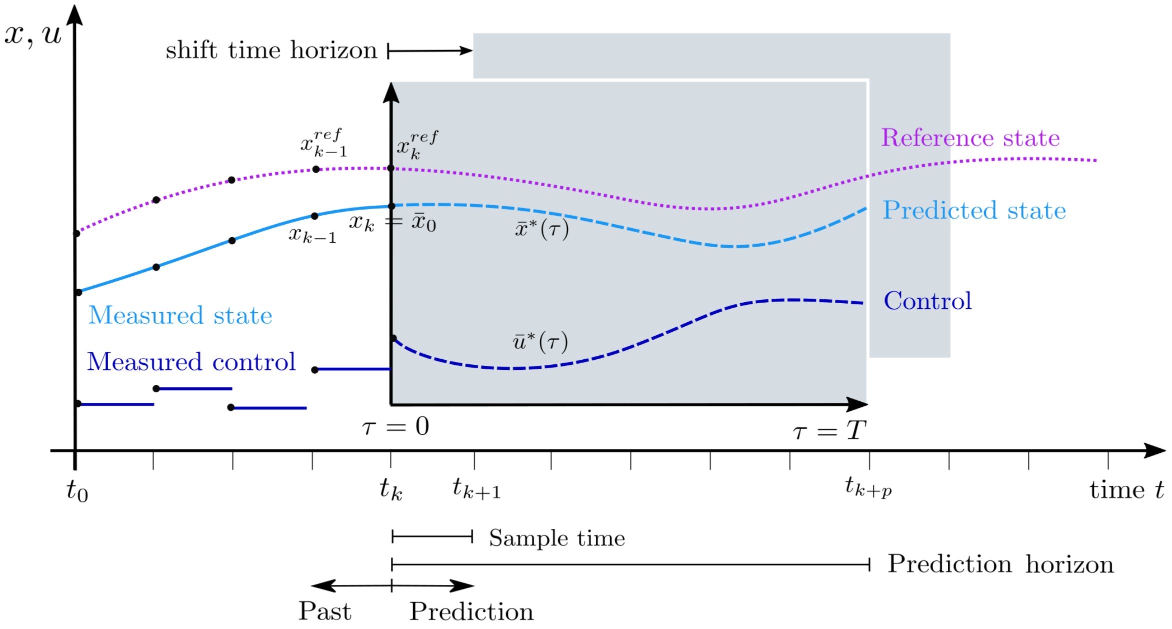

Figure 4.2.: *Schematic visualization of the NMPC procedure at time t_k; using the current measurements x_k, an OCP with $\bar{x}_0^{[k]} = x_k$ is solved on the prediction horizon $[0, T]$ and the optimal control $\bar{u}^*$ is applied on the interval $[t_k, t_{k+1}]$.*

By solving the optimal control problem, an optimal control $\bar{u}^{*,[k]}(\tau; \bar{x}_0^{[k]})$ for $\tau \in [0, T]$ is obtained that minimizes a previously defined objective functional $J(\bar{x}_0^{[k]}, \bar{u}^{[k]})$.
The predicted trajectories of the states and the controls are each shown as dashed blue lines in Figure 4.2. The measurements of the real system are represented as points $\{x_k, x_{k-1}, \dots, x_0\}$ and $\{u_k, u_{k-1}, \dots u_0\}$. If the used model perfectly represents the system behavior and no external disturbances affect the process, and if the optimal control process could be solved on an infinite time horizon, the computed solution would be optimal for all $t \geq t_k$. In reality, the solution of an OCP on an infinite time horizon can only be approximated numerically. Moreover, disturbances and model uncertainties exist in almost every real-world application. External influences, such as wind or temperature changes, may occur, or internal parameters change over time, e.g., resistances. As a result, measurements $x(t_k + T)$ usually deviate from the predicted system behavior $\bar{x}^{[k]}(\tau = T)$. To be able to react to uncertainties, only the first part of the computed control is applied to the process:

$$u(t_k + \tau) = \bar{u}^{*,[k]}\left(\tau; \bar{x}_0^{[k]}\right) \text{ for } \tau \in [0, \Delta t), \tag{4.4}$$

where $\Delta t = t_{k+1} - t_k$ is the *sampling time*. The corresponding trajectory of the states is given by

$$x(t_k + \tau) = \bar{x}^{*,[k]}\left(\tau; \bar{x}_0^{[k]}\right) \text{ for } \tau \in [0, \Delta t). \tag{4.5}$$

When new measurements x_{k+1} are provided at time t_{k+1}, the time interval for the optimal control problem is shifted forward by Δt in time. The new data x_{k+1} is used as the initial state, and a new control is determined by solving the revised OCP on the shifted time-domain $\tau \in [0, T]$. Therefore, the terms *receding horizon control* or *moving horizon control* are often used synonymously with model predictive control. Overall, the system trajectories of $x(t)$ and $u(t)$ are composed of the optimal solutions $\bar{x}^{*,[k]}(\tau)$ and $\bar{u}^{*,[k]}(\tau)$ on the time segments of the successive NMPC iterations. Thus, with consecutive process time $t \in [t_0, \infty)$, we obtain:

$$\begin{aligned} u(t) &= u(t_k + \tau) = \bar{u}^{*,[k]}\big(\tau; \bar{x}_0^{[k]}\big) \text{ for } \tau \in [0, \Delta t), \\ x(t) &= x(t_k + \tau) = \bar{x}^{*,[k]}\big(\tau; \bar{x}_0^{[k]}\big) \text{ for } \tau \in [0, \Delta t), \end{aligned}$$

for $t \in [t_k, t_{k+1})$, $k \in \mathbb{N}_0^+$. Since it is $\bar{x}_0^{[k]} = x_k$ the definition of $u(t)$ corresponds to the definition of a feedback control law in (3.7), that we already used in the linear case. The reference state $x^{ref}(t)$ is the desired equilibrium state to which the system should be transferred or stabilized. It is common to consider the stabilization of the origin of the system, $f(0, 0) = 0$. This is because any other equilibrium $(x^{ref}, u^{ref}) \in \mathbb{R}^{n_x} \times \mathbb{R}^{n_u}$ with $f(x^{ref}, u^{ref}) = 0$ can be transformed to the origin by

$$\hat{x} = x - x^{ref} \quad \text{and} \quad \hat{u} = u - u^{ref}.$$

However, a time-dependent (previously determined) trajectory can also be followed. For this purpose, the distance to the predicted trajectory to $x^{ref}(t)$ is minimized within the objective function of the OCP. This is done in addition to optimizing the control effort. Continuing as before in Definition 4.2.1, we consider an objective functional of the form

$$J(x_k, \bar{u}) = \varphi(\bar{x}(T)) + \int_0^T \ell(\bar{x}(\tau), \bar{u}(\tau))\, \mathrm{d}\tau.$$

Basic assumptions for the existence of a solution are that the integral cost term $\ell(\bar{x}, \bar{u})$ and the final cost $\varphi(\bar{x})$ are continuously differentiable in their arguments and positive definite functions with $\ell(0, 0) = 0$ and $\varphi(0) = 0$. Further, for theoretical stability statements, we need that for certain constants $c_\ell, C_\ell > 0$ and $c_\varphi, C_\varphi > 0$ the following quadratic estimates are valid:

$$c_\ell(\|x\|^2 + \|u\|^2) \le \ell(x, u) \le C_\ell(\|x\|^2 + \|u\|^2), \tag{4.6}$$

$$c_\varphi\|x\|^2 \le \ \varphi(x) \ \le C_\varphi\|x\|^2. \tag{4.7}$$

A typical choice for the objective function is equivalent to our former Definition 4.2.1:

$$J(x_k, \bar{u}^{[k]}) = \bar{x}^{[k]}(T)^T P \bar{x}^{[k]}(T) + \int_0^T \bar{x}^{[k]}(\tau)^T Q \bar{x}^{[k]}(\tau) + \bar{u}^{[k]}(\tau)^T R \bar{u}^{[k]}(\tau)\ \mathrm{d}\tau, \tag{4.8}$$

with symmetric and positive definite weighting matrices $P \in \mathbb{R}^{n_x \times n_x}$, $Q \in \mathbb{R}^{n_x \times n_x}$, and $R \in \mathbb{R}^{n_u \times n_u}$. These can be used by the user to choose a weight between running and final costs. The optimal control problem that has to be solved at time $t_k \in [t_0, t_f]$ within an MPC iteration can be represented more generally as:

$$\begin{aligned}
\min_{\bar{u}^{[k]}} \quad & J(\bar{x}_0^{[k]}, \bar{u}^{[k]}) = \varphi(\bar{x}^{[k]}(T)) + \int_0^T \ell(\bar{x}^{[k]}(\tau), \bar{u}^{[k]}(\tau))\, \mathrm{d}\tau \\
\text{w.r.t.} \quad & \dot{\bar{x}}^{[k]}(\tau) = f(\bar{x}^{[k]}(\tau), \bar{u}^{[k]}(\tau)), \quad \tau \in [0, T], \\
& \bar{x}^{[k]}(0) = x_k, \\
& \bar{x}^{[k]}(T) \in S, \\
& \bar{x}^{[k]}(\tau) \in X, \quad \tau \in [0, T], \\
& \bar{u}^{[k]}(\tau) \in U, \quad \tau \in [0, T].
\end{aligned} \tag{4.9}$$

It is assumed that the function on the right-hand side of the differential equation system in problem (4.9) is continuously differentiable and that for every $\bar{x}_0^{[k]} \in X$ and every control trajectory $\bar{u}^{[k]}(\tau) \in U$, $\tau \in [0, T]$ exists a bounded solution of the dynamical system. Moreover, let $f(0, 0) = 0$ and establish the initial conditions based on the current measured states x_k. In (4.9), the states and control constraints are given by sets to simplify the formulation of stability statements for model predictive control:

$$\bar{x}^{[k]}(\tau) \in X \subseteq \mathbb{R}^n, \quad \text{for all} \ \ \tau \in [0, T],$$

and,

$$\bar{u}^{[k]}(\tau) \in U(x) \subseteq \mathbb{R}^{n_u}, \quad \text{for all} \ \ \tau \in [0, T].$$

For the following theoretical statements, some further assumptions are needed on the sets X and $U(x)$, among others. It is assumed that X and $U(x)$ are convex and that $U(x)$ is additionally compact. Moreover, the origin should lie inside the sets:

$$0 \in X \subseteq \mathbb{R}^{n_x} \text{ and } 0 \in U \subseteq \mathbb{R}^{n_u}.$$

Furthermore, we introduce the set for terminal conditions $S \subseteq X$ with $0 \in S$. This is important in model predictive control for the stability propositions in the following Section 4.4. Assuming that the optimal control process (4.9) is solvable, the basic procedure of nonlinear model predictive control can be summarized in Algorithm 4.1. The central question now is under which conditions stability according to Definition 3.1.9 can be shown for the NMPC process.

4.4. Different NMPC Formulations and Stability

The applicability of NMPC to real processes depends crucially on the efficient solution of the optimal control problem and the stability of the closed-loop system. Even if the optimization algorithm finds a solution to the OCP, this fact alone does not guarantee

Algorithm 4.1 Basic Nonlinear Model Predictive Control (NMPC)

```
1: Set time $t_0 = 0$.
2: Choose a prediction horizon $T$ for the OCP.
3: Set iterator $k = 0$.
4: Measure the current system state $x(t_k)$.
5: Set the initial state of the OCP to $\bar{x}_0^{[k]} = \bar{x}^{[k]}(\tau_0 = 0) := x(t_k)$ and solve 4.9 on
   $\tau \in [0, T]$.
6: Denote the obtained optimal control sequence $\bar{u}^{*,[k]}(\cdot) \in U$ and apply its first se-
   quence as control to the system for the sample time $\Delta t$.
7: Set $k = k + 1$ and continue with point 4.
```

closed-loop stability. The main results concerning closed-loop stability for nonlinear systems are based on very different techniques. These include using terminal constraints or penalty terms in the objective functional and Lyapunov functions or invariant sets. Usually, the regulator problem, i.e., the stabilization of the origin, is considered in the state-space framework [CB07]. The definition of stability, we introduced for linear systems in Definition 3.1.9, is also valid for nonlinear systems. Without loss of generality, the origin is considered as an equilibrium of the system. The length of the prediction horizon and possible terminal conditions have an important influence on the stability of the MPC scheme. Thus, terminal constraints of the form $\bar{x}^{[k]}(T) \in S$ can be helpful to guarantee the stability. However, guaranteeing the existence of a solution in the constrained MPC formulation is difficult, and the numerical effort required to solve the OCP is usually much higher than in the case of a free final state. For this reason, these aspects are the focus of the following statements. The former assumptions on the state and control sets will enable the following statements. Note that in applications, it is often hard or impossible to verify them, and the applicability of NMPC is proven by numerical experiments.

4.4.1. Infinite Time Horizon

For better readability, we omit writing the index of the OCP in this section. Consider first the optimal control problem with an infinite time horizon.

$$\begin{aligned}
\min_{\bar{u}} \quad & J(\bar{x}_0, \bar{u}) = \int_0^\infty \ell(\bar{x}(\tau), \bar{u}(\tau)) \, \mathrm{d}\tau \\
\text{w.r.t.} \quad & \dot{\bar{x}}(\tau) = f(\bar{x}(\tau), \bar{u}(\tau)), \quad \tau \in [0, \infty), \\
& \bar{x}(0) = x_k, \\
& \bar{x}(\tau) \in X, \quad \bar{u}(\tau) \in U, \quad \tau \in [0, \infty).
\end{aligned} \tag{4.10}$$

The problem has neither a terminal cost term in the objective functional nor terminal constraints. This is because, if there exists an optimal solution $(\bar{x}^*, \bar{u}^*)$ of Problem (4.10), then the objective functional must have a finite value, i.e., $\lim_{\tau \to \infty} \ell(\bar{x}^*(\tau), \bar{u}^*(\tau)) = 0$. Thus, the desired final states are reached for an objective functional with integral term satisfying condition (4.6). A first theorem on the stability of the NMPC algorithm can be formulated.

Theorem 4.4.1 (Stability of NMPC with Infinite Time Horizon, [Gra12])
Let the set $\Gamma \subseteq X$ of all initial conditions x_k, for which the optimal control process with infinite time horizon (4.10) is solvable, be nonempty. Then, the origin of the controlled system (4.1) is asymptotically stable in terms of $\lim_{t\to\infty} \|x(t)\| = 0$.

Since the solution of an optimal control process with an infinite time horizon can only be approximated numerically, statements for problems on finite time horizons are more relevant for us.

4.4.2. Finite Time Horizon

To ensure stability despite a finite time horizon $T < \infty$, several options are presented in [Gra12]. The first version provides equality constraints on the final state. The resulting optimal control problem then becomes

$$\begin{aligned} \min_{\bar{u}} \quad & J(\bar{x}_0, \bar{u}) = \int_0^T \ell(\bar{x}(\tau), \bar{u}(\tau))\, \mathrm{d}\tau \\ \text{w.r.t.} \quad & \dot{\bar{x}}(\tau) = f(\bar{x}(\tau), \bar{u}(\tau)), \quad \tau \in [0, T], \\ & \bar{x}(0) = x_k, \\ & \bar{x}(T) = 0, \\ & \bar{x}(\tau) \in X, \quad \bar{u}(\tau) \in U, \quad \tau \in [0, T]. \end{aligned} \tag{4.11}$$

This demands that the optimal state trajectory $\bar{x}^*(\tau; \bar{x}_0)$ reaches the origin exactly at the end of the prediction horizon T. Assuming that, the following theorem holds.

Theorem 4.4.2 (Stability of NMPC with Finite Time Horizon and Terminal Constraint, [Gra12])
Let the set $\Gamma \subseteq X$ of all initial conditions x_k, for which the optimal control process with finite time horizon (4.11) is solvable, be nonempty. Then, the origin of the system (4.1) is asymptotically stable.

The set Γ is the *region of attraction* and can become very small, especially for a short prediction horizon T. In practice, especially when disturbances occur, this can lead to the fact that solvability can no longer be guaranteed. In addition, the equality constraint can increase the numerical solution effort. Alternative formulations replace the equality constraint with a final region. For this purpose, a set $S_\varepsilon \subseteq X$ with $0 \in S_\varepsilon$ is defined, in which the final states should lie. The optimizer is given some freedom by softening the constraints, which may increase its convergence rate. Moreover, the terminal conditions in the objective functional are again considered by adding the function $\varphi(x(T))$.

$$
\begin{aligned}
\min_{\bar{u}} \quad & J(\bar{x}_0, \bar{u}) = \varphi(x(T)) + \int_0^T \ell(\bar{x}(\tau), \bar{u}(\tau))\ \mathrm{d}\tau \\
\text{w.r.t.} \quad & \dot{\bar{x}}(\tau) = f(\bar{x}(\tau), \bar{u}(\tau)), \quad \tau \in [0,T], \\
& \bar{x}(0) = x_k, \\
& \bar{x}(T) \in S_\varepsilon, \\
& \bar{x}(\tau) \in X, \quad \bar{u}(\tau) \in U, \quad \tau \in [0,T].
\end{aligned}
\tag{4.12}
$$

If additional assumptions are imposed on the final region, asymptotic stability can be proved:

Theorem 4.4.3 (Stability of NMPC with Finite Time Horizon and Terminal Region, [Gra12])
Let the set $\Gamma \subseteq X$ of all initial values x_k, for which the optimal control process with finite time horizon described in (4.12) is solvable, be nonempty. Moreover, let the set $S_\varepsilon = \{x \in \mathbb{R}^{n_x} |\ \varphi(x) \leq \varepsilon\} \subseteq \Gamma$ with $\varepsilon > 0$ be compact and nonempty. Further, let there exist a control law $\mu(x) \in U$ for all $x \in S_\varepsilon$ such that

$$
\frac{\partial \varphi}{\partial x} f(x, \mu(x)) + \ell(x, \mu(x)) \leq 0, \quad \forall\ x \in S_\varepsilon \tag{4.13}
$$

holds. Then the origin of the system (4.1) is asymptotically stable.

The above Theorem 4.4.3 requires the existence of a control law $\mu(x)$ such that the final cost weight $\varphi(x)$ satisfies the inequality (4.13). This inequality is also known as the *control Lyapunov inequality*. A function $\varphi(x)$ that satisfies (4.13) is called a *control Lyapunov function*. In linear MPC, the control Lyapunov inequality can be guaranteed in a relatively straightforward (and even global) way by using the solution of the algebraic Riccati equation to formulate the terminal cost term and using the LQR controller as a feedback law. In the nonlinear case, an additional weighting must be introduced between the terminal cost term and the integral term in the objective functional to ensure the existence of a nonvanishing terminal region S_ε. More approaches to the construction of φ can be found in Chen and Allgöwer [CA98].

Using a final region S_ε instead of a strict equality constraint, the set of initial values for which the optimal control process is solvable has significantly increased. The set should be as large as possible, preferably $S_\varepsilon = X$. This yields the same optimal control problem as in (4.12), but with free final states:

$$
\begin{aligned}
\min_{\bar{u}} \quad & J(\bar{x}_0, \bar{u}) = \varphi(x(T)) + \int_0^T \ell(\bar{x}(\tau), \bar{u}(\tau))\ \mathrm{d}\tau \\
\text{w.r.t.} \quad & \dot{\bar{x}}(\tau) = f(\bar{x}(\tau), \bar{u}(\tau)), \quad \tau \in [0,T], \\
& \bar{x}(0) = x_k, \\
& \bar{x}(\tau) \in X, \quad \bar{u}(\tau) \in U, \quad \tau \in [0,T].
\end{aligned}
\tag{4.14}
$$

Finally, although there are no explicit conditions on the final states in the optimal control problem, the asymptotic stability again depends on the existence of a final region S_ε.

Theorem 4.4.4 (Stability of NMPC with Finite Time Horizon, [Gra12]) Let the set $\Gamma \subseteq X$ of all initial values x_k, for which the optimal control process (4.14) is solvable, be nonempty. Moreover, there exists a compact and nonempty set $S_\varepsilon = \{x \in \mathbb{R}^{n_x} | \ \varphi(x) \leq \varepsilon\} \subseteq \Gamma$ with $\varepsilon > 0$ and a control law $\mu(x) \in U(x)$ for all $x \in S_\varepsilon$ such that

$$\frac{\partial \varphi}{\partial x} f(x, \mu(x)) + \ell(x, \mu(x)) \leq 0, \quad \forall \, x \in S_\varepsilon \tag{4.15}$$

holds. Then the origin of the system (4.1) is asymptotically stable.

Graichen [Gra12] shows that in this case the *region of attraction* is defined as the compact set

$$\Gamma_\alpha = \{x \in \Gamma | \ J(\bar{x}_0 = x, \bar{u}^*) \leq \alpha\}, \quad \text{with } \alpha = \beta \left(1 + \frac{c_\ell}{C_\varphi} T\right). \tag{4.16}$$

Here, the constants c_ℓ, $C_\varphi > 0$ are from our assumptions in (4.6). In particular, it is emphasized that the region of attraction increases with a longer prediction horizon. Accordingly, the set of solvable problems and, in particular, the set of controllable perturbations can be influenced by the choice of the prediction horizon.

To summarize, the theoretical aspects of nominal stability for nonlinear model predictive control are understood very well, and many different NMPC formulations can be used to guarantee the stability of the closed-loop control system. However, many practical problems exist in NMPC, that are the motivations of current research, for example, robust NMPC methods, [GP17; LP14].

4.5. Further Considerations on NMPC

Many advanced challenges and problems, such as the delays caused by computation times, need to be considered in the context of NMPC. In the following sections, we address some of these issues, but it is beyond the scope of this thesis to go into detail.

Length of the Prediction Horizon

We have just observed that a longer prediction horizon can increase the stability of the considered MPC problem. However, with a larger prediction horizon, the computation time required to solve the OCP also increases. Further, when considering problem (4.10), we have shown that stability can be guaranteed for an infinitely long horizon even without a final cost term. Thus, the obvious question is whether asymptotic stability can also be guaranteed without final cost weighting for a sufficiently long horizon.

If a problem formulation without final cost term in the objective functional and without final state constraints is considered, exponential stability can be shown under more restrictive assumptions, [Gra12; GP17]. Among other things, the (local) Lipschitz continuity of the control law is required here. However, it is only ensured that a finite minimum horizon T_{min} exists so that the controlled system is stable for all $T \geq T_{min}$. Nevertheless, in practice, this value is usually yet to be discovered. In particular, the sufficient size of the prediction horizon depends strongly on the system under consideration and is usually determined through simulation.

In general, the lowest possible computation time for the OCP is a crucial point for the applicability of NMPC methods. NMPC formulations with final cost weighting or with a sufficiently long horizon are particularly well suited for implementation in the real process since the numerical effort is usually lower for formulations with a free final state. Moreover, the computational effort is significantly influenced by the length of the prediction horizon and the number of discretization points. Therefore, problem-specific trade-offs between accuracy and performance have to be made.

Computation Times of the OCPs

The presented standard NMPC Algorithm 4.1 assumes that the considered optimal control problems can be solved immediately and that the computed control sequence is instantly applied to the process. In practice, however, it is clear that the computation time will always lead to delays. For relatively slow processes, such as production planning or chemical processes, this can often be feasible without further problems. Special techniques may need to be applied to incorporate computation time for faster dynamics, such as robotics or autonomous vehicles with low response times. Some of the most interesting results in this direction include *real-time iterations* [Die+05], *advanced-step NMPC* [ZB09], and *multilevel iterations* [Kir+10].

The problem of time delay can be illustrated as follows. At any point in time t_k within the NMPC procedure, the initial states $x_k = x(t_k)$ are measured, and an OCP is solved. This requires a time span Δt. At the time $t_k + \Delta t$, the resulting optimal control sequence is applied to the system with the initial states $x(t_k + \Delta t))$ instead of t_k and x_k. The more computing time is spent on solving the OCP, the longer is the delay. The delay generally causes the computed control sequence to be no longer optimal but only suboptimal. In the worst-case scenario, this can lead to infeasibility or even instability. Therefore, strategies for reducing delays or dealing with delays are recommended, especially for fast-sampled systems, La [La16].

- One approach yields to make the solution of the OCP faster by not waiting for the full convergence of the optimization algorithm but performing only a few iterations.

Often, this provides a good approximation of the optimal solution. For this purpose, the solver performs only a few iterations that fulfill certain properties, such as the reduction of the target function, see, for example, the section *nonoptimal NMPC* in [GP17].

- Another concept are *real-time iterations* or *real-time* NMPC, [Die+05]. This is also a Newton-type method for optimization in NMPC. The process is characterized by a parallel solution of the system dynamics and the current solution estimates of the optimizer, resulting in an efficient online optimization algorithm.
- A different concept for dealing with delays is *advanced-step NMPC*. The main idea of this approach is to use the current control action to predict the future state of the plant to solve the OCP in advance as the current sampling period develops, [ZB09]. A similar approach is proposed by Findeisen and Allgöwer in [FA04], where conditions are provided under which the stability of the closed-loop control can be guaranteed.
- Current research is also focused on *multi-level iteration schemes* that extend the idea of real-time iterations, [Kir+10; NAD19; Fra+12]. These algorithms take into account different time scales inherent in the dynamical model by updating the data of the feedback-generating quadratic program, i.e., Hessian and Jacobian matrices, at different levels.
- In addition, there is a close relationship between the successive OCPs. Not only their structure but also their solutions are likely to be similar. Therefore, it makes sense to use the computed solution of an OCP to initialize the next OCP. This could immediately ensure feasibility and a good approximation of the optimal solution. The initialization technique includes so-called *warm-start strategies*, where the entire computed control sequence $u(t_0), u(t_1), \dots, u(t_{N-1})$ becomes the initialization for the next OCP, and *shift-initialization strategies*, where the next OCP is initialized by $u(t_1), u(t_2), \dots, u(t_{N-1}), u(t_{N-1})$. For the shift strategy, the trajectory should be continued meaningfully, e.g., constant or by extrapolation.

In the context of this work, delays originating from OCP computation times are neglected. However, knowing that this problem arises in almost all real applications, we attempt to keep the computation times of the individual optimal control problems low. For this purpose, shift initialization strategies are applied.

Multistep NMPC

The NMPC method described so far requires the solution of the optimal control problem on each sampling interval. However, if it is not achievable to solve the OCP within this time window despite all efforts, a compromise can be provided by the *multistep NMPC* scheme. This approach implements not only one control step of the computed solution but several steps at once. For this purpose, a so-called *control horizon* $T_C \leq T$ is chosen, with $T_C = \kappa \Delta t$ and $\kappa \in \mathbb{N}$, $\kappa \geq 1$. After solving the OCP on the prediction horizon T, the solution is then applied to the system for the duration of the control horizon T_C.

Algorithm 4.2 describes the general procedure of the multistep NMPC scheme. Its basic structure corresponds to the standard NMPC Algorithm 4.1. For $\kappa = 1$ the two algorithms are even identical. The important difference is that in step 6 of Algorithm 4.2 the application interval $[t_k, t_k + T_C]$ of the optimal control is extended. Using terminal constraints in the OCPs, as described in Section 4.4, the corresponding stability proofs can be extended from standard NMPC to multistep procedures, [GP17]. The stability results for problems without stabilizing terminal conditions can also be adapted to the multistep method, [Grü+10].

One advantage of multistep NMPC is that it reduces the number of optimal control problems to be solved compared to the classical NMPC. This allows more time to solve each OCP. The disadvantage of the method is that the controlled system runs in open-loop mode for a longer time. This means that deviations and disturbances cannot be reacted to during this phase. Therefore, multistep NMPC is usually less robust than classical NMPC. Approaches to overcome this drawback are *multistep NMPC with re-optimization* or *multistep NMPC with sensitivity updates*, [YB13; GP15]. Gerdts shows the feasibility of multistep NMPC schemes in a numerical comparison with standard NMPC using a tracking problem for a car model, [Ger18]. The comparison of the performance and robustness of the methods shows that although the multistep schemes have larger tracking errors than the classical NMPC scheme, they are feasible.

We will take advantage of this approach to be able to use NMPC in fast-sampled systems. For this purpose, we apply longer sequences of the computed optimal control of an NMPC step in our numerical examples.

4.6. NMPC with Uncertainties

Our previous considerations in this chapter assumed that we accurately represent the true process with our system model. However, this is hardly the case in real applications since either unknown disturbances can occur, non-modeled interactions are present, or

Algorithm 4.2 Multistep NMPC

```
1: Set time $t_0 = 0$.
2: Choose a prediction horizon $T$ for the OCP and a control horizon $T_C = \kappa \Delta t$, with
   $\kappa \in \mathbb{N}$, $\kappa \geq 1$ and $T_C \leq T$.
3: Set iterator $k = 0$.
4: Measure the current system state $x(t_k)$.
5: Set the initial state of the OCP to $\bar{x}_0^{[k]} = \bar{x}^{[k]}(\tau_0 = 0) := x(t_k)$ and solve 4.9 on
   $\tau \in [0, T]$.
6: Denote the obtained optimal control sequence $\bar{u}^{*,[k]}(\cdot) \in U$ and apply its first $\kappa$
   sequences for the time $[t_k, t_k + T_C]$ as control to the system.
7: Set $k = k + \kappa$ and continue with point 4.
```

the system changes over time. This results in nonlinear perturbed system dynamics, which can be written as

$$\dot{x}(t) = f\left(x(t), u(t), v(t), p\right). \tag{4.17}$$

Here, $v(t)$ denotes disturbances or measurement noise, and p describes parametric uncertainties of the system. The influence of such perturbations often results in a mismatch between the predicted and realized trajectories. The question is whether we can still reach our control objective. There are further unknowns. To what extent are the previous stability statements still valid? How can the used OCPs be formulated to generate closed-loop stability further?

Answers to these questions are classically provided by concepts in the areas of *robust NMPC* and *stochastic NMPC*. These offer a systematic investigation of the effects of certain disturbance and uncertainty classes. In the robust case, the perturbations and parameters in (4.17) are usually assumed to lie in compact sets $\mathcal{P}, \mathcal{V}$: $p \in \mathcal{P} \subseteq \mathbb{R}^{n_p}$, $v(t) \in \mathcal{V} \subseteq \mathbb{R}^{n_x}$. In contrast, stochastic approaches assume the disturbances and parameters to be random variables with certain probability distributions, i.e., $v(t) \sim \mathbb{P}^v$, $p \sim \mathbb{P}^p$. Typically, instead of the problem formulation (4.2) then a constrained *stochastic optimal control problem* (SOCP) is considered:

$$\begin{aligned}
\min_{x,u} \quad & \mathbb{E}_{x_0}\left[\varphi(x(t_f)) + \int_{t_0}^{t_f} \ell(x(t), u(t))\ \mathrm{d}t\right] \\
\text{w.r.t.} \quad & \dot{x}(t) = f(x(t), u(t), v(t), p), \quad t_0 \leq t \leq t_f, \\
& x(t_0) = x_0, \\
& u(t) \in U, \quad t_0 \leq t \leq t_f, \\
& v(t) \sim \mathbb{P}^v, \quad p \sim \mathbb{P}^p, \\
& \mathrm{Pr}_{x_0}[c_i(x(t), u(t)) \leq 0] \geq \beta_i, \qquad \text{for } t_0 \leq t \leq t_f \text{ and all } i = 1, \dots, n_c,
\end{aligned} \tag{4.18}$$

where $\mathbb{E}_{x_0}[\cdot]$ denotes the *conditional expectation* and $\mathrm{Pr}_{x_0}[\cdot]$ the *conditional probability*. Both expressions indicate the dependence on the initial state $x(t_0) = x_0$. The objective of (4.18) is the minimization of the expectation value of the cost function, which depends on the solution of the perturbed dynamical system in the constraints. General constraints are typically defined in the form of *chance constraints* over the prediction horizon

$$\mathrm{Pr}_{x(t_0)}[c_i(x(t), u(t)) \leq 0] \geq \beta_i.$$

With these, for given lower bounds $\beta_i \in \mathbb{R}$ for $i = 1, \dots, n_c$, it is required that each constraint $c_i(x(t), u(t)) \leq 0$, $i = 1, \dots, n_c$, must be fulfilled for all $t_0 \leq t \leq t_f$ with at least this probability β_i, [Mes16].

The following subsections briefly mention some prominent concepts of robust NMPC and stochastic MPC as related work. Furthermore, we introduce some new approaches from the more recent field of *learning-based NMPC*, [Hew+20]. There are several advanced ideas on how to deal with uncertainties and to adapt the controller. In particular, we discuss so-called *adaptive NMPC* schemes in Subsection 4.6.2.

4.6.1. Robust NMPC

Considering the nonlinear perturbed dynamical system described in (4.17), robust MPC methods aim to analyze all possible solutions of a disturbed optimal control problem in the MPC setting. For nonlinear MPC, the analysis of robustness properties is an ongoing research area. In Findeisen and Allgöwer [FA02] three different robust NMPC approaches that directly consider uncertainties are presented.

One is robust NMPC based on open-loop min-max problems. In this formulation, the standard NMPC setup is maintained, but the cost function to be minimized is formulated to include the worst-case perturbation. This results in a min-max problem. One disadvantage is that additional stability constraints may result in not finding a feasible solution. Another approach is *H_∞-NMPC*. For this, the standard H_∞-problem[2] is considered in the context of NMPC. However, solving the infinite horizon min-max problem can be very time-consuming. The third method is a robust NMPC scheme that optimizes a feedback control law instead of the input signals. In this way, it can be achieved that between the sampling points, feedback enables the controller to react to disturbances. However, the complexity of the computation is often unacceptable, Magni et al. [Mag+02]. We refer to [FA02; May14; HV19] for more detailed information. A comprehensive overview of stochastic MPC gives Mesbah [Mes16; Mes+14].

The classical robust and stochastic NMPC methods belong to offline methods, i.e., there is a strict separation between the controller design phase and the application phase of the control. In this work, we are particularly interested in adapting controllers during operation. Therefore, we want to focus on online learning methods. This involves adjusting the controller based on collected data during operation.

4.6.2. Learning-Based and Adaptive NMPC

The research areas of *learning-based*, *data-based NMPC*, and *adaptive NMPC* contain many different concepts. We can only address a few of them here. A more detailed overview of the existing methods is given by Hewing et al. [Hew+20]. The field of learning-based NMPC is divided into three main areas: 1) the goal of model learning, 2) controller-parameter learning, and 3) NMPC with safe learning.

1) The first one aims to improve model quality automatically. This focuses on learning the system model during online operation. Since the performance of NMPC strongly depends on a sufficiently accurate model representation, these methods aim at learning the system dynamics from data during operation or between two

[2] H_∞-control is a robust control technique. It is based on uncertainty modeling, resulting in an extended transfer function used to compute the H_∞-controller. The term H_∞ refers to a special vector norm in the Hardy function space.

instances when performing repetitive tasks. In particular, the optimal control problems to be solved in the NMPC process are modified, and thus the predictions are improved.

2) The second approach attempts to address the parameterization of the NMPC controller directly. In this context, attention is paid less to the prediction model and more to the remaining configurations, such as the cost function, the constraints, or the prediction horizon, and the overall behavior of the MPC controller is optimized. There are numerous research activities on directly learning the MPC controller, e.g., [LK18] aim to learn the robust NMPC policy employing deep neural networks, and Hertneck et al. [Her+18] propose a neural network-based NMPC framework with guarantees of stability and constraint satisfaction.
3) The third research area deals with NMPC as a complement to learning-based control methods to achieve safety during operation and learning, [Kol+18; Wab+21]. In this case, the optimization of the objective function is separated from the condition of constraint satisfaction using MPC techniques.

In this thesis, the focus is directed to the first field, the learning of the system model. Therefore, it is described further. Much research has been undertaken in this area for linear system models, where results for controller stability for certain problem classes exist, [LCA19]. In Aswani et al. [Asw+13], a learning MPC algorithm based on statistical learning techniques is presented that uses linear approximate models of the system along with a bound on the uncertainty to improve controller performance. Further, adaptive model predictive control algorithms for linear, time-invariant systems with constraints are presented in [Tan+14; BZB18; Ter+19].

However, we are most interested in nonlinear models. Research in nonlinear model learning in NMPC is further divided into parametric and nonparametric approaches. The first field aims to determine the parameter values from the data so that the model output matches the observations as closely as possible. In contrast, nonparametric concepts form their estimate of the model function directly from the measurements. These are often based on neural networks like in [Sha08; Hed13]. There is also recent research in the area of *data-driven MPC*, which builds on the fundamental lemma of Willems [Wil+05]. The lemma provides a way to parameterize all trajectories of a linear time-invariant system based on a data trajectory with persistently exciting input component [Ber+22]. A model-free, data-based approach is used to design an optimal predictive controller, and current research is addressing the formulation of stability statements for different linear system formulations, [SFW22; Sch+22; Bil+22; Hua+21].

Our interest is in parametric modeling approaches. The works on parameter learning methods by Mayne and Michalska [MM93] and Adetola et al. [ADG09] estimate nonlinear system models but are linear in parameters. Advanced studies on online learning of nonlinear parametric models are the subject of current research. Valluru and Patwardhan [VP19] develop a controller combining integrated frequent real-time optimization and adaptive NMPC. There, a common dynamic model based on state and parameter estimation is used to update the steady-state model for real-time optimizations and the dynamical model used for predictions in NMPC. Hanover et al. [Han+22] propose a hybrid adaptive NMPC to learn and compensate for model uncertainties online and demonstrate its performance on a quadrocopter. They expect future work to include online updates of the dynamical model.

From a broader perspective, there is also related research in the area of iterative learning control, see Cuelei and Bordons [CB08]. Furthermore, Recht [Rec19] provides in his review on reinforcement learning a brief discussion of learning an MPC controller in this context. A further challenge in online learning is the question of sufficient data and the resulting trade-off between information acquisition and optimal performance. This issue is considered in the research on *dual NMPC*, [Mes18; La16]. We will briefly discuss ideas on dual NMPC in the outline in Section 7.2.

The proposed approach for an adaptive NMPC described in Chapter 6 of this thesis can be classified in the area of parametric model-based NMPC, where the aim is to learn the nonlinear system dynamics by nonlinear parameter identification.

5. Online Parameter Estimates for Linear Optimal Feedback Control

In this chapter, we present an approach that extends the method of sensitivity-based updates for LQR from Subsection 3.3.2 using online identification for parametric perturbations. The approach of reacting to parametric perturbations with sensitivity-based updates for the feedback gain of LQR was analyzed in detail by Tietjen [Tie12]. We will refer to this method as *adaptive LQR* in the following and abbreviate it as `adLQR`. Tietjen also shows in his thesis on two simulative examples that the performance of `adLQR` is similar to a real recalculation of the feedback gain. However, previous research using the sensitivity-based updates for LQR has always assumed that the parametric perturbations $\Delta p = p - p_0$, i.e., the deviations from the initial assumed values p_0, are known. This condition can be fulfilled, for example, if the parameters are controller settings, more precisely, if they are associated with changes in the matrices $Q(p)$ or $R(p)$. Also, parametric disturbances in easily measurable quantities, such as the initial state, for example, the position of a car, can be determined directly. However, if the parametric changes appear as part of a complex system model, they usually cannot be accessed directly. Therefore, within this thesis, a new approach is developed to perform data-based parameter identification online to determine these unknown changes during operation. Preliminary numerical tests of this method have shown promising results, see [RFB21]. The procedure is illustrated in Figure 5.1.

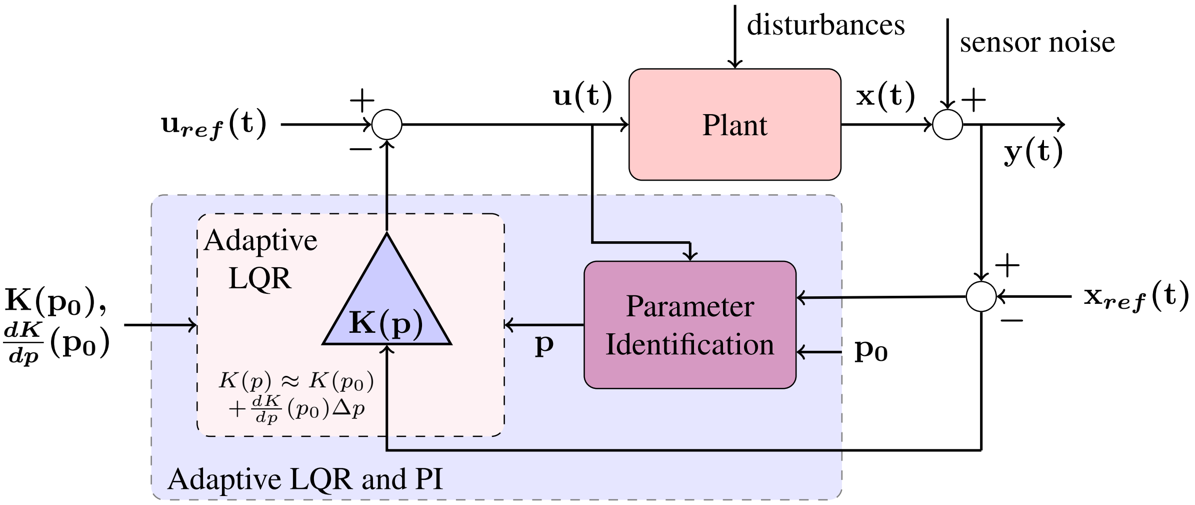

Figure 5.1.: *The control loop with adaptive LQR and online parameter identification.*

In contrast to the closed loop LQ regulator introduced in Section 3.2, a parameter identification process is integrated. It is shown as an extra red block in the bottom right-hand corner. This enables us to use measured data sampled from the output $y(t)$ to compute estimates for model parameters of the plant under consideration. These estimates are then provided to the controller to calculate an approximation of the feedback gain $K(p)$. This is done using the initial gain matrix $K(p_0)$ and the tensor of sensitivities that are previously computed offline. The depicted disturbances are internal and external influences that cause the model parameters to change. An additive noise in the measurements accounts for disturbances originating from sensors.

The process starts by computing the feedback control gain $K(p_0)$ and the sensitivities $\frac{dK}{dp}(p_0)$ for the initial parameters p_0 and given reference states x_{ref} and control u_{ref} in a first step. This is done offline by solving the Problem 3.18. Now, we can use this initial gain $K(p_0)$ to control the system during execution. Since we expect the parameters p to change or to be perturbed, we use the real-time approximation method `adLQR` to control the system. This is

$$u(t) = -\left(K(p_0) + \frac{dK}{dp}(p_0)\Delta p\right) x(t), \quad \text{with } \Delta p := p - p_0. \tag{5.1}$$

During the operation of the plant, measurements y are taken. We assume here full-state feedback, so all system states can be measured. The measured values then correspond to the states plus an unknown measurement noise

$$y(t) = Cx(t) + v(t), \quad \text{with } v(t) \in \mathbb{R}^{n_x}. \tag{5.2}$$

For full-state feedback, the matrix C in equation is the identity matrix. The measurements are used for the real-time approximations of the control law. Additionally, they are also fed to the online parameter identification. The samples are collected at equidistant discrete time points t_i, $i = 0, 1, \dots, M-1$, with $M \in \mathbb{N}$. The difference between two time points is called the *sample time* Δt. For each execution, the parameter identification is performed on an equally-sized *data sampling horizon* T_s. The identification procedure is performed periodically with a time interval of T_{PI} to identify changes $\Delta p := p - p_0$ in the parameter. Subsequently, the updated parameters p are provided for the adaptation process of the linear quadratic controller.

The rest of the chapter is organized as follows. First, we introduce how the identification problem in this LQR environment is formulated as a nonlinear optimization problem. Then, we formalize the algorithm that combines the adaptive linear quadratic controller, `adLQR`, with this nonlinear dynamic parameter identification problem. In the following, we will refer to this combined method as `adLQR`+`PI`. Additionally, we will present an extension of the algorithm in which the *Fisher information matrix* is used to assess the information content of the data. The chapter concludes with a numerical investigation of the presented method using the example of an inverted pendulum on a cart.

5.1. Formulation of NDPIP for `adLQR+PI`

We start with the formulation of the parameter identification problem for the combined algorithm `adLQR+PI`. Assume that we are performing parameter identification for the k-th time, with $k \in \mathbb{N}_0$, and that we have $M \in \mathbb{N}$ measurements $y_i^{[k]} \in \mathbb{R}^{n_x}$, $i = 0, \dots, M-1$, where n_x is the number of system states. The superscript here indicates the affiliation to the k-th PI problem. The subscript numbers are all the samples that belong to this problem. The current time is $t_{M-1}^{[k]}$. Thus, we identify the parameters for the k-th time interval as

$$I^{[k]} = [t_0^{[k]}, t_{M-1}^{[k]}] \subset [0, T],$$

which extends from the current time by a duration of T_s into the past and includes M measurements at discrete time points $t_0^{[k]} < t_1^{[k]} < \cdots < t_{M-1}^{[k]} \in I^{[k]}$. Let the set of measurements be

$$\{y\}^{[k]} := \{y_0^{[k]}, \dots, y_{M-1}^{[k]}\}.$$

These measurements correspond to our system states x (plus an additional measurement noise) at these time points, (5.2). Therefore, we can use the measurements directly for the objective function of our identification problem. This we define as

$$F(x_0^{[k]}, p^{[k]}) := \frac{1}{2M} \sum_{i=0}^{M-1} w_i^{[k]} \left\| x(t_i^{[k]}; x_0^{[k]}, p^{[k]}) - y_i^{[k]} \right\|_2^2. \tag{5.3}$$

Each summand of the objective function is weighted by a factor $w_i^{[k]}$, which can be chosen by the user. In the simplest case, the weights are $w_i^{[k]} = 1$ for all $i = 0, \dots, M-1$. However, these weights can also be used to reflect the reliability of individual measurements if, for example, the stochastic distribution of the measurements is known. In the same way, we assume in the numerical simulations in Section 5.4 that the measurements are normally distributed and that we know this distribution. Therefore, their variance is used for weighting to indicate how reliable the included measurements are. To formulate the NDPIP according to Definition 2.3.2 we add the nonlinear dynamical system of the considered process, including the unknown parameters as a constraint. We obtain the following identification problem.

$$\begin{aligned} \min_{x_0^{[k]}, p^{[k]}} \quad & F(x_0^{[k]}, p^{[k]}) = \frac{1}{2M} \sum_{i=0}^{M-1} w_i^{[k]} \left\| x(t_i^{[k]}; x_0^{[k]}, p^{[k]}) - y_i^{[k]} \right\|_2^2 \\ \text{w.r.t.} \quad & \dot{x}(t) = f(x(t), \tilde{u}(t), p^{[k]}, t), \\ & x(t_0^{[k]}) = x_0^{[k]}, \\ & g(x(t), p^{[k]}) \leq 0. \end{aligned} \tag{5.4}$$

Within the constraints, we insert the measured controls that were used to generate the data under consideration. However, since these are only available to us at the discrete times of the measurements, we need to approximate the control in between. For this purpose, we denote the approximation of the control u by $\tilde{u}$. It is equal to the measured control at each sample time point, and in between two samples, it is the result of a linear interpolation of the two adjacent points.

5.1.1. Alternative Approaches

For certain applications, the question might arise whether it is sufficient to identify only the linearized model $\dot{x}(t) = A(p)x(t) + B(p)u(t)$. This may be the case for simple dynamics, but in this work, we aim to develop an approach that is generally applicable even for highly nonlinear systems. Nevertheless, it may be of interest to the user to determine the parameters for the linearized model since this is used directly in the linear control laws. Thus, the identified system matrices can be used in other methods. The corresponding identification problem would have the following form:

$$
\begin{aligned}
\min_{x_0^{[k]},p^{[k]}} \quad & F(x_0^{[k]},p^{[k]}) = \frac{1}{2M}\sum_{i=0}^{M-1} w_i^{[k]} \left\| x(t_i^{[k]};x_0^{[k]},p^{[k]}) - y_i^{[k]} \right\|_2^2 \\
\text{w.r.t.} \quad & \dot{x}(t) = A(p^{[k]})x(t) + B(p^{[k]})\tilde{u}(t), \\
& x(t_0^{[k]}) = x_0^{[k]}.
\end{aligned}
\tag{5.5}
$$

It is expected that a comparison of the true parameters and the parameters determined by problem (5.5) is no longer possible or reasonable in the direct quantitative range. This stems, in particular, from the fact that the linearization error also enters into the result of the parameter identification.

We briefly present another approach, which was investigated in our research. It results from approaching the problem mainly from a control perspective. The idea here is to use the feedback law directly, in contrast to the first approaches where we treat the controls $u(t)$ as fixed values on the already passed time interval $I^{[k]}$. Previously, the controls had entered the NDPIP as the fixed measured values $\tilde{u}(t)$ determined by the feedback law $u(t) = -\big(K(p_0) + \frac{dK}{dp}(p_0)\Delta p^{[k]}\big)y^{[k]}(t)$. Here, instead of using the measured states $y^{[k]}$, we use the optimization variables x that are to be identified.

$$
\begin{aligned}
\min_{x_0^{[k]},p^{[k]}} \quad & F(x_0^{[k]},p^{[k]}) = \frac{1}{2M}\sum_{i=0}^{M-1} w_i^{[k]} \left\| x(t_i^{[k]};p^{[k]}) - y_i^{[k]} \right\|_2^2 \\
\text{w.r.t.} \quad & \dot{x}(t) = f(x(t), \underbrace{(K(p_0) + \frac{dK}{dp}(p_0)\Delta p^{[k]})x(t))}_{=u(t)}, p^{[k]}, t), \\
& x(t_0^{[k]}) = x_0^{[k]}.
\end{aligned}
\tag{5.6}
$$

Algorithm 5.1 `adLQR+PI`

```
Choose the start time $t_0$, initial states $x_0$, reference states $x_{ref}$, and controls $u_{ref}$.
Choose the nominal parameters $p^{[0]} = p_0$, data sampling horizon $T_s$ and frequency
    of parameter identification $T_{PI}$.
Compute offline $K(p_0)$ and $\frac{dK}{dp}(p_0)$ by solving the LQR problem (3.18).
Execute the system for $t \in I^{[0]} = [t_0, t_0 + T_s]$ with the control law

    $u(t) = -\big(K(p_0) + \frac{dK}{dp}(p_0)(p^{[0]} - p_0)\big)\big(y(t) - x_{ref}(t)\big) + u_{ref}(t).$

Collect measurements $\{y\}^{[0]}$ with sample time $\Delta t$.
Set $(x_0^{[1]}, p^{[1]}) \leftarrow$ *argmin* of NDPIP in (5.4) with $k = 0$.
for $k = 1, 2, \dots$ do
    Execute the system for a time period of $T_{PI}$ with the control law

        $u(t) = -\big(K(p_0) + \frac{dK}{dp}(p_0)(p^{[k]} - p_0)\big)\big(y(t) - x_{ref}(t)\big) + u_{ref}(t).$

    Take measurements $\{y\}^{[k]}$ with sample time $\Delta t$.
    Set $(x_0^{[k+1]}, p^{[k+1]}) \leftarrow$ *argmin* of NDPIP (5.4).
end for
```

This approach (5.6) seems unorthodox from the point of view of system identification. However, it has its justification if the aim is exclusively to identify a good controller and not necessarily to deal with true system parameters. In this work, we assume that we have a relatively good model of our system under consideration and that we not only want to control it optimally but also to improve the model further. Therefore, we decided to use the first problem formulation (5.4).

5.2. The Algorithm `adLQR+PI`

At this point, we formulate the combined method of the adaptive linear quadratic regulator with online parameter estimation denoted as `adLQR+PI` in Algorithm 5.1. We initialize the procedure by choosing an initial time $t_0 \in \mathbb{R}$, initial states $x_0 \in \mathbb{R}^{n_x}$, and reference states and controls $x_{ref} \in \mathbb{R}^{n_x}$ and $u_{ref} \in \mathbb{R}^{n_u}$. For the parameter identification, we specify a initial guess $p^{[0]} = p_0 \in \mathbb{R}^{n_p}$, the data sampling horizon $T_s \in \mathbb{R}$, and the value $T_{PI} \in \mathbb{R}$, i.e., the frequency with which an NDPIP is performed. Then, we start with the preprocessing phase. In this first phase, the offline computations of the feedback gain and the tensor of sensitivities are performed, and initial measurements for the update process are collected. This is because no measurements are available at the start to identify parameters. Then, as a next step, the classical parameter-dependent

LQR problem is solved offline using the initial parameters p_0. We thereby obtain the feedback gain $K(p_0)$ and the sensitivities $\frac{dK}{dp}(p_0)$. Combining (3.10) and (5.1) results in the following control law:

$$u(t) = -\left(K(p_0) + \frac{dK}{dp}(p_0)(p^{[0]} - p_0)\right)(y(t) - x_{ref}(t)) + u_{ref}(t).$$

This is then applied to the system under consideration for an initial time interval of length T_s. In our setting for the numerical tests, we assume the parameters to be equal to the initial guess at the beginning. It is $p^{[0]} - p_0 = 0$, so that no approximation of the gain is done and the control law is,

$$u(t) = -K(p_0)\,(y(t) - x_{ref}(t)) + u_{ref}(t).$$

In the first sampling horizon, measurements $y_i^{[0]}$ are collected at each sampling time $t_i^{[0]} \in [t_0, t_0 + T_s]$, $i = 0, 1, \ldots, M-1$ with an equidistant sample time $\Delta t = t_{i+1} - t_i$. The sample time Δt is specified by the system, for example, by the performance of the sensors. For the data set $\{y\}^{[0]} = \{y_0^{[0]}, y_1^{[0]}, \ldots, y_{M-1}^{[0]}\}$ the first parameter identification can now be performed. The result is used as a new parameter estimate $p^{[1]}$ and the preprocessing phase is completed with step 4.

The feedback gain is now updated to $K(p_0) + \frac{dK}{dp}(p_0)(p^{[k]} - p_0)$ as described in (5.1). The system is further controlled with the adapted control law

$$u(t) = -\left(K(p_0) + \frac{dK}{dp}(p_0)(p^{[k]} - p_0)\right)(y(t) - x_{ref}(t)) + u_{ref}(t).$$

This is done over a time period of T_{PI}, the fixed interval between two parameter identifications. Again, measurements are collected with the sample time Δt. The data collected during this time is added to the previous dataset $\{y\}^{[k-1]}$. For this, a corresponding number of measurements at the front of the set are sorted out to obtain the dataset $\{y\}^{[k]}$ that will be used in the next NDPIP. Thus, a consistently large time horizon T_s is always considered for the parameter identification. More precisely the interval $I^{[k]} = [t_0^{[k]}, t_{M-1}^{[k]}] = [t_0^{[k-1]} + T_{PI}, t_{M-1}^{[k-1]} + T_{PI}]$ is used. The next step in Algorithm 5.1 is to update the parameter estimate with the solution of the NDPIP. After that, the new parameter $p^{[k]}$ enters the feedback law again in step 8 to adapt it.

We assume here that for $p = p_0$ the original problem 3.15 has a unique solution $K(p_0)$. This is the case if the assumptions on the observability of $(A(p_0), \sqrt{Q(p_0)})$ and the stabilizability of $(A(p_0), B(p_0))$ from Theorem 3.2.5 are satisfied. For a perturbation $\Delta p = p - p_0$, the Theorem 3.3.2 predicts under the conditions on the differentiability of the system matrices $A(p)$, $B(p)$, $Q(p)$, and $R(p)$ the existence of a neighborhood $U(p_0)$ in which the solution is differentiable with respect to p as well. This means that the rank of the controllability and observability matrices is also unchanged. The optimal feedback law $u = -K(p)x(t)$ can then be approximated within this neighborhood by using

$$K(p) \approx K(p_0) + \frac{dK}{dp}(p_0)\Delta p.$$

However, since no statements can be made about the size of the neighborhood, we must ensure that the perturbed parameters p do not change the structure of our initial problem. In particular, we need to ensure that the controllability and observability matrices still have full rank, i.e.,

$$\operatorname{rank}\begin{pmatrix} B(p) & A(p)B(p) & \dots & A(p)^{n-1}B(p) \end{pmatrix} = n_x$$

and

$$\operatorname{rank}\begin{pmatrix} \sqrt{Q(p)} \\ \sqrt{Q(p)}A(p) \\ \vdots \\ \sqrt{Q(p)}A(p)^{n-1} \end{pmatrix} = n_x.$$

A range of acceptable perturbations can be determined in advance by appropriate simulations. For this purpose, numerical tests can be done by choosing different parametric perturbations and analyzing the effect on the rank of the controllability and observability matrices. If the chosen acceptable range is left during the process, recalculation of the control law may be recommended. We limit the parameters in the identification problem in the constraint function g with lower and upper bounds by this range.

Remark. In the following, it is assumed that parametric perturbations do not influence the structure of the initial problem, and, in particular, the rank of the above matrices remains unchanged.

5.3. Extension of Algorithm `adLQR+PI`

In this section, we make additional assumptions on the problem that allows us to add an extension to the previously presented Algorithm 5.1. This additional option uses the *Fisher information matrix (FIM)* to evaluate the information content of the collected data and estimates the probability of a successful parameter identification. Based on this, it is then decided whether a parameter identification is performed. For this purpose, the measured data is assumed to be a random variable, and its probability distribution is known. Furthermore, it is assumed that the individual measurements are independent. We will denote the Fisher information matrix as $\mathcal{I}_p$ in the following.

5.3.1. Fisher Information Matrix and Maximum Likelihood Estimation

In this subsection, we briefly present the theory for the FIM and how we adapt it for our use. More detailed information is provided, for example, in Bar-Shalom et al. [BLK01].

The Fisher information matrix is a statistical approach to measure how much information some observed data of a random variable contains about some unknown parameters involved in the underlying model of the random variable. In our case, the measurements are the realizations of this random variable. Considering the parametric control system from Definition 3.1.2, it is assumed that the probability distribution of the independent measurements is normally distributed. Thus, the measurement noise $v \sim \mathcal{N}(0, \Sigma)$ is added to the system output. With $\Sigma \in \mathbb{R}^{n_x \times n_x}$ we denote the covariance matrix of the measurements. The system is

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t), \tilde{p}, t), \quad x(t_0) = x_0 \in \mathbb{R}^{n_x}, \\ y(t) &= g(x(t; \tilde{p}), u(t), \tilde{p}, t) + v(t), \end{aligned} \tag{5.7}$$

where the initial states are given by $x_0 = (x_{0_1}, \dots, x_{0_{n_x}})^T \in \mathbb{R}^{n_x}$. The parameters that are needed to fully specify the system (5.7) are the unknown model parameters $\tilde{p}$ and the initial system states x_0. The parameter vector is defined as $p := (\tilde{p}, x_0)^T \in \mathbb{R}^{n_p + n_x}$. Now, for a collected series of measured data $\{\tilde{y}_i\}_{i=0,\dots,M-1}$ at time points t_i the parameters p are to be found such that the occurrence of these measurements is most probable. Since we assume to know the error distribution in the measurements, the *maximum likelihood method* can be used. For this purpose, the *log-likelihood function* $l(p; \tilde{y})$ is defined. It indicates how likely the parameters are given the measured data. By maximizing the log-likelihood function, the parameters p^* are determined that are most likely to produce the observed data. The solution parameters p^* are called *maximum likelihood estimator*. The associated optimization problem is formulated as

$$\begin{aligned} \max_p \quad & l(p; \tilde{y}) := -\sum_{i=0}^{M-1} (y(t_i; p) - \tilde{y}_i)^T \Sigma^{-1} (y(t_i; p) - \tilde{y}_i) \\ \text{w.r.t. } & \dot{x}(t) = f(x(t), \tilde{u}(t), p), \\ & y(t) = g(x(t; p), \tilde{u}(t), p) + v(t). \end{aligned} \tag{5.8}$$

For the problem defined in (5.8), the input signals $\tilde{u}(t)$ are the fixed controls that were applied to the system when obtaining the measurements $\tilde{y}$. To achieve the well-known standard minimization problem used in this work, we reformulate (5.8) as

$$\begin{aligned} \min_p \quad & F(p) := \sum_{i=0}^{M-1} (y(t_i; p) - \tilde{y}_i)^T \Sigma^{-1} (y(t_i; p) - \tilde{y}_i) \\ \text{w.r.t. } & \dot{x}(t) = f(x(t), \tilde{u}(t), p), \\ & y(t) = g(x(t; p), \tilde{u}(t), p) + v(t). \end{aligned} \tag{5.9}$$

This problem can now be solved in the same way by utilizing full discretization, as described in Section 2.4.

Considering the parameter identification problem as a maximum likelihood problem opens up new possibilities from the existing theory of such problems. One concept uses the *Fisher information matrix (FIM)*, which is defined as

$$\mathcal{I}_p := \sum_{i=0}^{M-1} \left(\frac{d}{dp} y(t_i, p) \right)^T \Sigma^{-1} \left(\frac{d}{dp} y(t_i, p) \right). \tag{5.10}$$

The FIM can be understood as a quantification of the maximum information available in the data about a parameter. This quantity is used in the *Cramèr-Rao inequality* to give a lower bound for the covariance of the maximum likelihood estimator p^*:

$$\mathrm{Cov}(p^*) \geq \mathcal{I}_p^{-1}. \tag{5.11}$$

The covariance indicates the estimator's accuracy and thus provides a measure of how good the estimate can be for the best case with the available data [BLK01]. Thus, we can use the Fisher information matrix to make a statement about whether a measurement series contains enough information for accurate parameter estimation. The terms $(\frac{d}{dp} y(t_i, p))$ in (5.10) denote the partial derivatives of the states according to the parameters. These cannot directly be derived analytically. They have to be computed numerically, e.g., with forward sensitivities, see Stapor et al. [SFH18].

Regarding the FIM, we want to make some remarks that should be taken into account when using it. We assumed that the model equations exactly describe the true system, which is generally difficult to attain in real-world applications. As a result, if the given model is not sufficiently good, an accurate result cannot be expected from the computation of the FIM either. Furthermore, the statistics are only valid if the FIM is evaluated at the true parameter values. In general, these are the values we are looking for, so the result of our FIM estimation depends on the current estimation of the parameters. In particular, the validity of the FIM relies heavily on the initial estimate of the parameters since the statements are local.

Also, it is worth mentioning that the FIM grows very fast and is very sensitive to numerical errors. However, since we keep recalculating it on smaller subsections, this impact is less severe in our case. Nevertheless, the analysis of the FIM is a good tool to make a profound statement about the information content of data for parameter estimation. In particular, it can help to decide whether measurements are not suitable for a reliable approximation of the parameters. We will examine this further in the subsequent sections.

5.3.2. The Algorithm `adLQR+PI+FIM`

In order to benefit from the Fisher information matrix within our algorithm, we assume a measurement noise with $v \sim \mathcal{N}(0, \Sigma)$ and that the output data is a normally distributed random variable. We also assume that the individual measurements are independent of each other. Further, we slightly modify our objective function in the parameter identification problem. For system (5.7) with full-state feedback, i.e., $g \equiv \mathrm{id}$, and the covariance matrix $\Sigma \in \mathbb{R}^{n_x \times n_x}$, we obtain an objective function similar to the maximum likelihood one:

$$F_{\mathrm{MHE}}(x, p^{[k]}) = \frac{1}{2M} \sum_{i=0}^{M-1} w_i^{[k]} (x(t_i^{[k]}; x_0^{[k]}, p^{[k]}) - y_i^{[k]})^T \Sigma^{-1} (x(t_i^{[k]}; x_0^{[k]}, p^{[k]}) - y_i^{[k]}). \tag{5.12}$$

This approach still includes our original problem formulation 5.4. It can immediately be obtained by setting the covariance matrix to the identity matrix, $\Sigma = I \in \mathbb{R}^{n_x \times n_x}$.

Algorithm 5.2 now supplements the procedure of the previous Algorithm 5.1 with computations of the FIM. The FIM is always estimated before solving an NDPIP, see steps 6 and 14.

Algorithm 5.2 `adLQR+PI+FIM`

```
 1: Choose start time $t_0$, initial states $x_0$, reference states $x_{ref}$, and controls $u_{ref}$.
 2: Choose nominal parameter $p^{[0]}$, data sampling horizon $T_s$, frequency of parameter
    identification $T_{PI}$ and threshold $\delta_{\mathrm{FIM}}$.
 3: Compute offline $K(p_0)$ and $\frac{dK}{dp}(p_0)$ by solving the LQR problem (3.18).
 4: Execute the system for $t \in I^{[0]} = [t_0, t_0 + T_s]$ with the control law
        $u(t) = -\big(K(p_0) + \frac{dK}{dp}(p_0)(p^{[0]} - p_0)\big)\big(y(t) - x_{ref}(t)\big) + u_{ref}(t).$
 5: Collect measurements $\{y\}^{[0]}$ with sample time $\Delta t$.
 6: Compute Fisher information matrix $\mathcal{I}_p$.
 7: if tr$(\mathcal{I}_p^{-1}) > \delta_{\mathrm{FIM}}$ then
 8:     Set $(x_0^{[1]}, p^{[1]}) \leftarrow$ argmin of NDPIP in (5.4) with objective as in (5.12).
 9: else Set $p^{[1]} \leftarrow p^{[0]}$.
10: end if
11: for $k = 1, 2, \ldots$ do
12:     Execute the system for a time period of $T_{PI}$ with the control law
            $u(t) = -\big(K(p_0) + \frac{dK}{dp}(p_0)(p^{[k]} - p_0)\big)\big(y(t) - x_{ref}(t)\big) + u_{ref}(t).$
13:     Collect measurements $\{y\}^{[k]}$ with sample time $\Delta t$.
14:     Compute Fisher information $\mathcal{I}_p$.
15:     if tr$(\mathcal{I}_p^{-1}) > \delta_{\mathrm{FIM}}$ then
16:         Set $(x_0^{[k+1]}, p^{[k+1]}) \leftarrow$ argmin of NDPIP (5.4) with objective as in (5.12).
17:     else Set $p^{[k+1]} \leftarrow p^{[k]}$.
18:     end if
19: end for
```

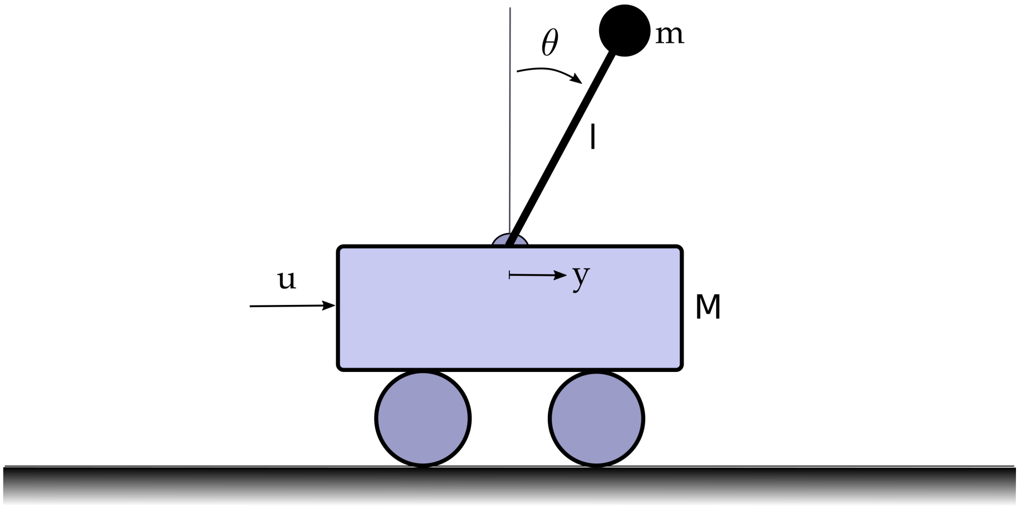

Figure 5.2.: *A schematic illustration of an inverted pendulum on a cart.*

It is then checked whether the trace of the FIM exceeds a certain predefined threshold $\delta_{\text{FIM}} \in \mathbb{R}$:

$$\operatorname{tr}(\mathcal{I}_p^{-1}) := \sum_{i=1}^{n_p} (\mathcal{I}_p^{-1})_{ii} > \delta_{\text{FIM}}.$$

This criterion is referred to as *A-optimality criterion* and is used to constrain the estimated average covariance of the parameter estimate. More information on A-optimality and other alternative criteria can be found in Atkinson et al. [ADT07]. The value δ_{FIM} is initially selected by the user according to the specific problem. It depends on the magnitude of the parameters and the measurement noise. If the information content is considered insufficient for parameter estimation, i.e., the trace of the inverse of the FIM is larger than the threshold, no identification is performed. Instead, the current parameter estimate is retained, compare step 17.

5.4. Numerical Analysis on the Inverted Pendulum on a Cart

We use an inverted pendulum on a cart, as depicted in Figure 5.2 as an example to illustrate the suggested control method. First, we introduce the pendulum-cart system and discuss the simulation setting. After that, we formulate the parameter-dependent LQR problem for this example. Next, the assumptions of the approximation theorem for parametric LQR problems 3.3.2 are verified. Then, for a nominal perturbation p_0, we compute the feedback gain $K(p_0)$ and the parametric sensitivities $\frac{dK}{dp}(p_0)$ offline. In the main part, we simulate the system controlled by the combined `adLQR+PI` method. We will consider different experimental scenarios, where we compare our control approach to the classical `LQR` with feedback gain $K(p_0)$ and the best possible linear-quadratic controller `rLQR` with $K(p^*)$. After each simulation interval, a new optimal feedback gain $K(p^*)$ is computed, knowing the true parameters. It is denoted by `rLQR` for repeated LQR in the following. Note that this optimal control strategy is used as a comparison in the present simulation but is usually not available in real-world applications because the parametric perturbations are commonly unknown, and it requires a large computational effort. In addition, we consider the optimal sensitivity-based adaptive LQR with the updated gain $K(p_0) + \frac{dK}{dp}(p_0)\Delta p^*$ as a further comparison. This is denoted by `adLQR` in the following and uses the true values Δp^* instead of the identified parameter deviations.

We start with a scenario that assumes constant parameter perturbations, where the task is to control the system to the rest position $x_{ref} = \big(0, 0, 0, 0\big)^T$. Subsequently, we will consider more complex settings in which the parameters change in time and the control law is continuously adapted. Using this example, we also reveal the difficulties that can be encountered in online parameter identification, e.g., concerning the information content of the data. We show how the additional computations of the Fisher information matrix in Algorithm 5.2 can help. We compare the results of our two algorithms for different scenarios.

5.4.1. Problem Description

For the classical example of a pendulum cart system, a rod with an attached mass m is mounted on a cart that moves along a longitudinal axis, see Figure 5.2. The massless rod is connected to the cart by a joint and can freely move in the joint. The state $y\colon [0, \infty) \to \mathbb{R}$ describes the position of the cart. The state $\theta\colon [0, \infty) \to \mathbb{R}$ is the angle of the rod's deflection with respect to the vertical axis. The control $u\colon [0, \infty) \to \mathbb{R}$ describes the force acting on the horizontal direction of motion of the cart. The nonlinear dynamical system is defined by

$$\begin{aligned} (M+m)\ddot{y} + ml\ddot{\theta}\cos(\theta) - ml\dot{\theta}^2\sin(\theta) &= u \\ ml^2\ddot{\theta} - mgl\sin(\theta) &= -ml\ddot{y}\cos(\theta). \end{aligned} \tag{5.13}$$

The values $m = 1\,\mathrm{kg}$ and M describe the mass of the round mass point and the mass of the trolley in kilogram. The value l is the length of the rod in meters, and $g = 9.8065\,\mathrm{m\,s^{-2}}$ is the gravitational acceleration. We define the state $x = x(t) := (y, \dot{y}, \theta, \dot{\theta})^T \in \mathbb{R}^4$. Then the nonlinear system equations (5.13) are given in state-space as

$$\dot{x} = f(x) = \begin{pmatrix} x_2 \\ \frac{1}{D}\left(u + mlx_4^2\sin(x_3) - mg\sin(x_3)\cos(x_3)\right) \\ x_4 \\ \frac{g}{l}\sin(x_3) - \frac{\cos(x_3)}{lD}\left(u + mlx_4^2\sin(x_3) - mg\sin(x_3)\cos(x_3)\right) \end{pmatrix}, \tag{5.14}$$

with $D = M + m\sin^2(x_3)$. Linearization of the system around the upright rest position $\hat{x} = \big(0,0,0,0\big)^T$, using the simplifications $\sin\theta \approx \theta$, $\cos\theta \approx 1$ and $\dot{\theta}^2 \approx 0$ for small angles θ, and assuming a non-vanishing pendulum length and mass $l \neq 0, M \neq 0$ the equations (5.13) can be written as

$$\begin{aligned}(M+m)\ddot{y} + ml\ddot{\theta} &= u\\ l\ddot{\theta} - g\theta &= -\ddot{y}.\end{aligned}$$

Then the linear system dynamics in state-space representation are obtained as

$$\dot{x} = \begin{pmatrix}\dot{x_1}\\ \dot{x_2}\\ \dot{x_3}\\ \dot{x_4}\end{pmatrix} = \begin{pmatrix}x_2\\ \frac{-mg}{M}x_3 + \frac{1}{M}u\\ x_4\\ \frac{(M+m)g}{Ml}x_3 + \frac{-1}{Ml}u\end{pmatrix} = \underbrace{\begin{pmatrix}0 & 1 & 0 & 0\\ 0 & 0 & \frac{-mg}{M} & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & \frac{(M+m)g}{Ml} & 0\end{pmatrix}}_{=:A(p)} x + \underbrace{\begin{pmatrix}0\\ \frac{1}{M}\\ 0\\ \frac{-1}{Ml}\end{pmatrix}}_{=:B(p)} u. \tag{5.15}$$

The simplified system dynamics are given by (5.15) in the well-known form

$$\dot{x}(t) = A(p)x(t) + B(p)u(t).$$

The states $x(t) \in \mathbb{R}^4$ are the position of the cart, the velocity of the cart, the angle of the pendulum, and its angular velocity. The parameter vector is given by $p := (M, l)^T \in \mathbb{R}^2$.

5.4.2. Simulation Environment and Settings

We consider the nonlinear system equations of the pendulum-cart system (5.14) for the system simulation with the nominal values $M = 10\,\text{kg}$ and $l = 1\,\text{m}$. We define the parameter vector as $p := (M, l)^T \in \mathbb{R}^2$ and thus it is $p_0 = (10, 1)^T$ the nominal parameter value. The simulation of the control process is implemented in MATLAB®. For a simulation horizon $T > T_s$, the pendulum cart system is simulated on an advancing time interval of length T_s by integrating the nonlinear system dynamics with the true parameter values using a Runge-Kutta scheme of fourth order.

The offline computation of the initial control law and the online parameter identification problems are implemented in C++ since they are solved with TransWORHP, which is based on C++. The solver settings and applied methods are described below the numerical tests in Subsections 5.4.4, 5.4.6 and 5.4.7. All computations in this chapter and the following are executed on a "Lenovo Thinkpad T460s" running "Ubuntu 16.04.7 LTS (64 bit)" as the Operating System with an "Intel(R) Core(TM) i5-6200U" CPU.

Sensor Noise

In practical applications, the sensors introduce some disturbances since their measurement accuracy is usually limited. To take this noise source into account in our simulations, we add white Gaussian noise $v(t) \sim \mathcal{N}(0, \Sigma)$ to the simulated states:

$$y(t) := x(t) + v(t). \tag{5.16}$$

For the normal distribution the covariance matrix is diagonal: $\Sigma = \text{diag}(\sigma_1^2, ..., \sigma_4^2)$, where $\sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4) \in \mathbb{R}^4$ is the vector of standard deviations in each state. A pseudo-random number generator generates the noise $v(t) \in \mathbb{R}^4$ as normally distributed noise with mean $\mu = 0$ and standard deviations σ_i. This sensor noise also disturbs the measured feedback that the controller directly uses to determine the controls $u(t) = -K(x(t) + v(t))$. To address this circumstance, the measurement noise is used within the integration method in the simulation. In real-world applications, measurement noise in feedback is often reduced with filters. Low-pass filters, such as moving average filters, are commonly used. However, they can blur signal sharpness as demonstrated by de Cheveigné and Nelken [CN19]. We do not utilize these filters in the presented simulation to obtain unbiased results. Consequently, the control appears slightly noisy since the feedback signal directly influences it.

In the following numerical experiments, a measurement noise $v(t)$ is included with the standard deviation $\sigma = (0.4, 0.09, 0.004, 0.004)$ corresponding to (5.16).

Simulation Scenarios

The main goal is to stabilize the upright pendulum position while controlling the system (5.15) from an initial state x_0 to another rest position x_{ref}, e.g., the origin $x_{ref} = \big(0, 0, 0, 0\big)^T$. In addition, the control effort is to be minimized. This is done by minimizing the objective function

$$F(x, u, p) = \int_0^\infty (x(t) - x_{ref}(t))^T Q (x(t) - x_{ref}(t)) + u(t)^T R u(t) \text{ d}t. \tag{5.17}$$

The weights are set to the following values:

$$R = 0.5 \text{ and } Q = \begin{pmatrix} 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

This implies that the controller wants to reach the position of the cart to a reference value with a minimum control effort at the same time. To test the performance of our control algorithms with online parameter identification on the inverted pendulum-cart system, we consider different parametric perturbations $p = p_0 + \Delta p$ with $p_0 = (10, 1)^T$.

The system is generally considered on a time horizon of $T = 120\,\mathrm{s}$ except for scenario 0 where we use $T = 40\,\mathrm{s}$.

$$
\begin{aligned}
&\text{Scenario A} \quad \Delta p = (2.0, 0.5)^T,\\
&\text{Scenario B} \quad \Delta p = (6.0, 1.5)^T,\\
&\text{Scenario C} \quad \Delta p = (-1.0, -0.5)^T,\\
&\text{Scenario D} \quad \Delta p = (2.0, -0.5)^T,\\
&\text{Scenario E} \quad \Delta p = (-2.0, 1.0)^T,\\
&\text{Scenario F} \quad \Delta p = (5.0 - 0.1t, 0.5 + 0.01t)^T, \text{ for } t \in [0, 120],\\
&\text{Scenario G} \quad \Delta p = \begin{cases} (5.0 - 0.1t, 1.8 - 0.0004(t-50)^2)^T, \text{ for } t \in [0, 50), \\ (0.14(t-50), 1.8 - 0.0004(t-50)^2)^T, \ t \in [50, 120], \end{cases}\\
&\text{Scenario H} \quad \Delta p = (1.0 + 2.0(\log 10(t+1)), 0.5 + 0.01t)^T, \text{ for } t \in [0, 120],\\
&\text{Scenario I} \quad \Delta p = \begin{cases} (4.0, -0.5)^T, \text{ for } t \in [0, 40] \text{ or } t \in (80, 120], \\ (-3.0, -0.5)^T, \text{ for } t \in (40, 80]. \end{cases}
\end{aligned}
\tag{5.18}
$$

Scenario A assumes a constant positive small perturbation of the two parameters. A large positive constant parameter deviation is considered in scenario B. Scenarios C, D, and E also consider negative perturbations and various combinations of them. Case F provides a linearly decreasing mass perturbation from $\Delta M = 5$ to $\Delta M = -5$ and a linearly increasing perturbation in the pendulum length. In scenario G, the perturbation in the mass is first linearly decreased and then strongly increased, and the pendulum length also changes at a moderate rate over the entire period. In the case of H, both parameters increase with time. Scenario I deals with the case where the mass suddenly changes while the pendulum length is constantly perturbed with $\Delta l = -0.5$.

Furthermore, the behavior of the algorithm is tested and analyzed in different scenarios for the reference state. For this purpose, we consider the following different trajectories for tracking:

$$
\begin{aligned}
&\text{Sc. 0} \quad x_{ref}(t) = \begin{cases} (0,0,0,0)^T, \quad \text{for } t \in [0, 40]. \end{cases}\\
&\text{Sc. 1} \quad x_{ref}(t) = \begin{cases} (10,0,0,0)^T, & \text{for } t \in [0, 20], (68, 88], \\ (-10,0,0,0)^T, & \text{for } t \in (20, 40], (88, 108], \\ (0,0,0,0)^T, & \text{otherwise.} \end{cases}\\
&\text{Sc. 2} \quad x_{ref}(t) = \begin{cases} (-20,0,0,0)^T, & \text{for } t \in [0, 20], (40, 60], (80, 100], \\ (0,0,0,0)^T, & \text{otherwise.} \end{cases}\\
&\text{Sc. 3} \quad x_{ref}(t) = \begin{cases} (20,0,0,0)^T, & \text{for } t \in [0, 50], \\ (0,0,0,0)^T, & \text{otherwise.} \end{cases}
\end{aligned}
\tag{5.19}
$$

$$\text{Sc. 4} \qquad x_{ref}(t) \quad = \begin{cases} (20+2\sin(t),0,0,0)^T, & \text{for } t \in [0,20],(80,100], \\ (-20+2\sin(t),0,0,0)^T, & \text{for } t \in (40,60], \\ (0,0,0,0), & \text{otherwise.} \end{cases}$$

$$\text{Sc. 5} \qquad x_{ref}(t) \quad = \begin{cases} (40+3\sin(2t),0,0,0)^T, & \text{for } t \in [0,12],(56,72], \\ (-40+3\sin(2t),0,0,0)^T, & \text{for } t \in (12,32],(100,120], \\ (0+3\sin(2t),0,0,0)^T, & \text{otherwise.} \end{cases}$$

Scenario 0 aims to stabilize the origin $x_{ref} = (0,0,0,0)^T$. Further, scenarios 1, 2, and 3 show the cart system moving back and forth with different speeds between two positions while the pendulum is stabilized in the upright position. Tasks 4 and 6 represent more dynamic reference states that require constant intervention by the controller. This results in a permanent excitation of the system, which, as we will see in Subsection 5.4.8, positively affects the identifiability of the model parameters.

The simulation horizon for the system is fixed to $T = 40\,\mathrm{s}$ for scenario 0 and $T = 120\,\mathrm{s}$ for all other scenarios. In scenario 0, the origin is the reference value and is usually reached quickly, so we have shortened the simulation horizon for visualization reasons. The sampling time is set in all cases to $\Delta t = 0.05\,\mathrm{s}$ and the sampling horizon to $T_s = 4\,\mathrm{s}$. This is the time length of a data set used for one parameter identification in each case. The choice of the data sampling horizon has to be made by the user and is often based on experience. It is necessary to find a balance between a sufficient horizon length and, thus, a larger data set on the one hand and the increasing computation time for large data sets on the other hand. The horizon must be long enough so that a sufficiently large amount of data is available for successful parameter identification. However, the computation time should also not become too long for online computation due to a large data set. Another aspect that needs to be considered is the rate of change of the parameters. The Fisher information matrix (5.10) can also be helpful here to evaluate whether a time interval contains enough information for a successful parameter identification or should be enlarged. We consider different scenarios to demonstrate the system's different possibilities in terms of dynamics and parameter changes.

5.4.3. Verification of the Assumptions for `adLQR`

Next, we check whether all necessary assumptions of the approximation theorem for parametric LQR problems in Theorem 3.3.2 are fulfilled and, thus, whether we can apply the presented algorithms that are based on `adLQR`. The assumptions are that

1. R is positive definite,
2. Q is positive semidefinite,
3. $(A(p_0), B(p_0))$ is stabilizable,
4. $(A(p_0), \sqrt{Q(p_0)})$ is observable,
5. $A(p)$, $B(p)$, $Q(p)$, $R(p)$, and $x_0(p)$ are three times continuously differentiable with respect to p.

It applies that $R = 0.5 > 0$ and therefore R is positive definite. This assumption (1.) is fulfilled. The eigenvalues of Q are $\lambda_1 = 0.5$ and $\lambda_{2,3,4} = 0$, so that it is $\lambda_i \geq 0$ for all $i \in \{1, 2, 3, 4\}$ and thus Q is positive semidefinite and (2.) is true. For assumption (3.) we compute the controllability matrix with $p_0 = (10, 1)^T$:

$$\begin{aligned}\mathcal{C} &:= \begin{pmatrix} B(p_0) & A(p_0)B(p_0) & A(p_0)^2B(p_0) & A(p_0)^3B(p_0) \end{pmatrix} \\ &= \begin{pmatrix} 0 & 0.1 & 0 & 0.0981 \\ 0.1 & 0 & 0.0981 & 0 \\ 0 & -0.1 & 0 & -1.0787 \\ -0.1 & 0 & -1.0787 & 0 \end{pmatrix}.\end{aligned}$$

According to the Kalman criterion in Theorem 3.1.6 it holds that $rank(\mathcal{C}) = 4$ and the system $(A(p_0), B(p_0))$ is controllable. The system is therefore also stabilizable since stabilizability is the weak version of controllability. The observability in (4.) is shown with the matrix of controllability of the dual system $(A(p_0)^T, \sqrt{Q(p_0)}^T)$. For this we compute

$$\begin{aligned}\mathcal{C} &:= \Big(\sqrt{Q(p_0)}^T \quad A(p_0)^T\sqrt{Q(p_0)}^T \quad A(p_0)^{T^2}\sqrt{Q(p_0)}^T) \quad A(p_0)^{T^3}\sqrt{Q(p_0)}^T\Big) \\ &= \begin{pmatrix} 0.7071 & \vdots & 0 & \vdots & 0 & \vdots & 0 & \vdots \\ 0 & \mathbb{O}_{4\times 3} & 0.7071 & \mathbb{O}_{4\times 3} & 0 & \mathbb{O}_{4\times 3} & 0 & \mathbb{O}_{4\times 3} \\ 0 & \vdots & 0 & \vdots & -0.6934 & \vdots & 0 & \vdots \\ 0 & \vdots & 0 & \vdots & 0 & \vdots & -0.6934 & \vdots \end{pmatrix},\end{aligned}$$

with $\mathbb{O}_{4\times 3} \in \mathbb{R}^{4\times 3}$ is the 4×3 zero matrix. Since it is $\text{rank}(\mathcal{C}) = 4$, the condition (4.) is fulfilled. It follows that all assumptions for an optimal solution of the LQR problem are fulfilled, cf. Theorem 3.2.5, and there exists the uniquely defined solution $S^* = S^*(p_0)$ of the algebraic Riccati equation:

$$SA(p_0) + A(p_0)^T S - SB(p_0)R^{-1}B(p_0)^T S + Q = 0.$$

Since the matrices

$$A(p) = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{-mg}{M} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{(M+m)g}{Ml} & 0 \end{pmatrix} \quad \text{and} \quad B(p) = \begin{pmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{-1}{Ml} \end{pmatrix}$$

are three times continuously differentiable with respect to p, also the last assumption (5.) of Theorem 3.3.2 is fulfilled and it exists a neighborhood $U(p_0) \subset \mathbb{R}^4$ of p_0 such that the optimal feedback matrix $K(p) \colon U(p_0) \to \mathbb{R}^{1\times 4}$ is a uniquely continuously differentiable function for all $p \in U(p_0)$.

5.4.4. Offline Computations

To illustrate the procedure, we compute the nominal feedback gain $K(p_0)$ for the first tracking scenario with $x_{ref} = (0, 0, 0, 0)^T$. This is accomplished by solving an optimization problem of the type 3.12. We compute the nominal feedback gain $K(p_0)$ once as an example for the first tracking scenario with $x_{ref} = (0, 0, 0, 0)^T$. This is done by solving an optimization problem of the type 3.12. Accordingly, we formulate the infinite optimization problem:

$$\begin{aligned} \min_{K(p_0),x} \quad & \int_0^\infty x(t)^T \left(Q + K(p_0)^T R K(p_0)\right) x(t) \,\mathrm{d}t \\ \text{w.r.t.} \quad & \dot{x}(t) = \left(A(p_0) - B(p_0)K(p_0)\right) x(t), \\ & x(0) = x_0. \end{aligned}$$

We implement this problem using the software library TransWorhp, described in Subsection 4.2.3 and use the WorhpZen module to obtain the parametric sensitivities $\frac{dK}{dp}(p_0)$. The optimization variables are the states $x(t)$ and the elements of the searched feedback gain $K(p_0) = (K(p_0)_1, \dots, K(p_0)_4) \in \mathbb{R}^4$. Since the repeated numerical evaluation of the infinite integral would become too costly, we restrict ourselves to a sufficiently large time horizon. This is possible because if the integral exists, the remainder that we cut off from the integral becomes from the integral becomes negligibly small. Experiments have shown that $T = 50\,\mathrm{s}$ is sufficient for a time discretization with 501 discretization points.

We use a full discretization approach with Hermite-Simpson discretization and obtain the solution,

$$K(p_0) = (-0.998781, -5.29252, -245.52, -75.1487).$$

The Hessian matrix is computed by Worhp using finite differences. The computation of the sensitivity derivatives yields,

$$\frac{dK}{dp}(p_0) = \begin{pmatrix} 0.00228937 & 0.00529661 \\ -0.203255 & -0.282624 \\ -20.9186 & -15.9428 \\ -6.67873 & -42.762 \end{pmatrix}.$$

Looking at the feedback gain $K(p_0)$, we can observe that the control u reacts much more sensitively to deviations from the reference state $x_{ref} = (0, 0, 0, 0)^T$ in the angle x_3 or the angular velocity x_4 than to deviations in the position x_1 or the velocity x_2. Further, we can see from the sensitivities that the gain is very sensitive to disturbances of the angle and the angular velocity. Thus, we conclude that in the presence of perturbations in the mass of the cart and the length of the pendulum, the gain will be modified, in particular in the entries multiplied by the deviations in the angles and angular velocities by the real-time approximation ($K(p) \approx K(p_0) + \frac{dK}{dp}(p_0)\Delta p$).

5.4.5. The NDPIP for the Inverted Pendulum on a Cart

We formulate the nonlinear parameter identification problem with the covariance matrix $\Sigma = \operatorname{diag}(0.16, 0.81 \times 10^{-2}, 1.6 \times 10^{-5}, 1.6 \times 10^{-5})$ as weighting matrix.

$$
\begin{aligned}
\min_{x_0,p} \quad & F(x_0,p) := \frac{1}{2M}\sum_{i=0}^{M}(x(t_i;p,x_0)-y_i)^T\Sigma^{-1}(x(t_i;p,x_0)-y_i),\\
\text{w.r.t.} \quad & \dot{x}(t) = \begin{pmatrix} x_2 \\ \frac{1}{M+\sin^2(x_3)}\left(u + l x_4^2\sin(x_3) - g\sin(x_3)\cos(x_3)\right) \\ x_4 \\ \frac{g}{l}\sin(x_3) - \frac{\cos(x_3)}{l(M+\sin^2(x_3))}\left(u + l x_4^2\sin(x_3) - g\sin(x_3)\cos(x_3)\right)\end{pmatrix},\\
& x(t_0) = x_0,\\
& \begin{pmatrix}-8.0\\-0.5\end{pmatrix} \le \Delta p \le \begin{pmatrix}10.0\\2.0\end{pmatrix}.
\end{aligned}
\tag{5.20}
$$

Further, it is $p = p_0 + \Delta p$ with $p_0 = (10,1)^T$. Here, lower and upper bounds are specified for the parametric perturbations $\Delta p = (\Delta M, \Delta l)^T$. These include physical limits. For example, the pendulum length and the mass cannot become zero or negative. Furthermore, the bound on the parameter space can prevent convergence to an undesirable local minimum. The states of the system are in different ranges of magnitude. Therefore, the states are adjusted with a scaling factor $c = (\frac{1}{20}, \frac{1}{10}, 5, 5)^T$ as described in Subsection 2.4.4 to increase the chances for convergence of the optimization.

The problem (5.20) is implemented in TransWORHP. As an initial guess for the parameters in the first identification problem, we use $p^{[0]} = p_0 = (10,1)^T$ and then use the solution of the previous problem in each successive problem. The superscript means the first iteration step in the NLP solution method. The states x_i are initialized at all discretization points by the respective measurements, thus $x^{[0]}(t_i) = y_i$ for all $i = 0, \dots, M-1$. By providing those initial guesses, the speed of convergence can be improved. We apply the full discretization approach with a trapezoidal scheme to transform the problems. The number of discretization points for the NDPIP is set to $N = \frac{T_s}{\Delta t} + 1 = 41$ and thus corresponds to the number of measurements on the considered interval for simplicity. The resulting NLPs are then solved again with WORHP. The Hessian matrix is approximated by a BFGS method.

The thresholds for the optimality and complementarity condition of the solution are changed from the default value 10^{-6} to 10^{-5} to speed up the computation times. The feasibility threshold is kept to maximize compliance with the dynamical system.

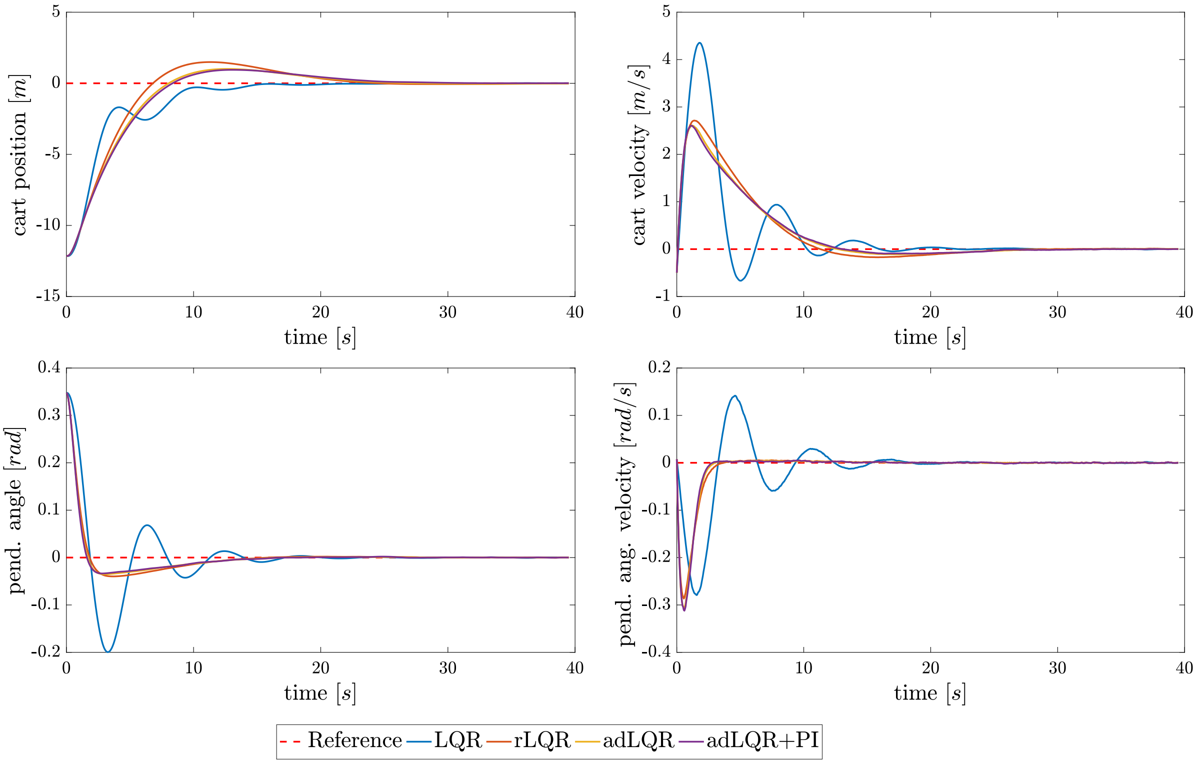

Figure 5.3.: *States of the inverted pendulum on a cart for scenario 0-B controlled with different LQR-based control laws; cart position, cart velocity, pendulum deflection angle, and pendulum angular velocity for a simulation horizon of $T = 40\,\mathrm{s}$.*

5.4.6. Numerical Results for Scenario 0-B: Constant Medium-Sized Parameter Perturbation

In the following, we examine the performance of the presented algorithm `adLQR+PI` and compare it to `adLQR` with the true parameters and the optimal LQR controller `rLQR`. For this, we assume that all control algorithms are in operation, and we choose a random time during operation and see how the controllers react to parametric perturbations.

First, we consider scenario 0-B, cf. (5.18) and (5.19), corresponding to a medium constant parameter perturbation and the stabilization of the origin. We study the system on a simulation horizon of $T = 40\,\mathrm{s}$. Figure 5.3 shows the states of the simulated inverted pendulum, i.e., the solution of the nonlinear system with the true parameters, controlled with different feedback laws. The initial state at time $t = 0$ is $x_0 = \begin{pmatrix} -12.127 & -0.497 & 0.348 & 0.008 \end{pmatrix}^T$. The red dashed line shows the reference value. The blue line represents the classical `LQR` with feedback gain $K(p_0)$ computed for the nominal parameters. In red is shown the best possible feedback law `rLQR` with feedback gain $K(p^*)$. In this case, the feedback matrix is optimized again after each simulation interval under the knowledge of the true parameters. The yellow curve shows the behavior using the adaptive feedback law `adLQR` with updated feedback gain $K(p_0) + \frac{dK}{dp}(p^*)$

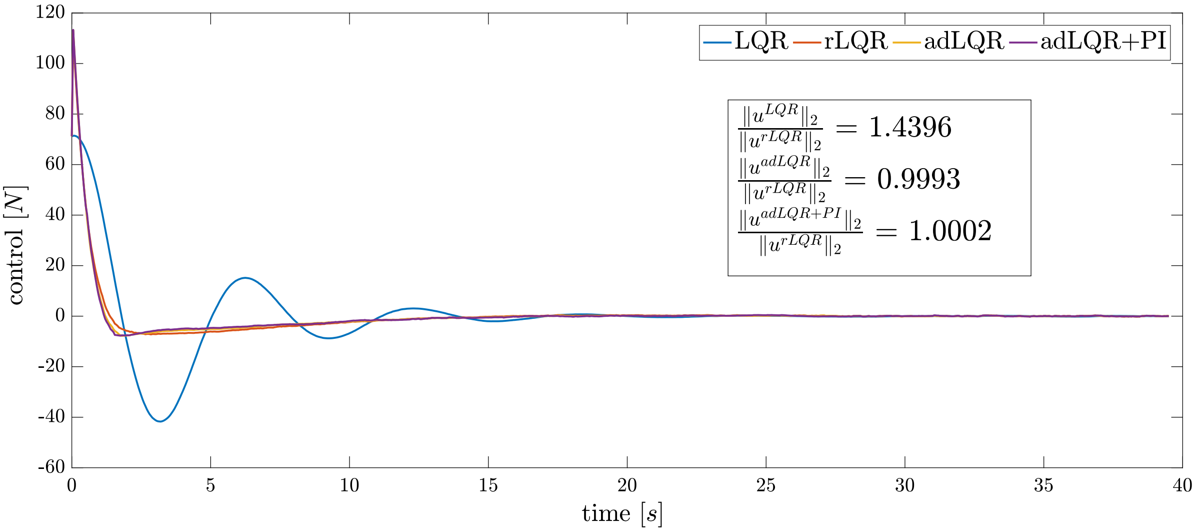

Figure 5.4.: *Control of the inverted pendulum on a cart for scenario 0-B controlled with different LQR-based control laws for a simulation horizon of* $T = 40\,\mathrm{s}$.

if the true parameter values would be available to the controller. The performance of this controller is considered as the benchmark for the proposed algorithm `adLQR+PI` with $K(p_0) + \frac{dK}{dp}(p)$. The states for this combined controller are shown in purple. All controllers are able to stabilize the system. It can be observed that nominal `LQR` develops significantly more oscillation than all other controllers. The proposed algorithm `adLQR+PI` performs similarly well to the perturbation-optimal controllers `rLQR` and `adLQR`, for which we assumed that the true perturbations are known.

This characteristic can also be observed in the control effort. By looking at the control curves in Figure 5.4, we can see that the control behavior of `adLQR+PI` almost exactly matches that of the optimal `adLQR`. Quantitatively, this can be shown by comparing the total control effort over the time under consideration. We present the norms of the different controls relative to the norm of the perturbation-optimal controller `rLQR` in Figure 5.4. It can be observed here that `LQR` has about 43.96% more control overhead compared to the optimal `rLQR`. The `adLQR+PI` approach requires only about 0.02% more control effort than the optimal `rLQR`. In this setup, the perturbation-adapted controller `adLQR` with known perturbation requires about 0.07% less control effort than `rLQR`. It is noted that the slight noise seen in the control is due to the unfiltered state feedback in the control law, compare Subsection 5.4.2. We use the objective function (5.17) of the LQR problem as an additional quality measure for each control algorithm. For this purpose, we evaluate it a posteriori for the different approaches. The results are shown in Figure 5.5 and the respective objective function values in Table B.1 in Appendix B. It is easy to see that the objective function value of classical `LQR` is more than two times larger than that of any other controller. The best result can be achieved with `rLQR` by repeatedly recalculating the feedback gain online with known perturbations. The algorithm `adLQR` has an objective functional of just about 0.05% larger than that of

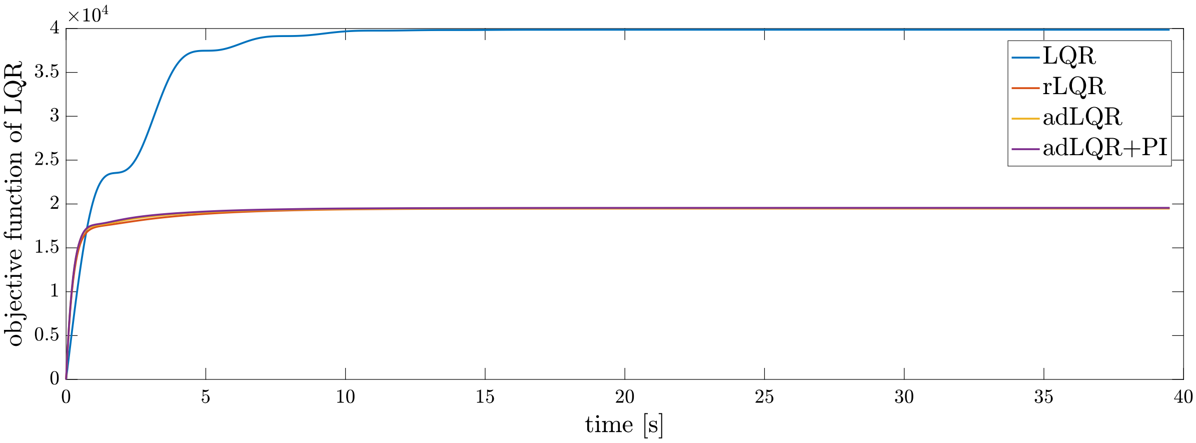

Figure 5.5.: *Objective function of LQR problem for the inverted pendulum on a cart for scenario 0-B controlled with different LQR-based control laws for a simulation horizon of* $T = 40\,\mathrm{s}$.

`rLQR`. The algorithm `adLQR+PI` achieves a very good result since the respective objective function value is not more than half a percent larger than that of `rLQR`.

In Figure 5.6, the estimates of the parameter perturbations Δp are shown as blue dots for each identification interval. In addition, the true value is depicted in red, which is constant in this scenario. It can be observed that the first parameter identifications determine very good estimates for the perturbations up to about $t = 6\,\mathrm{s}$. However, in some cases, the identification results deviate significantly from the true perturbation value, especially for the second parameter, the pendulum length. This is because the data in the later intervals does not contain enough information. Then, in the end, the box constraints become often active. If we look again at Figure 5.3, we can see that from $t = 15\,\mathrm{s}$ for the `adLQR+PI` there is hardly any dynamic behavior left in the system, compare the purple line. In particular, in the bottom two plots, there is already from $t = 6s$ almost no motion in the pendulum, making the identification of the pendulum length impossible. In such a case, we speak of *practical non-identifiability* of the parameters, see Section 2.4.4.

For the presented online parameter identification algorithm `adLQR+PI`, it is desirable in such a case that no parameter identification is performed, and instead, the parameters identified in the last iteration are retained. As soon as there is more dynamic in the measurements again, the identification of parameters is continued. It would, of course, be possible to make additional use of data further back in time to enable parameter identification. However, we are interested in performing a temporary valid identification for data that is as recent as possible since we generally assume that the parameters change over time. In this introductory demonstrative scenario, the parameters are time-invariant, but most of the other scenarios deal precisely with the case of dynamic parameters.

For this purpose, we develop the extension with the computation of the Fisher informa-

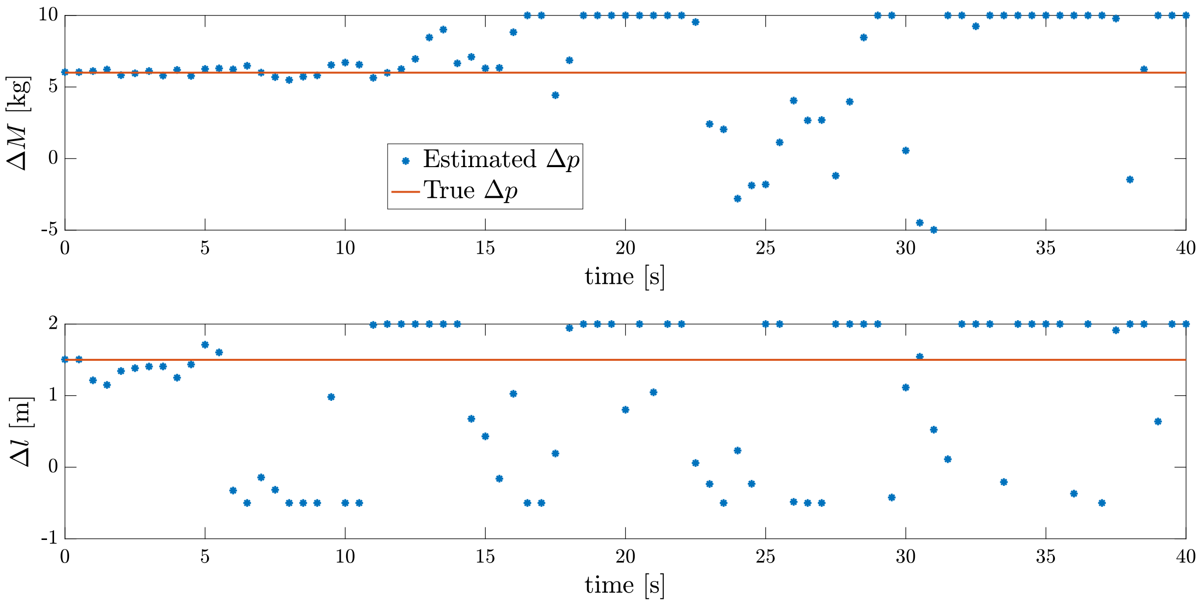

Figure 5.6.: *Parameter deviation of the inverted pendulum on a cart for scenario 0-B controlled with different LQR-based control laws for a simulation horizon of* $T = 40\,\mathrm{s}$.

tion matrix `adLQR+PI+FIM`, see Algorithm 5.2. There, we use the Fisher information matrix $\mathcal{I}_p$ to estimate the information content of measurements for a possible parameter estimation. In this example, we approximate the Fisher information matrix with forward sensitivities, see Appendix A.1. Since the diagonal of the inverse of the FIM $\mathcal{I}_p^{-1}$ contains the covariances of the maximum likelihood estimator, the trace of the inverse is used to indicate the mean accuracy of the parameter estimation. Figure 5.7 shows an approximation of the trace of the inverse before each PI given the data of the last sampling interval. If this value is larger than the fixed threshold $\delta_{FIM} = 0.2$, no parameter identification is performed, and the previous value is kept: $p^{[k+1]} = p^{[k]}$. The threshold is shown as an orange line in Figure 5.7.

By using Algorithm 5.2 with the additional FIM-check, the solution trajectories of the states and controls in scenario 0 remain qualitatively the same as the results of Algorithm 5.1, compare the Figures B.1 and B.2 in Appendix B. However, a significant improvement can be observed in the parameter estimates in Figure 5.8. The parameter estimates without FIM-check are shown in blue, and those with the additional FIM consultation in green. The FIM checks avoid inaccurate parameter estimates from time $t = 6\,\mathrm{s}$, and the true parameters over time are thus better approximated. We evaluate the *root mean square error (RMSE)* in the parametric perturbations as

$$\mathrm{RMSE}_{\Delta p} := \sqrt{\frac{\sum_{i=1}^{N_{PI}} (\Delta p_i - \Delta p_i^{ref})^2}{N_{PI}}}.$$

It is computed on the considered simulation interval for the total number of parameter

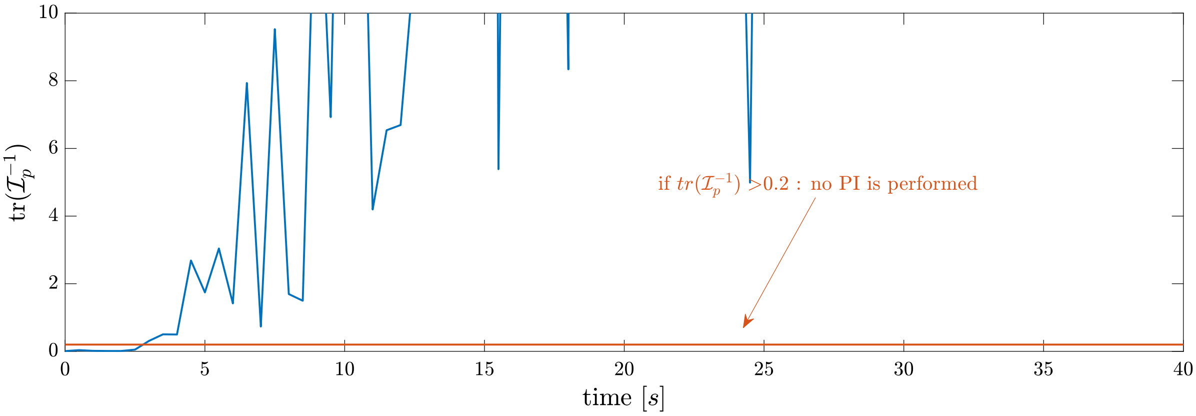

Figure 5.7.: *Trace of the inverse of the approximated Fisher information matrix* $\mathcal{I}_p$ *of the inverted pendulum on a cart for scenario 0-B for a simulation horizon of* $T = 40\,\mathrm{s}$.

identifications denoted with N_{PI}. For Algorithm 5.1 it is

$$\mathrm{RMSE}_{\Delta p} = \begin{pmatrix} 3.7538 \\ 1.1156 \end{pmatrix}$$

and for Algorithm 5.2 with additional FIM checks it results in

$$\mathrm{RMSE}_{\Delta p} = \begin{pmatrix} 0.1155 \\ 0.1052 \end{pmatrix}.$$

So, in this respect, a significant improvement can be produced, which also leads to a more stable control result, as we will see in this work. Since the solution trajectories are similar, one may argue that the parameter mismatches may not matter. In a second scenario, however, we will see that it is important not to make arbitrary identifications based on insufficient data. There, we can observe that this is beneficial, for example, to better stabilize the system behavior in case of a renewed sudden movement. The computation times per parameter identification are, on average, 50 milliseconds for this scenario. Since we provide a period of 0.5 seconds for identification, these problems can be reliably solved online and are real-time capable. Also, the additional 20-30 milliseconds for the computation of the FIM can be comfortably handled in this time frame. At this point, it should be noted that the individual computation times, particularly that of the `rLQR`, are neglected for this analysis.

5.4.7. Numerical Results for Scenario 2-G: Large and Time-Varying Parameter Perturbation

Now, we consider a more complex scenario where the different controllers track the reference state according to scenario 2 in (5.19). Here, the cart moves continuously back and forth between the positions $x_{1,ref} = 0$ and $x_{1,ref} = -20$. These reference values are

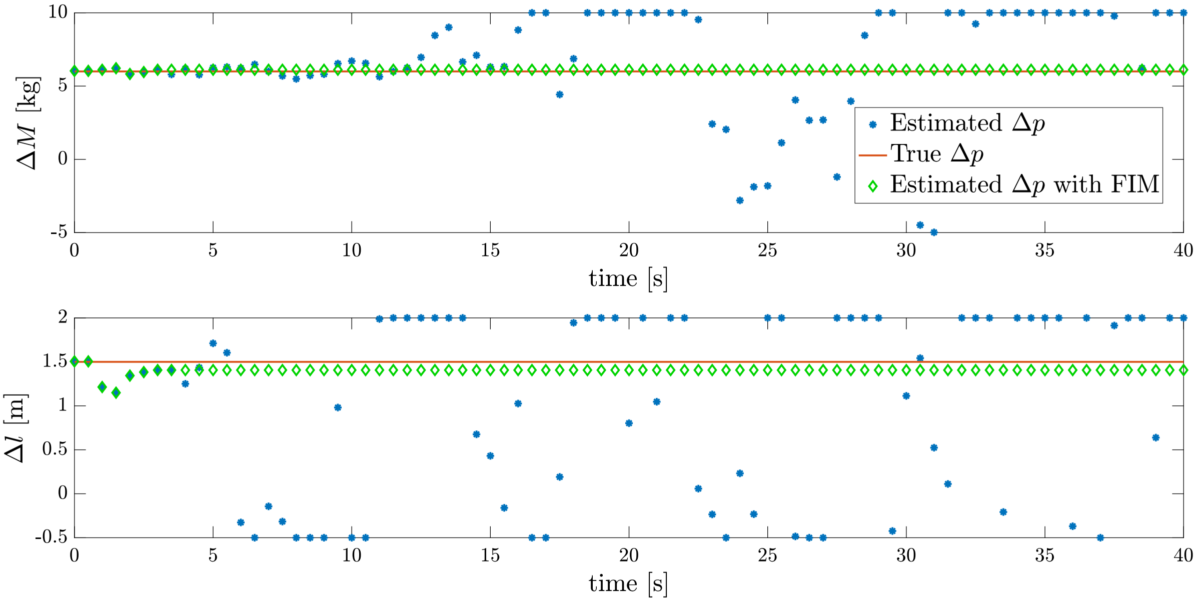

Figure 5.8.: *Comparison of parameter Δp of the inverted pendulum on a cart for Scenario 0-B with and without additional FIM-check for a simulation horizon of $T = 40\,\mathrm{s}$.*

again shown as a dashed red line in Figure 5.9. Moreover, we assume that the model parameters we are interested in change over time, as indicated in parameter scenario G in (5.18). In Figure 5.9, it can be seen that all control approaches reach the target positions, although the classical `LQR` has difficulties stabilizing the system when the perturbations of the parameters increase. Furthermore, it can be observed that `LQR` has a significantly more dynamic response in the deflection of the pendulum and the velocity of the cart. This is also reflected in the control effort. It can be found in Figure 5.10 that `LQR` requires almost 90% more control effort than `rLQR`, while the control approach `adLQR+PI` requires only about 1.4% more energy than the optimal `rLQR`. Regarding the control effort, `adLQR` is even better than `rLQR` by about one and a half percent.

In Figure 5.11, the trace of the inverse of the Fisher information matrix $\mathrm{tr}(\mathcal{I}_p^{-1})$ for the considered scenario is shown. This is computed for an interval of length $4s$. It can be observed that, as expected, the value for $\mathrm{tr}(\mathcal{I}_p^{-1})$ becomes large in the regions with little excitation in the system. This means that the variance of the parameter estimates would be large, and thus, no reliable parameter identification can be performed with this data. Therefore, the PI is not initialized, and the current parameter estimate is preserved. The estimated parameters can be found as green diamonds in Figure 5.12. Here, for comparison purposes, the parameter estimates gained with Algorithm 5.1 without the FIM-check are also shown in blue. It can be seen that without the preliminary FIM-check, arbitrary identification results would be produced for some PIs. The pendulum length is especially sensitive in this way. The mean error between the true and the estimated parameters is for Algorithm 5.1

$$\bar{\varepsilon}_p = 1.6056$$

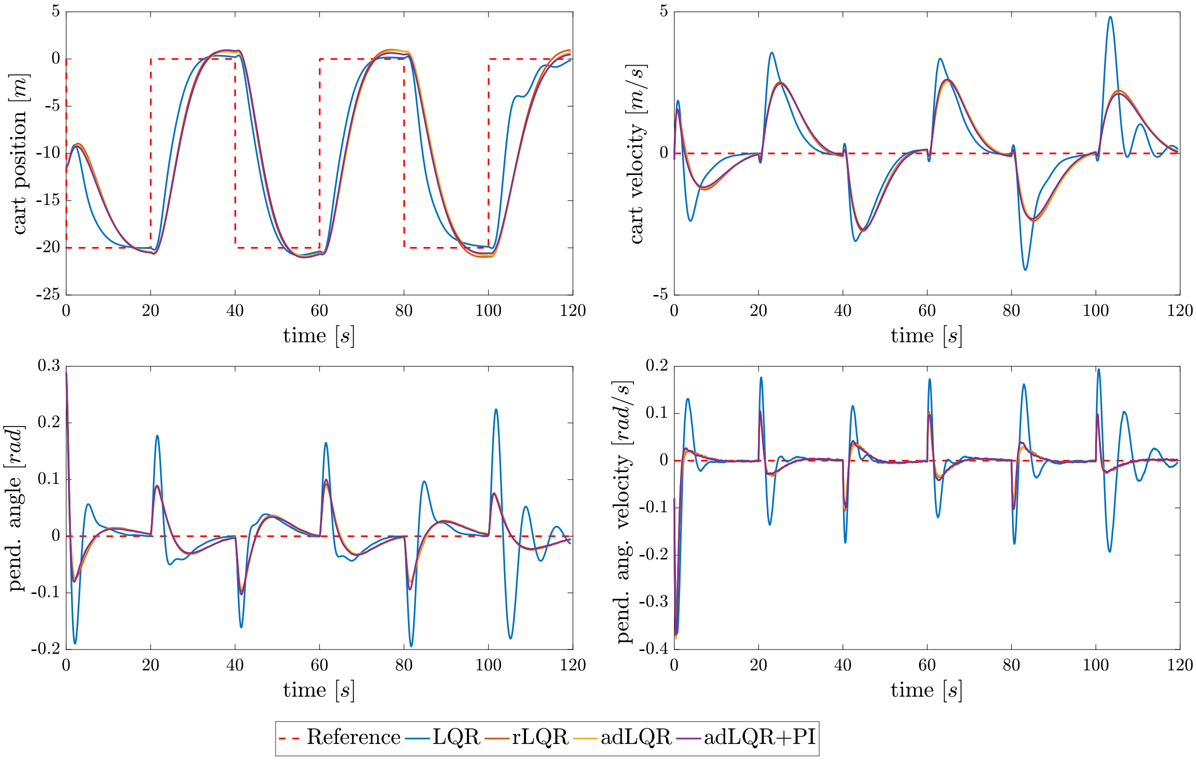

Figure 5.9.: *States of the inverted pendulum on a cart for scenario 2-G with additional FIM-check for a simulation horizon of* $T = 120\,\mathrm{s}$.

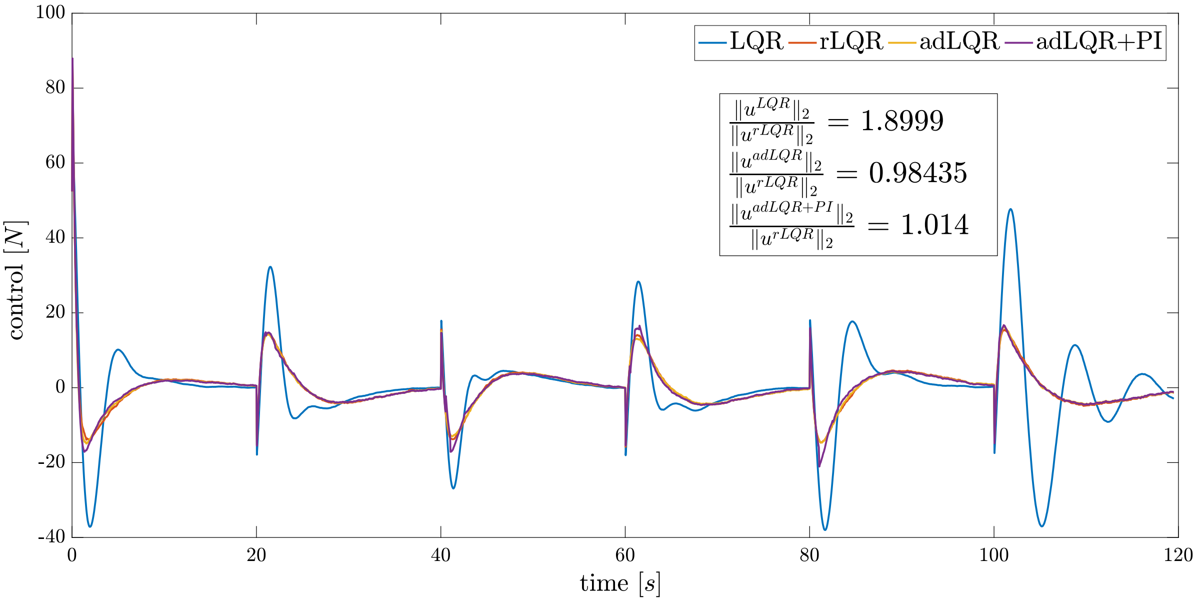

Figure 5.10.: *Control of the inverted pendulum on a cart for scenario 2-G with additional FIM-check for a simulation horizon of* $T = 120\,\mathrm{s}$.

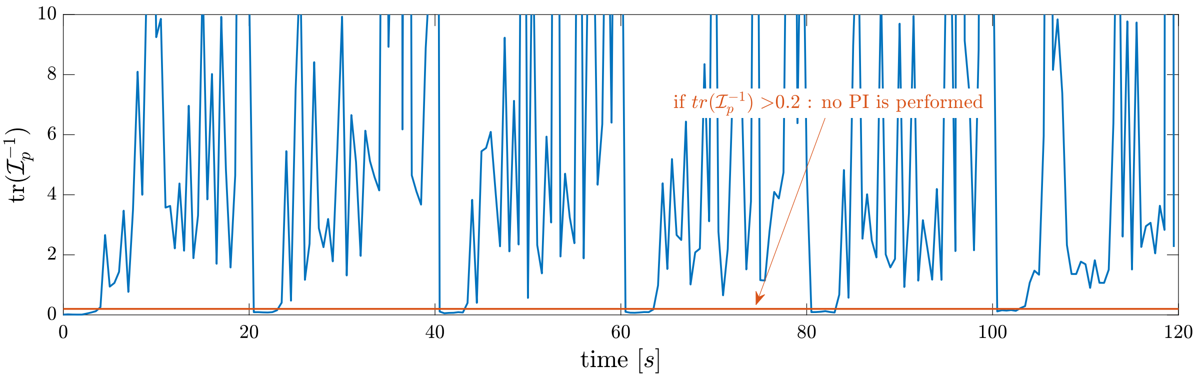

Figure 5.11.: *Trace of the inverse of the approximated Fisher information matrix $\mathcal{I}_p$ for scenario 2-G.*

and for Algorithm 5.2 it is

$$\bar{\varepsilon}_p = 1.1025.$$

This is an improvement of over 30%. At this point, it is emphasized again that the FIM can only be approximated and, therefore, does not guarantee a good identification result. In the same manner, it may occur that despite a large value for $\mathrm{tr}(\mathcal{I}_p^{-1})$, a good PI result could be obtained but is discarded. This is also due, as mentioned in Subsection 5.3.1, to various sources of error in the computation of the FIM. These include, for example, introducing an integration error in the numerical computation or evaluating the FIM in the previously estimated parameters instead of the optimal ones.

Next, the objective function of the LQR problem will be considered again to compare the different controllers. Thus, the objective function of the LQR problem is evaluated on the respective solution trajectory. Figure 5.13 shows the objective function F_{LQR} for scenario 2-G. It can be observed that the perturbation-adapted controllers `rLQR`, `adLQR`, `adLQR+PI` and `adLQR+PI+FIM` perform much better than the classical `LQR`, whose objective value is about 58% higher than the others. In particular, our proposed algorithms perform as well as `rLQR` and `adLQR`, where the parametric perturbation is taken as known. The specific values can be found in Appendix B in Table B.1. The results for all combinations of tracking and parametric perturbation scenarios for the different controllers are presented there.

5.4.8. Numerical Results for Scenario 4-F and 5-F: Increasing the Identifiability

Next, we will analyze two more problem scenarios that show how much effect the problem formulation has on the parameter identifiability. We consider the state reference scenarios 4 and 5 from (5.19). Further, we consider the model parameters from scenario F in (5.18). The desired tracking trajectory assumes that the cart moves back and forth between the positions $x_{1,ref} = 20$, $x_{1,ref} = 0$, and $x_{1,ref} = -20$. A special consideration

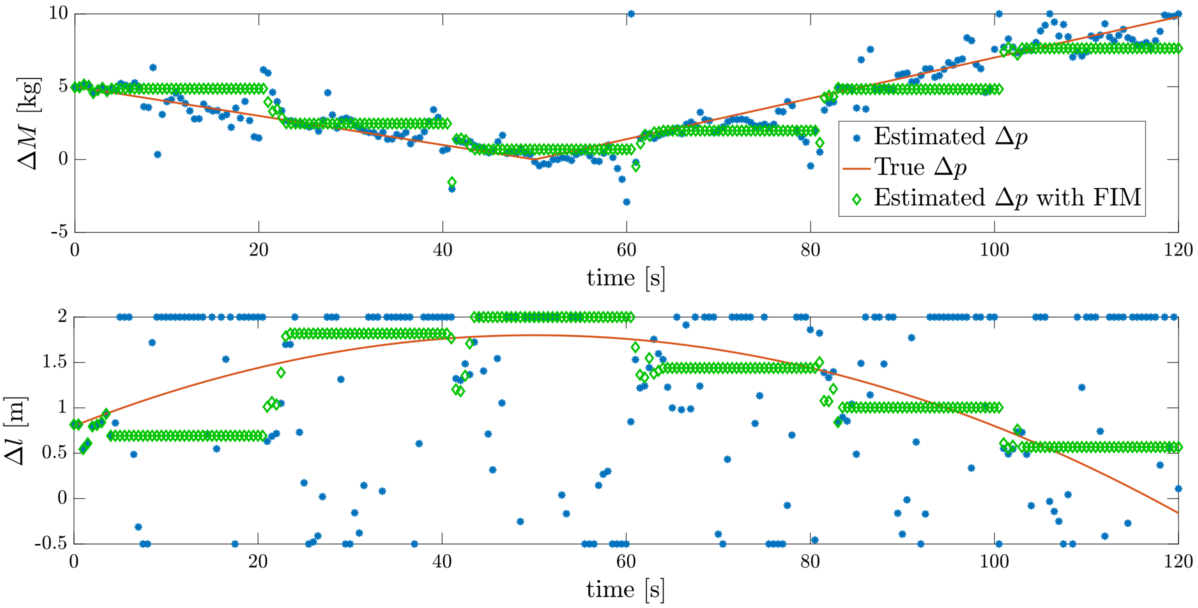

Figure 5.12.: *Comparison of parameter deviation of the inverted pendulum on a cart for scenario 2-G with and without additional FIM-check.*

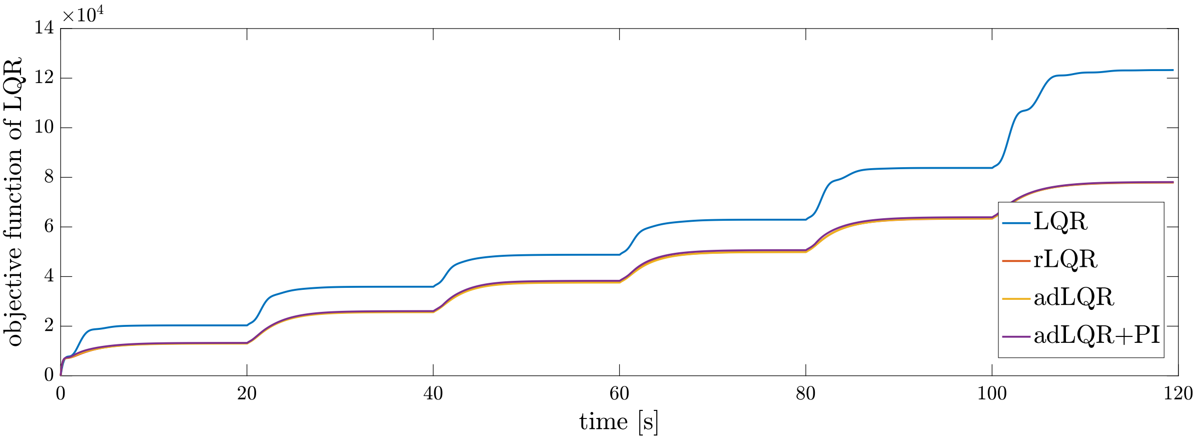

Figure 5.13.: *Objective function of the LQR problem for the inverted pendulum on a cart for scenario 2-G controlled with different LQR-based control laws for a simulation horizon of* $T = 120\,\mathrm{s}$.

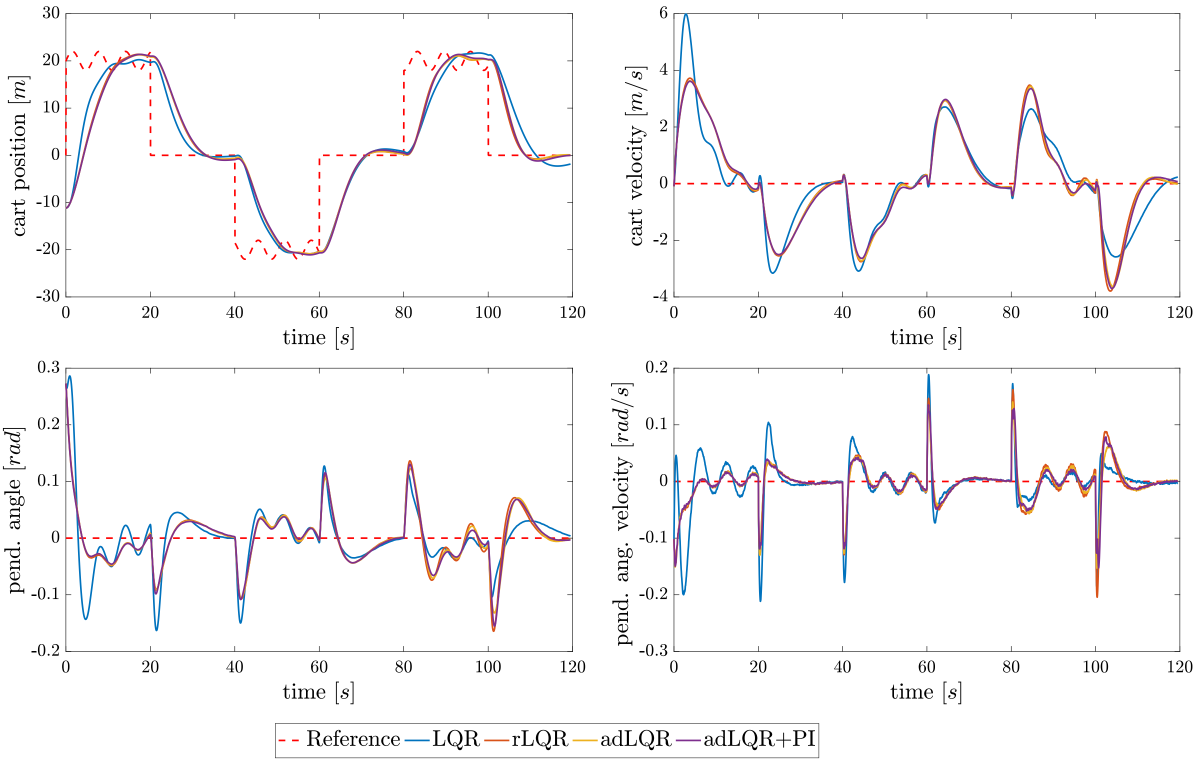

Figure 5.14.: *States of the inverted pendulum on a cart for scenario 4-F with additional FIM-check for a simulation horizon of* $T = 120\,\mathrm{s}$.

here is that a slight oscillation around the target position is introduced at positions $x_{1,ref} = 20$ and $x_{1,ref} = -20$ using a sine wave. Figure 5.14 shows the reference values as dashed red lines. It can be observed that all control approaches perform very well in following the references.

Figure 5.15 shows the trace of the inverse of the Fisher information matrix $\mathrm{tr}(\mathcal{I}_p^{-1})$ for the considered scenario. It can be observed that the values are significantly smaller compared to the previous scenario 2-G, indicating a higher information content of the trajectory. However, most of the time we are still above our threshold of $\delta = 0.2$ and thus no parameter identification is performed.

The violation of the threshold for the FIM can also be seen in the identified parameters in Figure 5.16. There, we show the identified parameters in blue and the parameters from the Algorithm 5.2 with FIM check again in green. The plateaus within the results with Algorithm 5.2 show that here, the information content of the data is considered too low for successful identification. Interestingly, in the results of the first algorithm without FIM-checks, where always a PI is performed, it can be seen that the trajectory allows good estimation results for the first parameter. The approximation of the second parameter can still be improved by the FIM checks, which avoid arbitrary estimates in periods with less information. To achieve this, we double the frequency of the sinusoid in scenario 5, see Figure 5.17. This gives a very good tracking result for all controllers

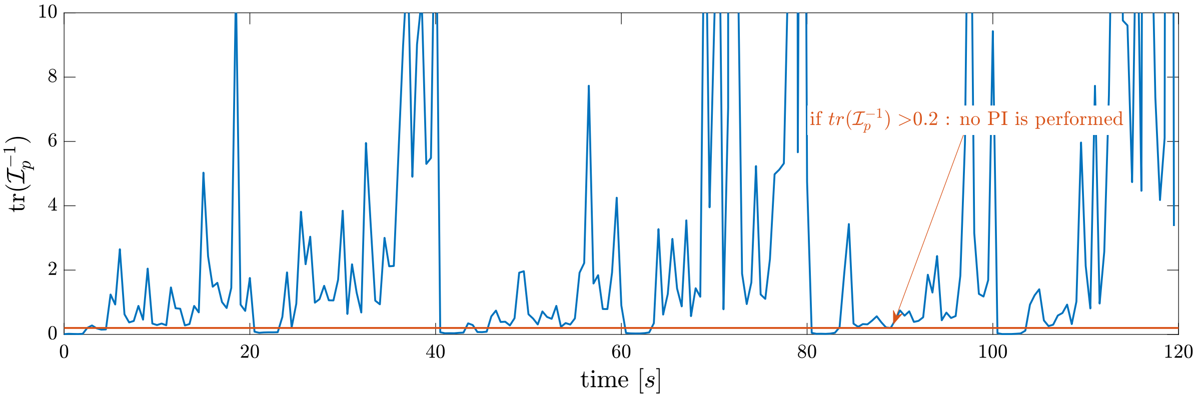

Figure 5.15.: *Trace of the inverse of the approximated Fisher information matrix $\mathcal{I}_p$ for scenario 4-F.*

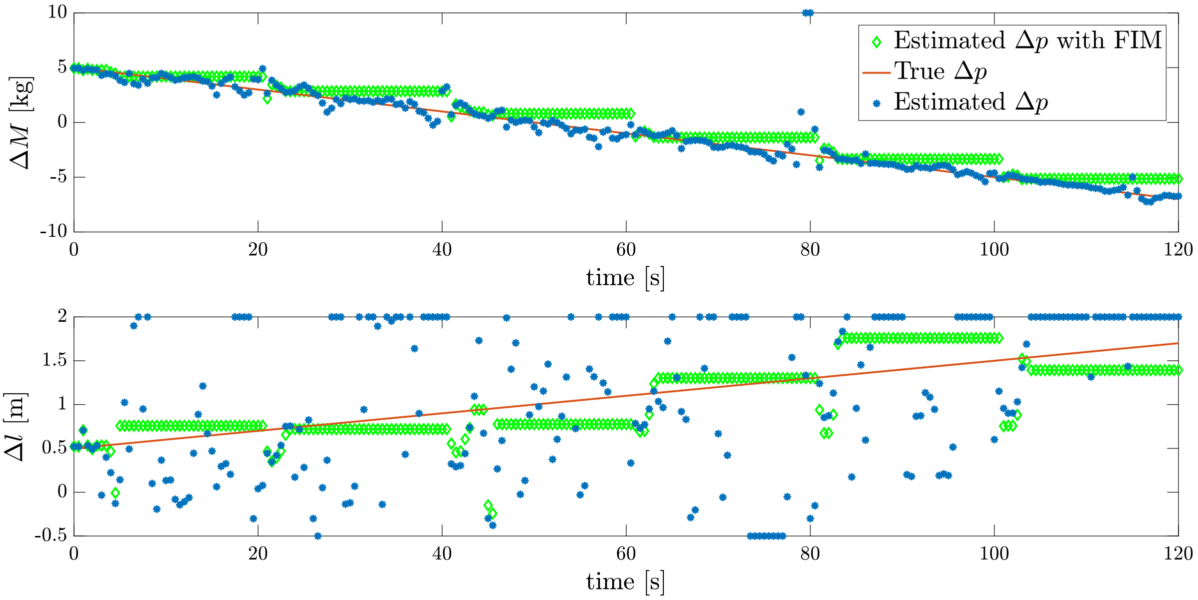

Figure 5.16.: *Comparison of parameter deviation of the inverted pendulum on a cart for scenario 4-F with and without additional FIM-check.*

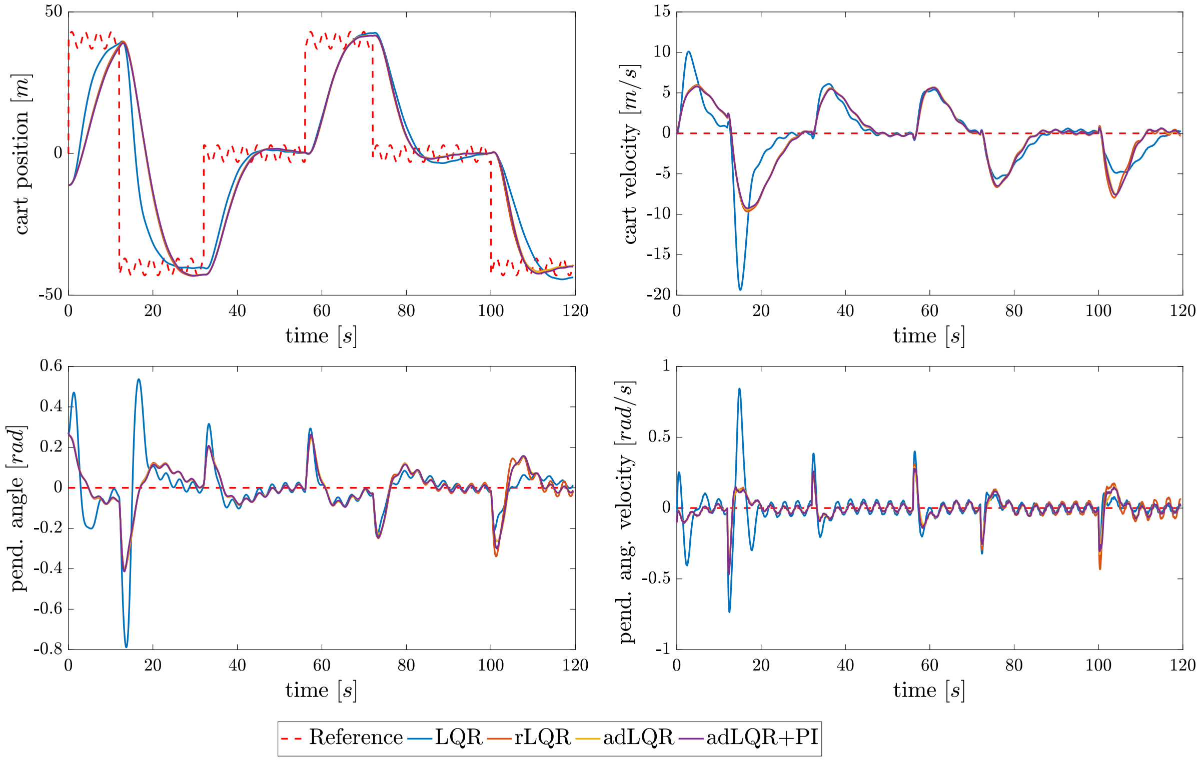

Figure 5.17.: *States of the inverted pendulum on a cart for scenario 5-F with additional FIM-check for a simulation horizon of* $T = 120\,\mathrm{s}$.

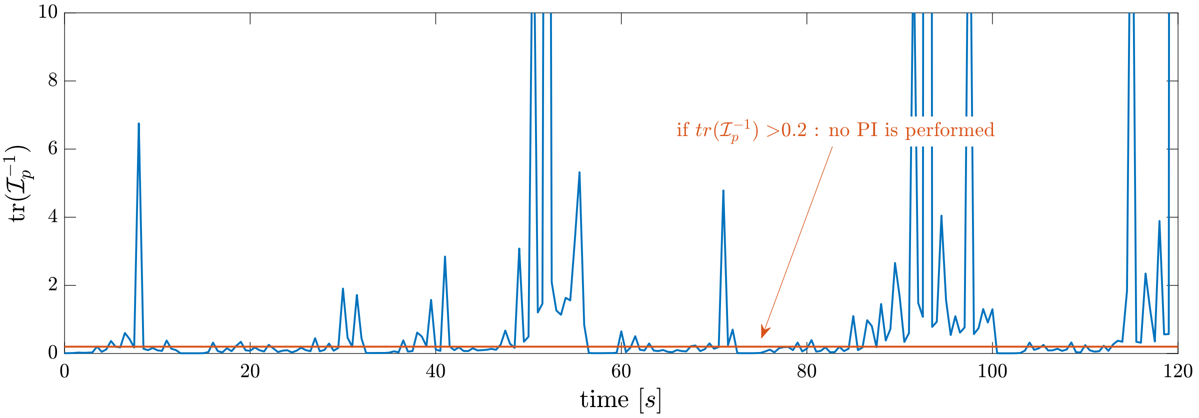

Figure 5.18.: *Trace of the inverse of the approximated Fisher information matrix* $\mathcal{I}_p$ *for scenario 5-F.*

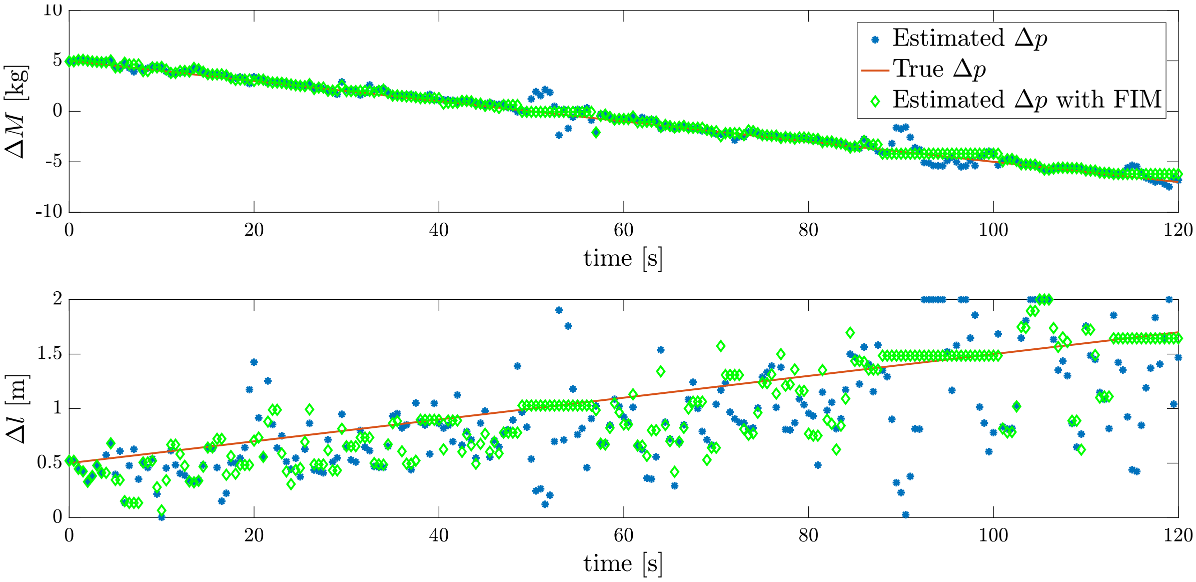

Figure 5.19.: *Comparison of parameter deviation of the inverted pendulum on a cart for scenario 5-F with and without additional FIM-check.*

used. Referring to Figure 5.18, it can be seen that the trace value of the inverted FIM is lower than in scenario 4-F. This means that the information content of the data is significantly higher.

Looking at the parameter estimates for scenario 5-F in Figure 5.19, we see that the parameter identification is better for both algorithms. The improvement in the identification result shows how much the identifiability of the parameters depends on the available data. There is almost no need to disable the parameter identification.

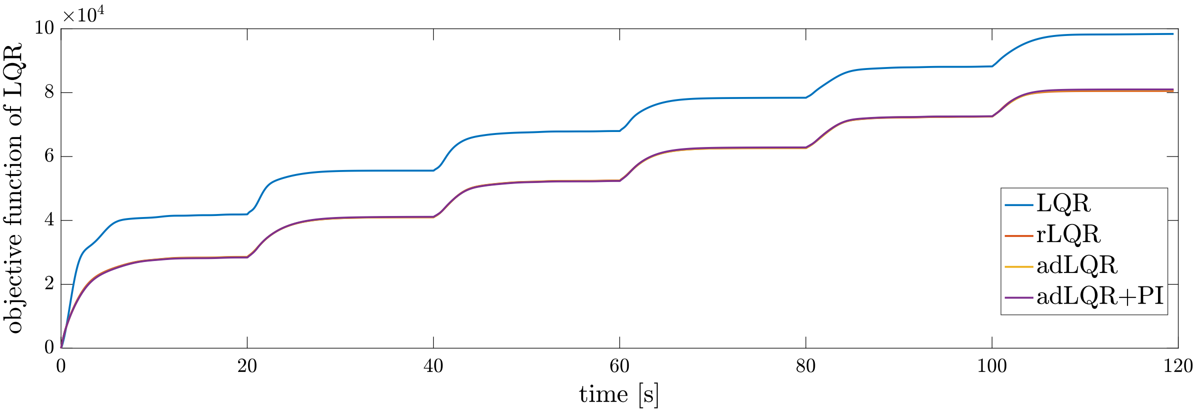

Figure 5.20.: *Objective function of the LQR problem for the inverted pendulum on a cart for scenario 4-F controlled with different LQR-based control laws for a simulation horizon of* $T = 120\,\mathrm{s}$.

This means that a suitable choice of trajectories can increase the information content.

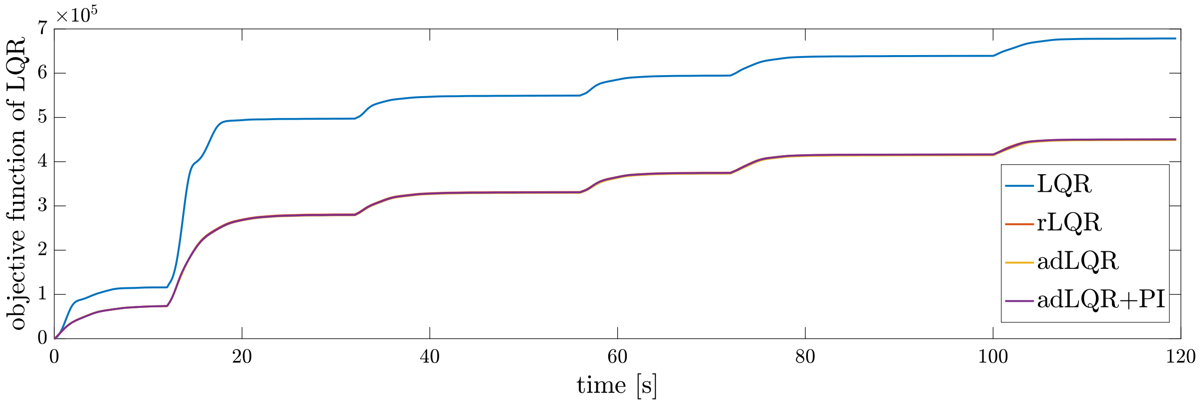

Figure 5.21.: *Objective function of the LQR problem for the inverted pendulum on a cart for scenario 5-F controlled with different LQR-based control laws for a simulation horizon of* $T = 120\,\mathrm{s}$.

If the choice is most suitable, the need to approximate the information content can become unnecessary. Thus, the additional effort for the FIM approximation can be avoided. However, suppose the trajectories are created without such trade-offs or even synthesized on operation. In that case, the identifiability should be evaluated by an approximation of the Fisher information matrix, as we have suggested.

The accumulated objective function values of the LQR problem F_{LQR} for scenarios 4-F and 5-F are shown in Figure 5.20 and Figure 5.21, respectively. It can be observed that the perturbation adapted controllers `rLQR`, `adLQR`, `adLQR+PI` and `adLQR+PI+FIM` have lower values than the classical `LQR`. In particular, our proposed algorithm performs as well as `rLQR` and `adLQR`, where the parametric perturbation is taken as known. The individual depicted values can be found in Table B.1 in Appendix B.

5.4.9. Conclusion of the Numerical Results

We define additional differences as performance values for a more concise evaluation and comparison of the different control methods for all analyzed scenarios. The objective function value of `adLQR` with the exact parameters is used as a benchmark. The differences are defined as

$$\begin{aligned}\xi_1 &= F_{\texttt{LQR}} - F_{\texttt{adLQR}},\\ \xi_2 &= F_{\texttt{adLQR+PI}} - F_{\texttt{adLQR}},\\ \xi_3 &= F_{\texttt{adLQR+PI+FIM}} - F_{\texttt{adLQR}},\end{aligned} \tag{5.21}$$

to compare the control laws `LQR`, `adLQR+PI` without and with FIM check. For this purpose, the relative objective function values ξ_1, ξ_2, and ξ_3 are listed in Table 5.1 for all considered scenarios. They are computed from the values from Table B.1 in Appendix B.

The best values for each combination of scenarios and parameters are highlighted in bold.

A look at the values shows that our approaches for online identification perform better than the non-adapted classical LQR in almost all cases. The latter becomes unstable in two cases, where $\xi_1 = \infty$. In almost 80% of the cases, `adLQR+PI+FIM` gives the best results concerning the value of the objective function. In about 20% of the of the scenarios, a better result is obtained without the additional FIM checks. It can be observed that these are mostly scenarios in which the trajectories can be regarded as sufficiently informative in large parts. The additional checks, if a parameter identification should be made, pay off, especially in the cases where there are longer areas with little information, like the x_{ref} in Scenario 3.

Further influencing factors like the sample time and the frequency of executed parameter identifications affect the quality of the algorithms. If the identifications follow each other closely with a relatively large amount of data, previously possibly error-prone identifications can be quickly compensated and make additional FIM-checks unnecessary. Whether additional FIM-checks should be used or not is a question of implementation effort, computational power, and the specific application. Overall, it can be said that `adLQR+PI` achieves very good results for this example.

Table 5.1.: *Relative objective function of the LQR problem for all scenarios for the inverted pendulum-cart system. The value in bold indicates the lowest value for each scenario.*

Scenario identifier		**Relative objective**			**Scenario identifier**		**Relative objective**		
x_{ref}	Δp	ξ_1	ξ_2	ξ_3	x_{ref}	Δp	ξ_1	ξ_2	ξ_3
0	A	164	3	**1**	3	A	2660	350	**70**
0	B	20370	**60**	**60**	3	B	70010	220	**140**
0	C	40	0	**-2**	3	C	640	640	**230**
0	D	298	11	**7**	3	D	1210	490	**160**
0	E	226	32	**29**	3	E	810	420	**10**
0	F	1888	60	**58**	3	F	10020	230	**-30**
0	G	2236	28	**27**	3	G	12360	150	**-60**
0	H	156	**-29**	-27	3	H	7390	290	**180**
0	I	1129	-12	**-14**	3	I	5380	740	**450**
1	A	2040	720	**360**	4	A	6380	640	**-30**
1	B	83310	230	**-110**	4	B	190000	900	**-100**
1	C	1150	1430	**520**	4	C	1440	**810**	1010
1	D	**620**	1253	1620	4	D	2450	420	**240**

Continued on next page

x_{ref}	Δp	ξ_1	ξ_2	ξ_3	x_{ref}	Δp	ξ_1	ξ_2	ξ_3
1	E	1740	810	**60**	4	E	1890	580	**200**
1	F	7370	630	**580**	4	F	17580	370	**200**
1	G	29170	340	**90**	4	G	49960	**1900**	2940
1	H	18570	390	**70**	4	H	37710	1140	**150**
1	I	5220	**1110**	3710	4	I	11140	1030	**820**
2	A	2630	430	**130**	5	A	50800	1200	**900**
2	B	155860	**160**	260	5	B	∞	**700**	1000
2	C	1930	2330	**1170**	5	C	10000	9100	**8800**
2	D	240	1440	**830**	5	D	6400	2000	**1600**
2	E	2390	720	**90**	5	E	16700	1400	**1000**
2	F	8900	**450**	1190	5	F	229600	**700**	1500
2	G	45320	640	**120**	5	G	∞	**600**	1400
2	H	33290	430	**0**	5	H	320700	**1200**	1700
2	I	6710	1890	**1140**	5	I	75900	1700	**1400**

6. Online Parameter Estimates for Nonlinear Optimal Feedback Control

Up to this point, online parameter identification in combination with linear feedback control was examined. Since we are considering nonlinear parameter identification in particular, the next step is the combination with nonlinear optimal feedback control. For this purpose, we consider nonlinear model predictive control as introduced in Chapter 4. NMPC can be interpreted as a feedback controller since it provides controls adapted to the current state through feedback with current system information and repeated optimization. In NMPC, a nonlinear dynamical model of the system to be controlled is used to predict the future behavior of the process as a function of the inputs. The performance of the procedure depends highly on the quality of this used model. Therefore, it is of particular interest to provide an accurate system model. Our approach of additional online parameter identification addresses this ambition. The initial design started with a sequential approach in [RFB18]. There, the parameter identification was carried out for a repeated movement of the system, and a Monte Carlo simulation showed significant improvement in the point tracking. The use of online parameter identification in the NMPC process was introduced in [RFB20]. This approach is the subject of further discussions in this chapter.

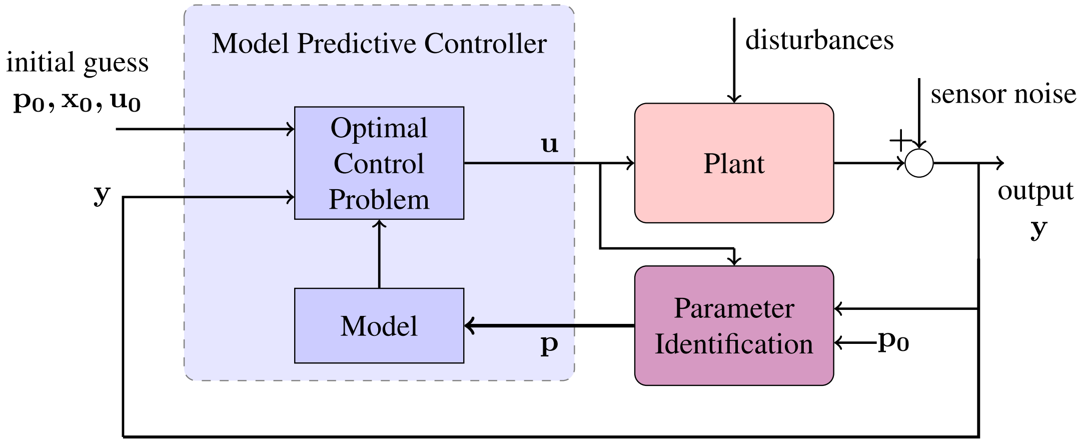

Figure 6.1.: *The control loop with NMPC and online parameter identification.*

Figure 6.1 is a structural representation of the proposed approach of NMPC and nonlinear dynamic parameter identification, denoted as `NMPC+PI` in the following. There, the classical NMPC control loop, introduced in Figure 4.1, is extended by an additional parameter identification. The PI for model adaption is added in the lower red block. It has the measured system outputs as input. The parameter identification then minimizes the error between the system output and the model output. The optimal parameters computed in this step are then fed to the system model in the lower blue block. In this way, the optimal control problem can be provided with a model that best adapts to the current conditions. Thus, the predictions in the next OCP can be improved. External disturbances are those that affect the behavior of the system. As a result, parameters in the system-describing model can change. Additionally, measurement noise, i.e., disturbances originating from sensors, is taken into account by an additive white Gaussian noise in the measurements. At each NMPC step, the output states of the system are measured. These measurements are then used as initial states for the new OCP. Since the focus of this work is on the identification of model parameters, we assume full-state feedback to avoid other side effects.

For the online parameter identification, we consider the data of a limited time window in the past. This *finite horizon approach* is also used for state estimation in NMPC schemata, e.g., moving horizon estimation (MHE), [Die+06; Fri+15]. The data horizon is limited and moved forward in time to keep the computational cost low. Since we assume that the parameters can change with time, we deliberately take into account that the knowledge from previous data is lost to a certain extent. However, the already gained information is indirectly used since the optimal solution of the previous problem is used as an initial estimate for the current problem.

6.1. The Parameter Identification Problem for NMPC

This section discusses the arising identification problems in more detail. Assume we are at the current process time t_k in our NMPC setup shown in Figure 6.2. It contains different time axes. For the general system, time t is used. With τ the time within the current OCP is described, as in the NMPC theory chapter, so an OCP always starts at $\tau = 0$. The time variable ξ is used within the NDPIP. The optimization problem for parameter estimation always starts with $\xi = 0$. During the operation of the system, measurements are taken. These samples y_i are collected at each time t_i, $i = 0, \dots, k$ with a fixed sample time $\Delta t = t_i - t_{i-1}$, $i = 1, \dots, k$. Using this sample of measurements, we could now set up a parameter identification problem as defined by 2.3.2. This problem, which uses all available data, is called a *full information problem* in the MHE context, Rao, Rawlings and Mayne [RRM03]. However, since this may produce a large amount of data in total, the computation time to solve such a problem would quickly increase with increasing operating time. These problems could no longer be solved online. To reduce the computational cost, we use only a limited number of

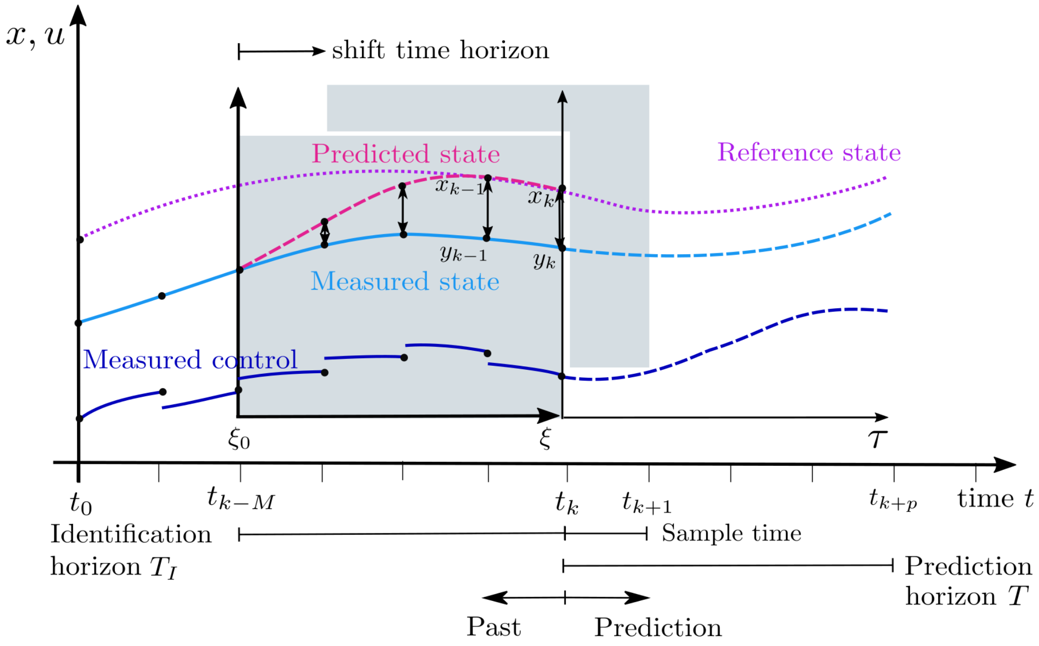

Figure 6.2.: *Scheme of the different time frames for the parameter identification problem for NMPC at system time t_k; an NDPIP is solved online for measurements from the past identification horizon.*

measurements for parameter identification. For a given number $M \in \mathbb{N}$ of measurements, we define the *identification horizon* as $T_I := M\Delta t$. Thus, only the measurements on the last period $[t_k - T_I, t_k] = [t_{k-M}, t_k]$ are used. This approach is advantageous in our case because we assume time-varying parameters, so only data near in the past is of interest for determining the current model parameters. The k-th NDPIP can be formulated on the interval $[t_{k-M}, t_k]$ as follows:

$$\begin{aligned}
\min_{x_0^{[k]}, p^{[k]}} \quad & F^{[k]}(x_0^{[k]}, p^{[k]}) := \sum_{i=0}^{M} \left(x(\xi_i; p^{[k]}, x_0^{[k]}) - y_{k-M+i} \right)^T W \left(x(\xi_i; p^{[k]}, x_0^{[k]}) - y_{k-M+i} \right) \\
\text{w.r.t.} \quad & \dot{x}(\xi) = f(x(\xi), \tilde{u}(\xi), p^{[k]}), \qquad \text{for } \xi \in [t_{k-M}, t_k], \\
& x(\xi_0) = x_0^{[k]}, \\
& g(\bar{x}(\xi), p^{[k]}) \leq 0.
\end{aligned} \tag{6.1}$$

The parametric dynamical model $f(x(\cdot), u(\cdot), p)$ of the considered plant is the same that is used in the optimal control problems for NMPC. Further y_{k-M} is the measurement at time $t_{k-M} = t_k - T_I$. By $\tilde{u}(\cdot)$, we denote an approximation of the controls at the discretization nodes. It is equal to the measured controls at each sample point, and between two samples, it is the result of a linear interpolation of the two neighboring samples.

Algorithm 6.1 `NMPC+PI`

```
At time $t_0$ choose nominal parameter $p^{[0]} := p_0$.
Choose a prediction horizon $T$ for the OCP, an identification horizon $T_I$, and a threshold $\delta$.
Set iterator $k = 0$.
Measure the current system state $y_k$.
if $k \geq M$ and $k \bmod M = 0$ and $\sum_{j=k-M}^{k} \frac{1}{M} \|x(t_j) - y_j\|^2 > \delta$ then
    Perform parameter identification (6.1) for $p_k$ on $[t_k - T_I, t_k]$ and
    set the parameters of the model used in the OCP to $p_k = arg\,min$ (6.1).
else Set the parameters of the model used in the OCP to $p_k = p_{k-1}$ for $k > 0$.
end if
Set the initial state of the OCP to $\bar{x}_0 = \bar{x}(\tau_0 = 0) := y_k$ and solve (4.9) on $[t_k, t_k+T]$.
Denote the obtained optimal control sequence $\bar{u}^*(\cdot) \in U$ and apply its first sequence for the sample time $\Delta t$ as control to the system.
Set $k = k + 1$ and continue with step 4.
```

Additionally, general equality or inequality constraints on the states and parameters are considered in the function g. In essence, parameters are sought so that the error between model output and measured output on the last identification horizon is as small as possible for the applied controls. With the weighting matrix $W = \operatorname{diag}(w_1, \ldots, w_{n_x}) \in \mathbb{R}^{n_x \times n_x}$ in the objective function, the influence of each state can be weighted. If it can be assumed that the distributions of the measurement uncertainties are known, then a skillful choice of the weights as the inverted covariance matrix $W = \Sigma^{-1}$ can be useful.

The next step is to formulate the algorithm that combines nonlinear model predictive control with nonlinear dynamic parameter identification.

6.2. The Combined Algorithm `NMPC+PI`

The entire process for the described algorithm `NMPC+PI` is summarized in Algorithm 6.1. The proposed method extends the classical NMPC Algorithm 4.1 by the option of online nonlinear parameter identification. We initialize the algorithm by choosing an initial guess for the parameters $p^{[0]} := p_0 \in \mathbb{R}^{n_p}$, an identification horizon $T_I \in \mathbb{R}$ for data sampling, and a longer prediction horizon $T \in \mathbb{R}$ with $T > T_I$. The identification horizon is a multiple of the sample time, i.e., $T_I = M\Delta t$, so that there are $M \in \mathbb{N}$ measurements available for parameter identification. The first interval to consider for parameter identification is then $I^{[0]} = [t_0, t_0 + T_I]$. The if-condition in Algorithm 6.1 specifies when a parameter identification will be made. The first condition ensures that we have collected at least M measurements to provide sufficient data. The second condition ensures that a parameter identification is only performed if M new measurement data are available. Further, the third condition estimates the deviation of the current trajectory from the previously predicted one. For this purpose, the multidimensional *mean square error (MSE)* between the predicted states and measurements is computed. The value is then compared to a previously defined threshold $\delta \in \mathbb{R}^+$, that represents the largest accepted deviation:

$$\mathrm{MSE}_x \coloneqq \sum_{i=1}^{n_x} \frac{1}{M} \sum_{j=k-M}^{k} \|x_i(t_j) - y_{i,j}\|^2 > \delta.$$

If this is exceeded, a parameter identification of the type (6.1) is performed. The newly obtained optimal estimates of the model parameters are used to adapt the model in the OCP. Subsequently, the optimal control problem is solved using the adapted system model. The obtained control sequence is then realized on the system, and the procedure continues with taking new measurements. As a reminder, the notation $\bar{x}$ and $\bar{u}$ are used for the states and controls of the k-th optimal control problem with the problem internal time variable τ.

The optimal control problems are solved with the transcription software TransWORHP, see Subsection 4.2.3. A direct approach via the full discretization method with a trapezoidal method is used here. The resulting NLPs are then solved with WORHP. For the first OCP, the initial guess for the states at each discretization point is generated by linear interpolation between the initial and target states. For the controls, zero is used as the initial guess. Afterwards, we use a shift-initialization strategy for the initial guess, see Subsection 4.5. For this, the solution trajectories of the previous OCP are used as initial guesses for both the states and the controls. The last section, which does not yet contain any approximate values, is filled with a constant value equal to the last value of the previous OCP solution. The derivative structures of the objective function and the Jacobian are provided to accelerate the computation of the solution. The derivatives are then computed with WORHP using finite differences. To further exploit the similarity of successive OCPs for efficient computation, starting from the second OCP, the obtained Lagrangian multipliers as part of the solution of the previous problem are provided as initial guesses of the Lagrangian multipliers for the current problem.

TransWORHP is also used to solve the parameter identification problems. Thus, the differential equations in the constraints are handled by full discretization. For this, a trapezoidal method is used. In order to increase the efficiency, we provide the derivative of the objective function together with the derivative structure of the differential equations. The numerical computation of the derivatives is then done by WORHP with a BFGS approach. The measurements are used as a reasonable initial guess for the states. For this purpose, the corresponding measured value or a linear interpolation of the two neighboring measurements is used for each discretized state. The solutions to the previous problem are taken as an initial guess for the parameters.

In the following, the performance of the algorithm is analyzed with two numerical examples. First, we consider the inverted pendulum-cart system in two different task scenarios with unknown constant parameters and dynamic larger parameter deviations. Second, we look at a state-tracking task of a robotic manipulator with unknown model parameters. For the following numerical evaluations, the proposed `NMPC+PI` algorithm was implemented in the programming language C++. The simulations of the real systems are also realized within C++. For this, the dynamical systems with the true parameters are integrated forward with an advanced integration method, i.e., a Runge-Kutta scheme of fourth order and a fine discretization. The numerical results show an improved stability behavior of the nonlinear model predictive controller with the additional parameter identifications compared to classical NMPC.

6.3. Numerical Analysis on the Inverted Pendulum on a Cart

We consider again the pendulum-cart system from Section 5.4. In the previous Chapter 5 we controlled this system with different linear controllers and showed that additional online parameter identifications lead to improved control behavior and reduced control cost. Now, we examine our nonlinear model predictive control algorithm with online model adaptations, `NMPC+PI`, on the same example. We examine what advantages and disadvantages our proposed algorithm `NMPC+PI` has compared to the classical NMPC without model fitting, denoted by `NMPC` in the following.

6.3.1. The OCP for the Inverted Pendulum on a Cart

The optimal control problem to be solved within the kth-NMPC step is formulated for the pendulum-cart system as follows:

$$
\begin{aligned}
\min_{\bar{x}^{[k]},\bar{u}^{[k]}} \quad J^{[k]} &= \int_0^T (\bar{x}^{[k]}(\tau;p^{[k]}) - x^{ref}(\tau))^T Q(\tau)(\bar{x}^{[k]}(\tau;p^{[k]}) - x^{ref}(\tau)) \cdots \\
&\qquad \cdots + \bar{u}^{[k]}(\tau)^T R(\tau)\bar{u}^{[k]}(\tau) \; \mathrm{d}\tau \\
\text{w.r.t. } \dot{\bar{x}}^{[k]}(\tau) &= f(\bar{x}^{[k]}(\tau), \bar{u}^{[k]}(\tau), p^{[k]}) \\
&= \begin{pmatrix}
\bar{x}_2^{[k]}(\tau) \\
\frac{1}{D^{[k]}}(\bar{u}^{[k]}(\tau) + l^{[k]}\bar{x}_4^{[k]^2}(\tau)\sin(\bar{x}_3^{[k]}(\tau)) - g\sin(\bar{x}_3^{[k]}(\tau))\cos(\bar{x}_3^{[k]}(\tau))) \\
\bar{x}_4^{[k]}(\tau) \\
\frac{g}{l^{[k]}}\sin(\bar{x}_3^{[k]}(\tau)) - \frac{\cos(\bar{x}_3^{[k]}(\tau))}{l^{[k]}D^{[k]}}\big(\bar{u}^{[k]}(\tau) + l^{[k]}\bar{x}_4^{[k]^2}(\tau)\sin(\bar{x}_3^{[k]}(\tau)) \cdots \\
\cdots - g\sin(\bar{x}_3^{[k]}(\tau))\cos(\bar{x}_3^{[k]}(\tau))\big)
\end{pmatrix} \\
\bar{x}^{[k]}(0) &= y_k, \\
\bar{x}^{[k]}(\tau) &\in [(-1000,-100,-1,-1)^T, (1000,100,1,1)^T], \\
\bar{u}^{[k]}(\tau) &\in [-500,500], \text{ for } \tau \in [0,T],
\end{aligned}
\tag{6.2}
$$

where it is $D^{[k]} = M^{[k]} + \sin^2(\bar{x}_3^{[k]}(\tau))$. The constraints ensure that the nonlinear system dynamics (5.14) are satisfied on the considered time horizon. Furthermore, the initial state is set to the current system state, which is taken from the current measurement y_k. Box constraints take into account the physical limitations of states and controls. In the optimal control problem (6.2), the weighting matrices $Q(\tau)$ and $R(\tau)$ are used to provide a trade-off between the tracking of individual states and the control effort, also in the temporal direction. After discretization in time and transcription of the control problem into an NLP, we choose the weights for each discretized state $x(\tau_i)$ with $i = 0, \dots, N-1$, to

$$Q(\tau_i) = Q_i = \frac{1}{(N-i)^2} \begin{pmatrix} 10 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 10 \end{pmatrix} \text{ and } R(\tau_i) = 0.1,$$

where $N \in \mathbb{N}$ is the number of discretization points. The changing weights for the states over time have the effect that deviations from the desired state are more influential at later times than at earlier points. This enables the system to reach the target state at the end. So, we are in the setting of the Section 4.4.2 with an objective function like the one presented in (4.8).

6.3.2. The NDPIP for the Inverted Pendulum on a Cart

As in Section 5.4 the parameter vector is defined as $p := (M, l)^T \in \mathbb{R}^2$ and the nominal value is set to $p^{[0]} = p_0 = (10, 1)^T$. The nonlinear dynamical parameter identification problem at time t_k is formulated with $D^{[k]} = M^{[k]} + \sin^2(x_3(\xi))$ as:

$$\begin{aligned} \min_{x_0^{[k]}, p^{[k]}} \quad & F^{[k]} := \sum_{i=0}^{M} \left(x(\xi_i; p^{[k]}, x_0^{[k]}) - y_{k-M+i} \right)^T W \left(x(\xi_i; p^{[k]}, x_0^{[k]}) - y_{k-M+i} \right) \\ \text{w.r.t.} \quad & \dot{x}(\xi) = f(x(\xi), \tilde{u}(\xi), p^{[k]}), \text{ for } \xi \in [t_k - T_I, t_k], \\ & = \begin{pmatrix} x_2(\xi) \\ \frac{1}{D^{[k]}} \left(\tilde{u}(\xi) + l^{[k]} x_4^2 \sin(x_3(\xi)) - g \sin(x_3(\xi)) \cos(x_3(\xi)) \right) \\ x_4(\xi) \\ \frac{g}{l^{[k]}} \sin(x_3(\xi)) - \frac{\cos(x_3(\xi))}{l^{[k]}(D^{[k]}} \big(\tilde{u}(\xi) + l^{[k]} x_4(\xi)^2 \sin(x_3(\xi)) \cdots \\ \cdots - g \sin(x_3(\xi)) \cos(x_3(\xi)) \big) \end{pmatrix} \qquad (6.3) \\ & x(\xi_0) = x_0^{[k]}, \\ & \begin{pmatrix} -5.0 \\ -0.5 \end{pmatrix} \leq \Delta p^{[k]} \leq \begin{pmatrix} 10.0 \\ 2.0 \end{pmatrix}. \end{aligned}$$

For the weighting matrix $W = \text{diag}(w_1, w_2, w_3, w_4) \in \mathbb{R}^{n_x \times n_x}$ in the objective function all entries are initially set to $w_i = 1$, $i = 1, 2, 3, 4$, and thus all states are weighted identically. In the constraints, lower and upper bounds are specified for the parametric perturbations $\Delta p^{[k]} = (\Delta M^{[k]}, \Delta l^{[k]})^T$. Since it is $p^{[k]} = p_0 + \Delta p^{[k]}$ with $p_0 = (10, 1)^T$, these constraints also indirectly provide lower and upper bounds for the parameters. The controls in the dynamics are an approximation from the measured controls at the discrete time points described for problem (6.1).

6.3.3. Scenario 0-A: Constant Small Parameter Deviation

First, we consider scenario 0-A from (5.18) and (5.19). This corresponds to a small constant parameter perturbation $\Delta p = (2.0, 0.5)^T$ and the stabilization of the origin. We simulate the system for $T_s = 40\,\mathrm{s}$ with a sample time $\Delta t = 0.05\,\mathrm{s}$. In standard nonlinear model predictive control, an optimal control problem must be solved online in each NMPC step. The prediction horizon of each OCP is $T = 10\,\mathrm{s}$. We choose a multistep-NMPC approach and solve a new OCP not for every incoming measurement but one for each 0.2 seconds.

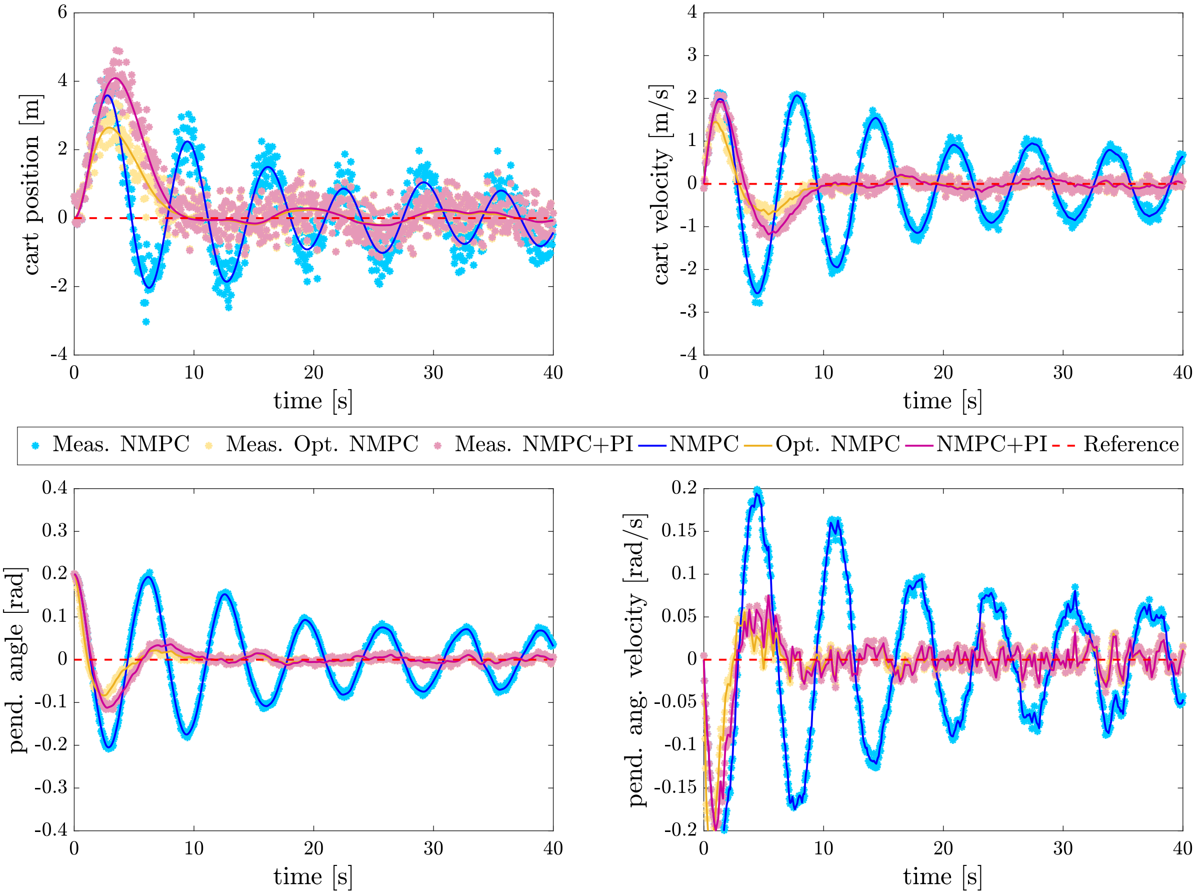

Figure 6.3.: *States of the inverted pendulum on a cart for scenario 0-A controlled by classic* `NMPC` *(blue line),* `NMPC+PI` *(violet line),* `optimal NMPC` *(yellow line) and the corresponding noisy measurements (dots in similar color); cart position, cart velocity, pendulum deflection angle, and pendulum angular velocity for a simulation horizon of* $T = 40s$.

Thus, the control horizon is $T_C = 0.2\,\mathrm{s}$. This interval size allows the solution of an OCP since the computation times for an optimal control problem in this scenario are, on average, 35 ms. Further, it is not too conservatively chosen to account for incoming feedback sufficiently. The identification horizon is set to $T_I = 1\,\mathrm{s}$. Thus, we obtain 21 measurements per parameter identification.

Figure 6.3 shows the plots of the states of the pendulum-cart system controlled by different NMPC methods. The first plot on the top left shows the cart position. The second graph at the top right shows the cart velocity. Further, the third and fourth plots depict the pendulum's angular deflection and angular velocity over time. To simulate the real system, the nonlinear dynamical system with the true parameters is integrated forward for each NMPC step with a Runge-Kutta method of fourth order.

We compare the control behavior of three different approaches. First, the classical `NMPC` controller is used, which only knows the nominal parameter values. This is shown in blue. Further, the proposed algorithm `NMPC+PI` is plotted in violet. The `optimal NMPC` controller, which has the true model with the true parameters available at any time, serves as a further comparison controller and is given in yellow. It can be observed that all three controllers can stabilize the system over time. However, the classic `NMPC` needs considerably longer and, as we will see in the following, also needs considerably more control effort. The proposed `NMPC+PI` controller stabilizes the system very quickly. Initially, it still agrees with `NMPC`, but with several parameter identifications, it approaches the `optimal NMPC` and then performs comparably well. For further illustration, the measured data is additionally plotted as points. Depending on the method, a similar color shade is used. The measurement data result from the true simulated states of the system with a measurement noise implemented as additive normally distributed noise as described in (5.16). It is $\Sigma = \mathrm{diag}(\sigma_1^2, ..., \sigma_4^2)$ with the standard deviation vector $\sigma = (0.4, 0.09, 0.004, 0.004)$.

In Figure 6.4, the used controls are depicted. The classical `NMPC` is again plotted in blue, `NMPC+PI` in violet, and the `optimal NMPC` in yellow. We can observe that the two trajectories of `NMPC` and `NMPC+PI` are identical until the first parameter identification. After that, the control effort can be reduced significantly by adjusting the model used in the OCP in `NMPC+PI`. Already from $t = 6\,\mathrm{s}$, the applied control maneuvers are mainly only caused by the measurement noise. We use the `optimal NMPC` as a comparative value. Then, the classical `NMPC` requires about 150% more control effort than `optimal NMPC`, whereas our approach `NMPC+PI` requires only 12% more than the optimal controller:

$$\frac{\|u^{\mathtt{NMPC}}\|_2}{\|u^{\mathtt{opt.NMPC}}\|_2} = 2.4951 \quad \text{and} \quad \frac{\|u^{\mathtt{NMPC+PI}}\|_2}{\|u^{\mathtt{opt.NMPC}}\|_2} = 1.1158.$$

The computation times for the additional parameter identifications are, on average, 11 milliseconds. The maximum computation time that occurs is 22 milliseconds. This makes the combined method `NMPC+PI` real-time capable.

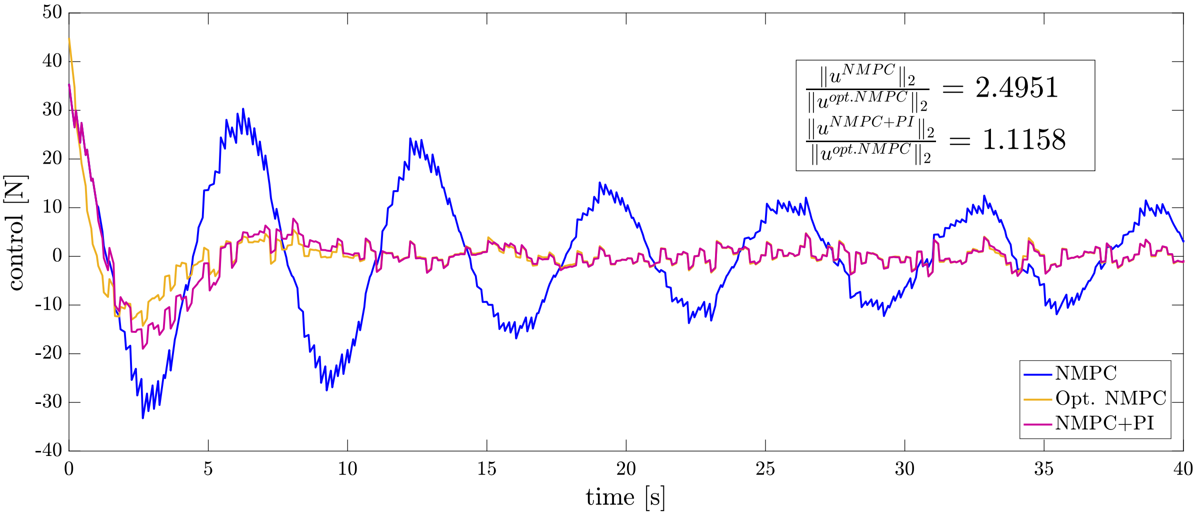

Figure 6.4.: *Control of classic* `NMPC` *(blue line),* `NMPC+PI` *(violet line) and* `optimal NMPC` *(yellow line) for the inverted pendulum on a cart for scenario 0-A for a simulation horizon of* $T = 40\,\mathrm{s}$.

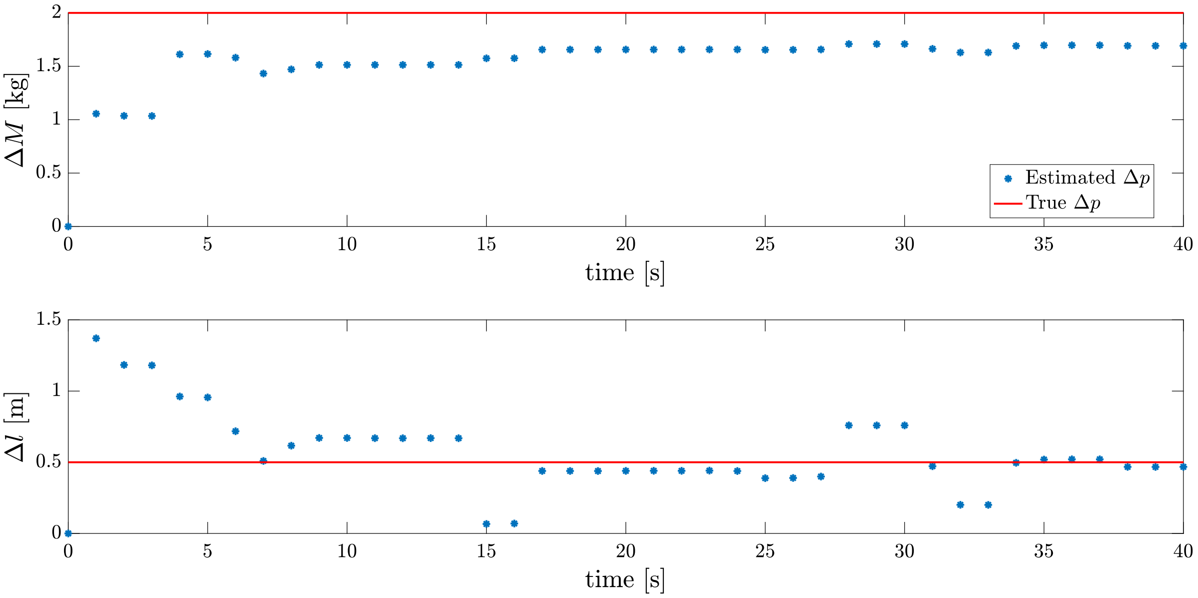

Figure 6.5.: *Identified parameter variations* Δp *(blue dots) of the inverted pendulum on a cart for scenario 0-A controlled with* `NMPC+PI` *for a simulation horizon of* $T = 40\,\mathrm{s}$.

In Figure 6.5, the identified parametric perturbations Δp are shown as blue dots for each parameter identification. The red line represents the true parameter perturbations used within the system simulation. The parameter estimates are close to the true values, although only 21 measurements per PI are available in this example. The root mean square error (RMSE) is:

$$\mathrm{RMSE}_{\Delta p} := \sqrt{\frac{\sum_{i=1}^{M}(\Delta p_i - \Delta p_i^{ref})^2}{M}} = \begin{pmatrix} 0.7940 \\ 0.4230 \end{pmatrix}.$$

In this example, $N = 21$ discretization points are used within the full discretization method to solve the parameter identification problem. The number may be selected differently. If necessary, the measurement data will be approximated at the support points. This is done by linear interpolation of the two surrounding measurements.

For the first scenario 0-A, we demonstrated that the model adaptation in the optimal control problem significantly improves the control behavior of the NMPC method. Considering a larger parametric perturbation in the model, e.g., scenario 0-B, the classical NMPC can no longer stabilize the perturbed system. The proposed combined algorithm, in contrast, shows good performance here as well. The corresponding diagrams for these tests can be found in Appendix B.2. The following presents the even more challenging problem of time-variable parametric perturbations.

6.3.4. Scenario 2-G: Large and Time-Varying Parameter Deviation

The next scenario that will be considered is scenario 2-G from (5.18) and (5.19). This represents a task where the cart continuously moves back and forth between the states $x_{ref} = (0, 0, 0, 0)^T$ and $x_{ref} = (-20, 0, 0, 0)^T$, and the model parameters are changing over time. For this scenario, we compare the adaptive `NMPC+PI` controller only with the optimal `optimal NMPC` controller since the classical `NMPC` is not able to stabilize the system. A graph of the states with the classical `NMPC` can be found in Appendix B.3. Figure 6.6 shows the states of the inverted pendulum on a cart for the two other control approaches. The four plots contain again the curves of the cart position, its velocity, the pendulum deflection, and the angular velocity of the pendulum over time. It can be observed that the state tracking with the proposed `NMPC+PI` approach works very well, and the trajectories are hardly distinguishable from those of the `optimal NMPC`. In Figure 6.7 the control effort of the two controllers is shown. This also highlights the excellent performance of the presented adaptive approach. The resulting quotient of the total control effort is:

$$\frac{\|u^{\texttt{NMPC+PI}}\|_2}{\|u^{\texttt{opt.NMPC}}\|_2} = 1.0718.$$

Only about 7% more control effort is required than for the optimal controller. The estimates of the parametric deviations Δp are shown in blue in Figure 6.8. In addition,

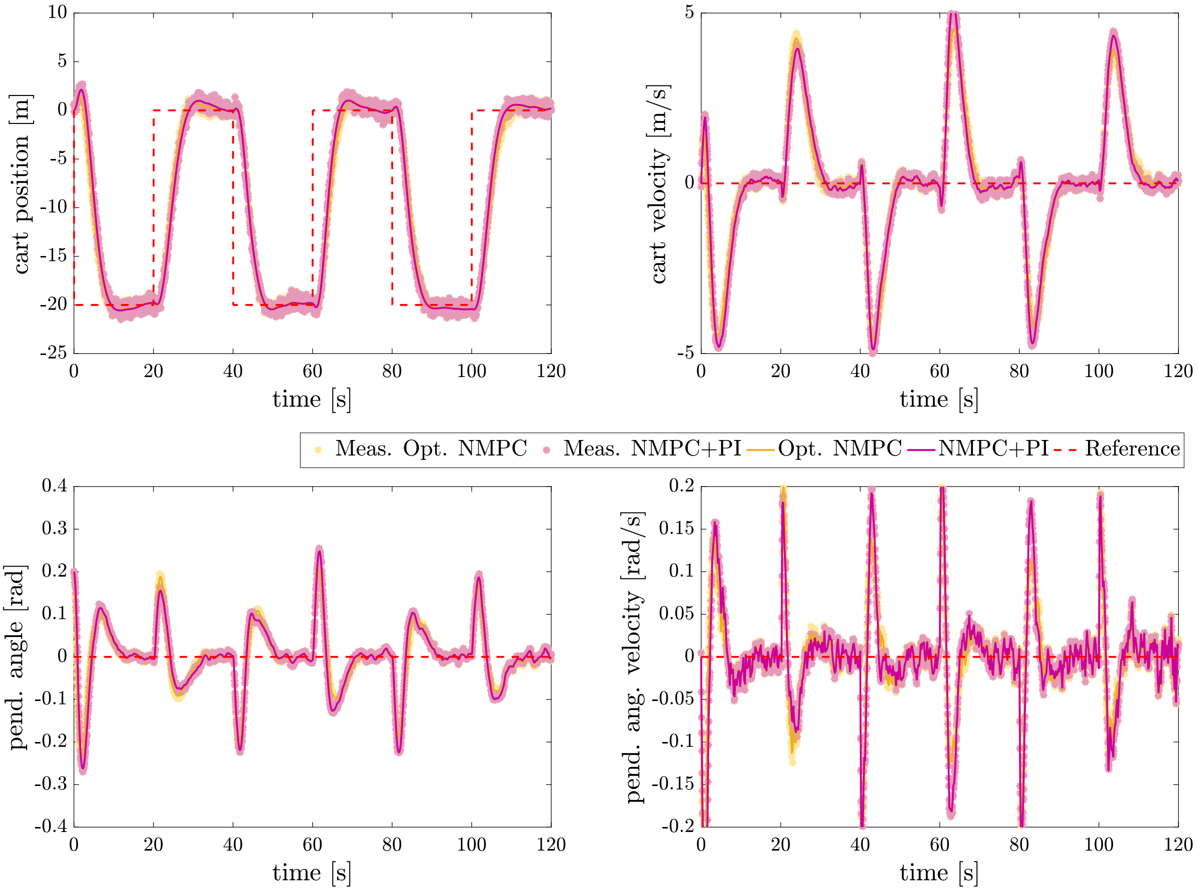

Figure 6.6.: *States of the inverted pendulum on a cart for scenario 2-G controlled by* NMPC+PI *(violet line) and* optimal NMPC *(yellow line) and the corresponding noisy measurements (dots in similar color); cart position, cart velocity, pendulum deflection angle, and pendulum angular velocity for a simulation horizon of* $T = 120\,\mathrm{s}$.

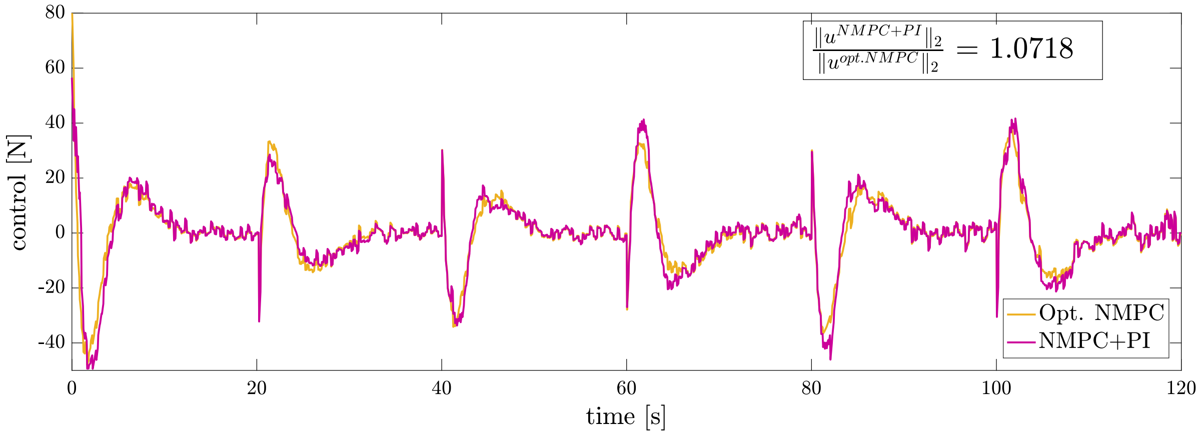

Figure 6.7.: *Control of* NMPC+PI *(violet line) and* optimal NMPC *(yellow line) for the inverted pendulum on a cart for scenario 2-G for a simulation horizon of* $T = 120\,\mathrm{s}$.

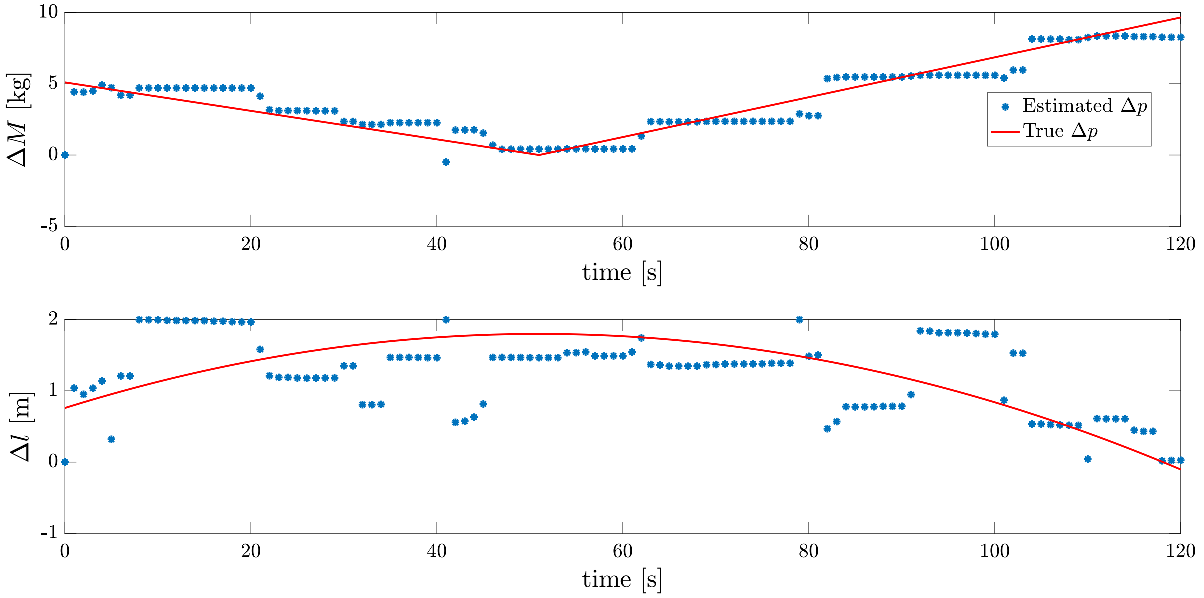

Figure 6.8.: *Identified parametric variations Δp (blue dots) of the inverted pendulum on a cart for scenario 2-G controlled with `NMPC+PI`, reference values (red line).*

the true parametric deviations of the system are given as reference values in red. The initial values $\Delta M = \Delta l = 0$ were used in each case. It can be seen that both parameters are satisfactorily identified. The root mean square error is:

$$\mathrm{RMSE}_{\Delta p} := \sqrt{\frac{\sum_{i=1}^{M}(\Delta p_i - \Delta p_i^{ref})^2}{M}} = \begin{pmatrix} 0.8991 \\ 0.5168 \end{pmatrix}.$$

Smaller outliers in the parameter estimates can appear once a satisfactory stabilization of the system occurs. As a result, there is less activity in the system, and the data for parameter identification also contains less information. For such cases, a measure of the information content of the data, such as the FIM, would again be useful. However, the additional computation time would be too much of a delay in NMPC. Nevertheless, these outliers have little effect on the overall behavior of the method since frequent PIs enable another fast adaptation to be performed. The predictions are significantly improved by tuning the model in the OCPs with the estimated parametric perturbations. This results in an improved controller performance in state tracking.

6.4. Numerical Analysis on a Robotic Manipulator

In this section, we demonstrate the functionality of the presented method using a second example of an industrial robotic manipulator. First, we explain the general approach to model a robot manipulator with n joints. For the following numerical analysis of our algorithms, we apply the proposed controller to a robot with two links. For this, we formulate the specific optimal control and parameter identification problems. The performance of the controller `NMPC+PI` is discussed for a state-tracking task in terms of tracking accuracy and identification quality in the embedded parameter estimation process. Furthermore, we compare the control performance of `NMPC+PI` to classical `NMPC` and `optimal NMPC`.

6.4.1. Dynamical Model for a Robotic Manipulator

A typical industrial robotic manipulator consists of a control unit and a manipulator arm that is built of separated links and has an end-effector on the top, where tools can be mounted. The links are connected through rotatory joints. Building a mathematical model of this multi-body system with rigid links connected through rotational movements ends up in an open-kinematic chain.

Two commonly used methods exist for the representation of the equations of motion of robotic systems in joint-space formulation, the Newton-Euler formulation and the Lagrange formulation, see Chapter 2 in Siciliano et al. [Sic08]. In this thesis, the energy-based Lagrangian method is chosen. For a robotic manipulator with $n \in \mathbb{N}$ links we define the joint angles $\theta = (\theta_1, \dots, \theta_n)^T$ as generalized coordinates, their corresponding joint angular velocities $\dot{\theta} = (\dot{\theta}_1, \dots, \dot{\theta}_n)^T$ are the derivatives and $i \in \{1, \dots, n\}$ is the index of the joint. The Lagrangian L is defined as the difference between the kinetic energy T and the potential energy V of the system, i.e.,

$$\begin{aligned} L(\theta, \dot{\theta}) = T(\theta, \dot{\theta}) - V(\theta) &= \sum_{i=1}^{n} T_i - \sum_{i=1}^{n} V_i \\ &= \tfrac{1}{2} \sum_{i=1}^{n} m_i \|\dot{S}_i\|^2 + \tfrac{1}{2} \sum_{i=1}^{n} \omega_i^T I_i \omega_i - \sum_{i=1}^{n} m_i g S_{i,z}, \end{aligned} \tag{6.4}$$

where $S_i = (S_{i,x}, S_{i,y}, S_{i,z})^T$ is the position of the center of mass of link i relative to the base frame, m_i is the mass of link i, ω_i is the angular velocity of link i relative to the base frame, I_i is the inertia tensor of link i and $g = 9.81\,\mathrm{m\,s^{-2}}$ the standard gravitational constant. For a detailed derivation of the dynamics, we refer to Murray et al. [Mur+94]. The equations of motion for the mechanical system are given by:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}} - \frac{\partial L}{\partial \theta} = h_f, \tag{6.5}$$

where $h_f = (h_{f,1}, \dots, h_{f,n})$ is the vector of external forces and $h_{f,i}$ is the force that acts on the i-th generalized coordinate. Examples are moments that act as external controls on the motors or friction effects. To briefly depict how we obtain the matrices that

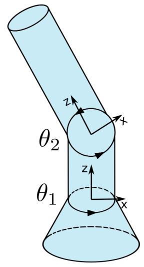

Figure 6.9.: *Two link open-chain robotic manipulator with two degrees of freedom.*

represent our dynamics, we insert the above-defined Lagrangian function (6.4) in (6.5). For each coordinate $i = 1, \dots, n$, it follows that,

$$
\begin{aligned}
h_{f,i} &= \frac{d}{dt}\frac{\partial T}{\partial \dot{\theta}_i} - \frac{\partial T}{\partial \theta_i} + \frac{\partial V}{\partial \theta_i} = \sum_{j=1}^{n} \frac{\partial^2 T}{\partial \dot{\theta}_j \partial \dot{\theta}_i} \ddot{\theta}_j + \sum_{j=1}^{n} \frac{\partial^2 T}{\partial \theta_j \partial \dot{\theta}_i} \dot{\theta}_j - \frac{\partial (T - V)}{\partial \theta_i} \\
&=: \sum_{j=1}^{n} M_{ij}(\theta)\ddot{\theta}_j - v_i(\theta, \dot{\theta}) - w_i(\theta).
\end{aligned}
$$

Finally, the equations of motion can be written in the common compact form:

$$
M(\theta)\ddot{\theta} = v(\theta, \dot{\theta}) + w(\theta) + h_f =: F(\theta, \dot{\theta}) + h_f. \tag{6.6}
$$

The symmetric and positive definite matrix $M(\theta) \in \mathbb{R}^{n \times n}$ is called the manipulator *inertia matrix* or *mass matrix*, the vector $v(\theta, \dot{\theta}) \in \mathbb{R}^n$ describes the centrifugal and Coriolis forces and the vector $w(\theta) \in \mathbb{R}^n$ represents the gravitational part. We summarize them in the force vector $F(\theta, \dot{\theta}) \in \mathbb{R}^n$. Note that the developed ordinary differential equations of second order in (6.6) show a highly nonlinear behavior due to nested trigonometric functions.

6.4.2. Problem Description

In the following, we consider a robotic manipulator with two cylindric-shaped links with diameter $d = 20\,\text{cm}$ and the lengths $l_0 = 18.15\,\text{cm}$, $l_1 = 16.35\,\text{cm}$, $l_2 = 57.5\,\text{cm}$, compare Figure 6.9. Further, we assume that all system parameters in the Table 6.1 are known, except for the two in brackets. The parameters to be estimated are the mass of the second link and moment of inertia in x of the second link: $p = (p_1, p_2) = (m_2, I_{x,2}) \in \mathbb{R}^2$. The vector of external forces is defined for each robotic link as

$$
h_{f,i} = k_{i,1} u_i - k_{i,2}\dot{\theta} - k_{i,3} \tanh(a_i \dot{\theta}), \qquad \text{for } i = 1, 2.
$$

It contains the control and forces arising from frictional effects, [Kna01]. The controls are the currents in each joint.

Table 6.1.: *Model parameters for the robotic manipulator at start time $t = 0$, the embraced values are the ones to be estimated.*

Parameter	**Symbol**	**Link 1**	**Link 2**	**Unit**
mass	m	6	(23)	kg
center of mass, x	s_x	0	0	m
center of mass, y	s_y	0	0	m
center of mass, z	s_z	-	0.2875	m
moment of inertia, x	I_x	-	(0.6912)	kgm^2
moment of inertia, y	I_y	-	0.6912	kgm^2
moment of inertia, z	I_z	0.03	0.115	kgm^2
factor current	k_1	0.7	0.4	-
factor vis. friction	k_2	1	2	Nm/s
factor dry friction	k_3	8	3	Nm
factor modeling	a	50	50	-

For the robotic manipulator with two links and the parameters from Table 6.1 the mass matrix and the right-hand side from (6.6) result in,

$$\begin{aligned} &M(\theta, p) = \\ &\begin{pmatrix} 0.08265625\big(1-\cos^2(\theta_2)\big)p_1 + 0.03 + p_2 + p_2\cos^2(\theta_2) & 0 \\ 0 & 0.08265625p_1 + 0.6912 \end{pmatrix}, \\ &F(\theta, \dot\theta, p) + h_f = \\ &\begin{pmatrix} -0.3125\sin(\theta_2)\cos(\theta_2)\big(0.529p_1 + 6.4p_2 - 0.736\big)\dot\theta_1\dot\theta_2 + 0.7u_1 \\ 0.3125\sin(\theta_2)\cos(\theta_2)\dot\theta_1^2\big(0.2645p_1 + 3.2p_2 - 0.368\big) + 2.8204\sin(\theta_2)p_1 + 0.4u_2 \end{pmatrix}. \end{aligned} \tag{6.7}$$

Both terms now depend on the parameters we defined above. The task is a repeated pick-and-place job, where a payload is picked up and transported to another position. This is represented by a point-to-point-control, where the end-effector of the manipulator moves from a start position, described through a start configuration of the joint angles, to a desired final position of the end-effector given by the final configuration of the joints. The start configuration and the desired point to track over time are:

$$\begin{aligned} \theta(t=0) &= \left(-\frac{1}{2}\pi, \frac{3}{4}\right)^T, \\ \theta^{ref}(t) &= \begin{cases} \left(\frac{1}{2}\pi, \frac{1}{4}\pi\right)^T, & \text{for } t \in [0,2], (8,10], (16,18], \\ \left(-\frac{1}{2}\pi, \frac{3}{4}\pi\right)^T, & \text{for } t \in (2,4], (6,8], (10,12], (14,16], \\ \left(0, \frac{1}{4}\pi\right)^T, & \text{for } t \in (4,6], (12,14], \\ \left(0, \frac{3}{4}\pi\right)^T, & \text{for } t \in (18,20]. \end{cases} \end{aligned} \tag{6.8}$$

For the simulation, we assume the following true parameters:

$$p(t) = (m_2, I_{x,2})^T = \begin{cases} (23.0, 0.6912)^T, & \text{for } t \in [0,2], (4,6], (8,10], (12,14], (16,18], \\ (33.0, 0.6912)^T, & \text{for } t \in (2,4], \\ (31.1, 0.6912)^T, & \text{for } t \in (6,8], \\ (36.0, 0.6912)^T, & \text{for } t \in (10,12], \\ (32.0, 0.6912)^T, & \text{for } t \in (14,16], \\ (30.3, 0.6912)^T, & \text{for } t \in (18,20]. \end{cases} \tag{6.9}$$

The changing mass represents some unknown payload that is transported by the robotic manipulator in the pick-and-place scenario. To simulate the system, we integrate the nonlinear system dynamics with the true parameter values with the integration scheme Runge-Kutta of fourth order with a fine discretization. For the measurement simulation, we add again a white Gaussian noise to the system outputs with $v \sim \mathcal{N}(0, \Sigma)$ and $\Sigma = \operatorname{diag}(0.1^2, ..., 0.1^2) \in \mathbb{R}^{4\times4}$.

6.4.3. The OCP for a Robotic Manipulator

For the optimal control problem within NMPC, we optimize the joint angles using the currents as controls. To formulate the concrete OCP, we must first transform the equations of motion from second-order to first-order ordinary differential equations. This is done by introducing two additional optimization variables:

$$x(t) = \big(x_1(t), x_2(t), x_3(t), x_4(t)\big)^T := \big(\theta_1(t), \theta_2(t), \dot{\theta}_1(t), \dot{\theta}_2(t)\big)^T.$$

We assume the start and final joint angles from (6.8) and that the robotic manipulator is at a rest at the start and end of each movement. Therefore, we demand the angular velocities there to be zero. This leads to:

$$\begin{aligned} x_0^{[0]} &= \left(-\frac{1}{2}\pi, \frac{3}{4}, 0, 0\right)^T, \\ x^{ref}(t) &= \begin{cases} \left(\frac{1}{2}\pi, \frac{1}{4}\pi, 0, 0\right)^T, & \text{for } t \in [0,2], (8,10], (16,18], \\ \left(-\frac{1}{2}\pi, \frac{3}{4}\pi, 0, 0\right)^T, & \text{for } t \in (2,4], (6,8], (10,12], (14,16], \\ \left(0, \frac{1}{4}\pi, 0, 0\right)^T, & \text{for } t \in (4,6], (12,14], \\ \left(0, \frac{3}{4}\pi, 0, 0\right)^T, & \text{for } t \in (18,20]. \end{cases} \end{aligned} \tag{6.10}$$

The mass matrix is positive definite by definition and thus invertible. In practice, the invertibility of the matrix is also guaranteed in the algorithm by the box constraints within the parameter identification. The dynamics are reformulated with the values from (6.7) to:

$$\dot{x}(t) = \begin{pmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \end{pmatrix} = \begin{pmatrix} x_3(t) \\ x_4(t) \\ [M^{-1}(x,p)(F(x,p) + h_f(x,u))]_1 \\ [M^{-1}(x,p)(F(x,p) + h_f(x,u))]_2 \end{pmatrix} =: f(x,u,p) \tag{6.11}$$

Here, $[\cdot]_1$ and $[\cdot]_2$ denote the first and second entries of the specified vector, respectively. Consequently, the optimal control problem to be solved in the k-th NMPC step has the following form:

$$\begin{aligned} \min_{\bar{x}^{[k]},\bar{u}^{[k]}} \quad & J^{[k]} = \int_0^T \left(\bar{x}^{[k]}(\tau;p^{[k]}) - x^{ref}(\tau)\right)^T Q(\tau) \left(\bar{x}^{[k]}(\tau;p^{[k]}) - x^{ref}(\tau)\right) \cdots \\ & \qquad \cdots + \bar{u}^{[k]}(\tau)^T R(\tau) \bar{u}^{[k]}(\tau) \,\mathrm{d}\tau \\ \text{w.r.t. } & \dot{\bar{x}}^{[k]}(\tau) = f(\bar{x}^{[k]}(\tau), \bar{u}^{[k]}(\tau), p^{[k]}), \text{ for } \tau \in [0,T], \\ & \bar{x}^{[k]}(0) = y_k, \\ & \bar{x}^{[k]}(\tau) \in [(-\pi, -\tfrac{178}{180}\pi, -\tfrac{510}{180}\pi, -\tfrac{408}{180}\pi)^T, (\pi, \tfrac{178}{180}\pi, \tfrac{510}{180}\pi, \tfrac{408}{180}\pi)^T], \\ & \bar{u}^{[k]}(\tau) \in [(-200,-200)^T, (200,200)^T], \text{ for } \tau \in [0,T]. \end{aligned} \tag{6.12}$$

Note that the parameters are fixed here to the most recent values from the parameter estimation. The reference state is changing over time as described in (6.10) with $\bar{x}^{ref,[k]}(\tau) := x^{ref}(t_k + \tau)$ for $\tau \in [0,T]$. The constraints require that the nonlinear system dynamics (6.11) are satisfied on the considered time horizon. Furthermore, the initial state is set to the current system state, which is taken from the new measurement y_k. Box constraints are used to take into account the physical limitations of joint angles in the states and the controls. Note that the values for the controls in this particular example are dimensionless and can, therefore, be comparatively large. After discretization with $N \in \mathbb{N}$ discretization points, the weights in the objective function for each discretized state $x(\tau_i)$ with $i = 0, \ldots, N-1$, are chosen to

$$Q(\tau_i) = Q_i = \frac{1}{(N-i)} \begin{pmatrix} 100 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 100 \end{pmatrix} \text{ and } R(\tau_i) = 0.01.$$

6.4.4. The NDPIP for a Robotic Manipulator

As described above, the unknown parameters are $p = (p_1, p_2)^T = (m_2, I_{x,2})^T \in \mathbb{R}^2$. The true values are denoted with $p^*(t)$ and are changing over time as defined in (6.9). The initial guess for the first NDPIP is $p_0 := p(t = 0) = (19.0, 0.5)^T$. Subsequently, the result of the previous parameter identification is used as the initial guess. This leads to the following nonlinear dynamical parameter identification problem at time t_k:

$$\begin{aligned}
\min_{x_0^{[k]},p^{[k]}} \quad & F^{[k]} := \sum_{i=0}^{M} \left(x(\xi_i; p^{[k]}, x_0^{[k]}) - y_{k-M+i} \right)^T W \left(x(\xi_i; p^{[k]}, x_0^{[k]}) - y_{k-M+i} \right) \\
\text{w.r.t.} \quad & \dot{x}(\xi) = f(x(\xi), \tilde{u}(\xi), p^{[k]}), \text{ for } \xi \in [t_k - T_I, t_k], \\
& x(\xi_0) = x_0^{[k]}, \\
& p^{[k]} \in [(10, -1)^T, (40, 1)^T],
\end{aligned} \tag{6.13}$$

where the differential equations in the constraints from (6.11) represent the equations of motion of the robotic manipulator. The controls are an approximation from the measured controls at the discrete time points like described for problem (6.1). The box constraints on the parameters ensure that they remain in physically feasible ranges.

6.4.5. Numerical Results

The described system is simulated for $T_s = 20\,\text{s}$ with a sample time $\Delta t = 0.005\,\text{s}$. The prediction horizon of each OCP is $T = 0.6\,\text{s}$. We choose a multistep-NMPC approach and solve a new OCP every 0.1 seconds. Thus, the control horizon is $T_C = 0.1\,\text{s}$. The identification horizon of the parameter identification problems is chosen to $T_I = 0.2\,\text{s}$, so that there are 41 measurements available for each NDPIP. The discretization is done with $N = 21$ discretization points. This keeps the computation times small enough to perform online. The mean computation time per PI problem is

$$\varnothing t_{\text{PI},\texttt{NMPC+PI}} = 93.23\,\text{ms}.$$

Figure 6.10 shows the states of the manipulator angles and angular velocities for each link, which is performing a repeated pick-and-place scenario described by (6.10). The combined approach `NMPC+PI` is depicted in violet. For comparison, the trajectories for the classical `NMPC` and `optimal NMPC` are displayed in blue and yellow, respectively. The dotted red line shows the desired reference value for each state, which is tracked within the objective function of the OCPs. It can be observed that all control approaches are able to stabilize the system to the desired states. However, standard `NMPC` is less precise in reaching the desired states, especially in the angle of the second link. The average computation time for a solution of an optimal control problem for the classical `NMPC` is,

$$\varnothing t_{\text{OCP},\texttt{NMPC}} = 1505.75\,\text{ms},$$

in this example. The average computation time for an OCP for `NMPC+PI` is,

$$\varnothing t_{\text{OCP},\texttt{NMPC+PI}} = 291.43\,\text{ms},$$

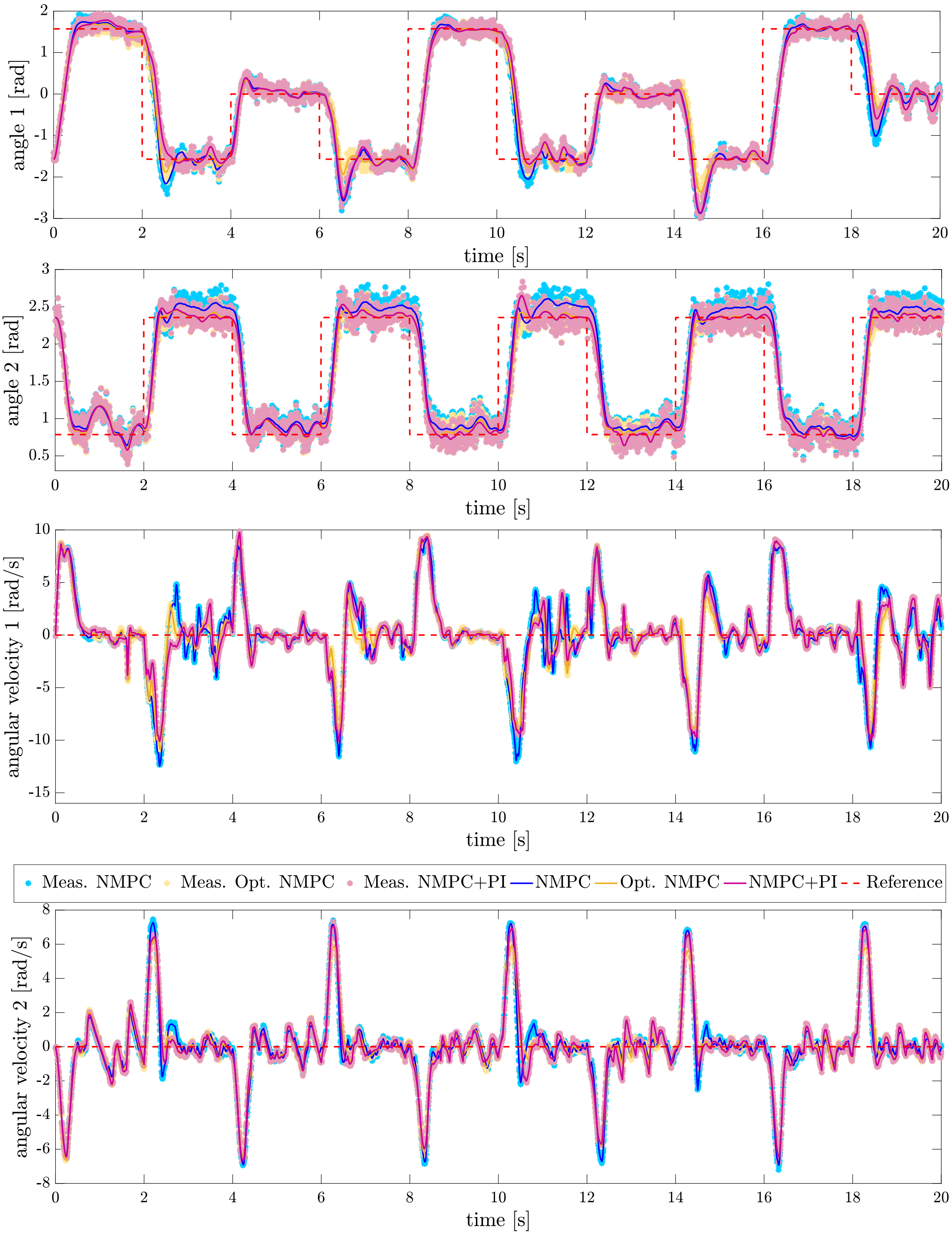

Figure 6.10.: *States of the two-link robot for a pick-and-place scenario controlled by classic `NMPC` (blue line), `NMPC+PI` (violet line), `optimal NMPC` (yellow line), the corresponding noisy measurements (dots in similar color) and the desired reference values (dotted red line); angle link 1, angle link 2, and angular velocities link 1 and 2.*

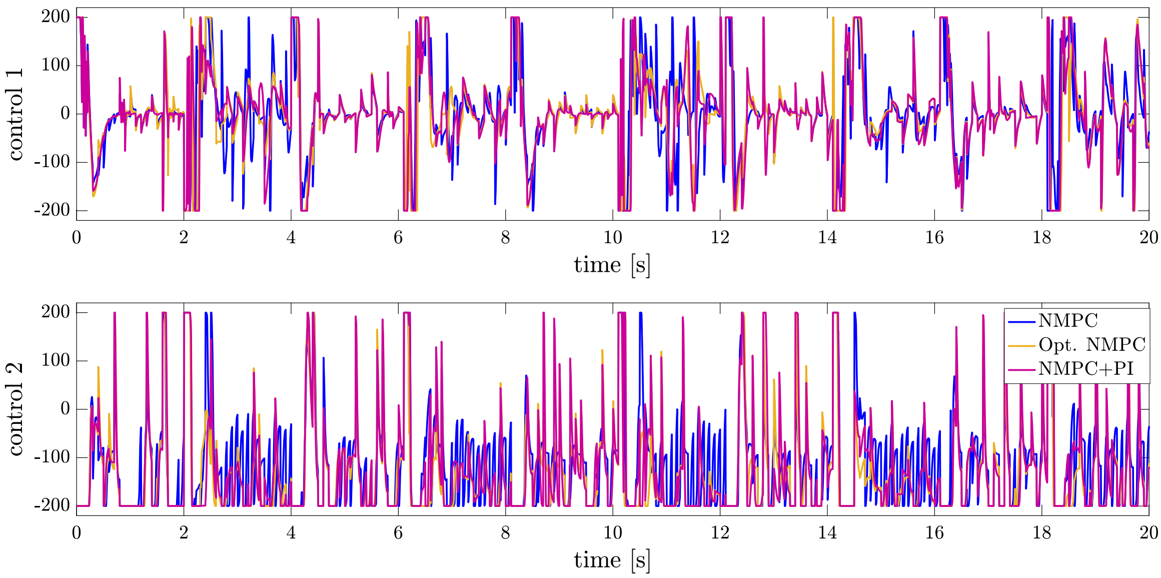

Figure 6.11.: *The trajectories of the two controls for standard* `NMPC` *(blue line),* `NMPC+PI` *(violet line) and* `optimal NMPC` *(yellow line) for the two-link robot.*

and for the `optimal NMPC` it is,

$$\varnothing t_{\mathrm{OCP},\texttt{op.NMPC}} = 254.48\,\mathrm{ms}.$$

This indicates that providing a better model in the optimal control problem leads to shorter computation times. That is particularly interesting since computation time is critical, especially in model predictive control. Even the approximately 250 milliseconds in the optimal NMPC are still too large for a practical application. For the results presented, already very good initial estimates of all optimization variables from the predecessor problem and the Lagrange multipliers from the former problem are used. However, there are many suggestions on how to shorten the computation times further. For example, a smaller number of discretization points could be used. Thus, there would be fewer optimization variables, and the computational cost would decrease immediately. It is also possible to weaken the tolerances for the optimality of the solution within the optimizer WORHP, then an optimal solution is accepted earlier. Furthermore, it is another option for acceleration to use the *hot start* of the optimizer, which enables a particularly efficient start of the optimization from the previous problem. In Figure 6.11, the corresponding control trajectories are plotted. Note that the controls are dimensionless. Comparing the whole control effort of standard `NMPC` and `NMPC+PI` with `optimal NMPC` as benchmark, we get:

$$\frac{\|u^{\texttt{NMPC}}\|_2}{\|u^{\texttt{op.NMPC}}\|_2} = 0.92855 \quad \text{and} \quad \frac{\|u^{\texttt{NMPC+PI}}\|_2}{\|u^{\texttt{op.NMPC}}\|_2} = 0.99866.$$

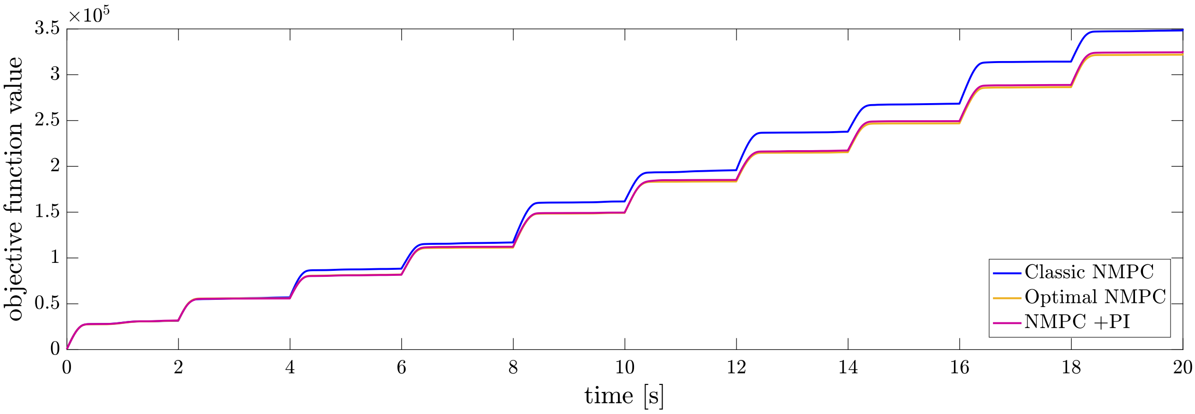

Figure 6.12.: *A posteriori computed accumulated objective function value over the complete simulation horizon for the two-link robot; standard* `NMPC` *(blue),* `NMPC+PI` *(violet) and* `optimal NMPC` *(yellow).*

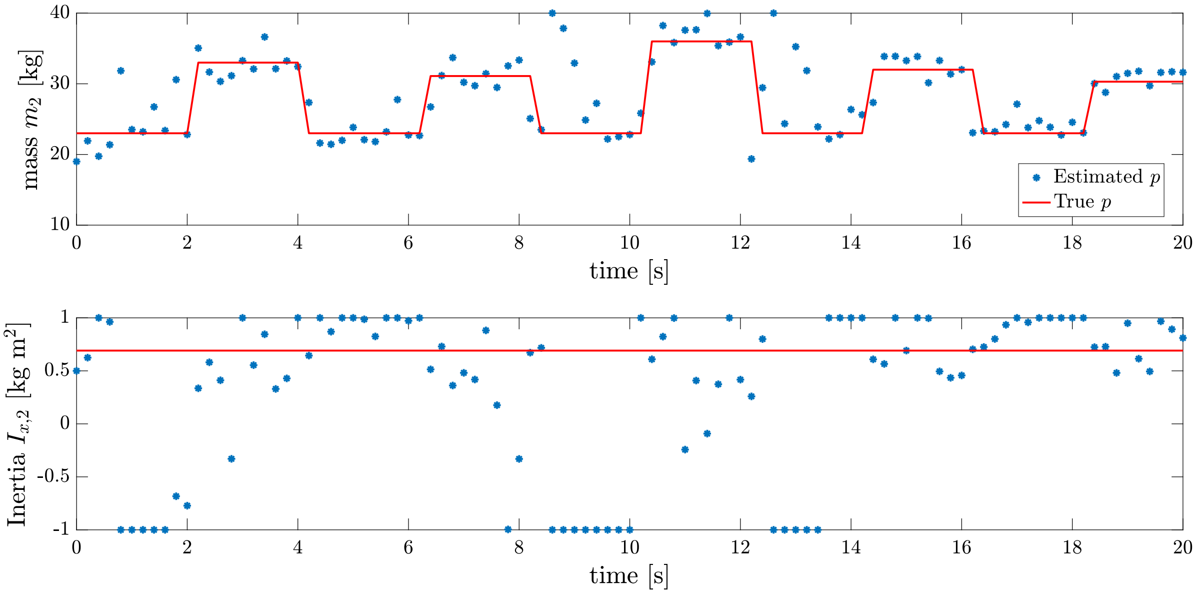

Figure 6.13.: *Identified parameters (blue dots) and reference values (red line) for the two-link robot controlled with* `NMPC+PI`*.*

This means that both standard `NMPC` and `NMPC+PI` need less control effort than the `optimal NMPC`. However, it is important to remember that not only is the control effort an optimization criterion within the objective function of the OCPs, but also the tracking of the desired states is another one. Therefore, we evaluate an overall objective function of the result in a posteriori analysis for comparison. To do this, we accumulate the objective function from problem (6.12) over time. This contains the weighted squared tracking error and the control effort. In Figure 6.12, it can be seen that our proposed algorithm gives a better result than the standard NMPC and is almost as good as the optimal NMPC, which always has the exact parameters available.

Figure 6.13 shows the online identified parameters as blue dots for each performed parameter identification. The true values from (6.9) are shown as references in red. It can be observed that the changing mass is estimated well. The moment of inertia is estimated properly only when there is much motion in the system. Otherwise, the constraints on this second parameter often become active. Overall, however, the parameter identifications and the associated model adaptations lead to a better tracking behavior of the controller. The root mean square error in the parameters is:

$$\mathrm{RMSE}_p := \sqrt{\frac{\sum_{i=1}^{M}(p_i - p_i^{ref})^2}{M}} = \begin{pmatrix} 4.4780 \\ 0.8130 \end{pmatrix}.$$

Each identification problem is solved with the full discretization approach using $N = 21$ discretization points. The mean computation time of about 93 ms represents considerable additional computational effort in the NMPC context. Furthermore, other improvements can be made for faster computation, such as parallelizing the solution of the problems. However, we have already shown that the additional parameter identifications lead to savings in computational time for the OCPs, so there is still an absolute advantage. Overall, the parameter identifications and the associated model adaptations lead to a better tracking behavior of the controller.

7. Summary and Outlook

7.1. Summary

Classically, feedback control is used to compensate for model inaccuracies or to react to disturbances from the environment on the system under consideration. However, the incoming information about the system behavior is rarely used to adapt the underlying model of the controller. This work intends to propose new approaches that improve known feedback control algorithms by online nonlinear parameter identification.

This goal is accomplished in two different ways. First, an adaptive linear quadratic regulator with sensitivity-based updates is considered. For the adaptations of the feedback gain based on parametric sensitivities, the changes in the parameters are needed. In order to determine the deviations in the model parameters in real-time, this control concept is combined with an online parameter identification. This means that the system model in use is calibrated with the help of measurement data collected online. The formulation of the parameter estimation problem as a nonlinear optimization problem with constraints allows the formulation of conditions that ensure the applicability of the methods from parametric sensitivity analysis. More precisely, they were formulated in terms of feasibility sets for the parameters in the constraints of the NLP.

A common problem that can arise is insufficiently informative data to determine effective parameters. This problem is addressed by extending the presented method by estimating the Fisher information matrix. This provides an estimate of the collected data's information content and can help decide whether a PI should be done. The developed algorithms are investigated using the example of an inverted pendulum on a cart in different scenarios. It is shown how the online adaptation of the feedback gain can lead to improved control performance with less control effort. Furthermore, in contrast to the classical linear quadratic controller, it is possible to stabilize the system even in case of larger parametric disturbances. It is also demonstrated on time-varying parameters, discontinuous reference trajectories, and longer time intervals with little system motion how the additional computation of the FIM can avoid performing parameter identifications on insufficiently informative data. Compared to classical LQR and an optimal adaptive LQR that updates the feedback gain online with the true parameters, our methods outperformed the classical LQR. They were nearly as good as the optimal one.

The second approach of this work is to extend a nonlinear model predictive controller with model adjustments resulting from online parameter identifications. In particular, it is presented in this framework that the problems encountered, both the optimal control problems in the model-predictive controller and the parameter estimation problems, can be solved using the same direct numerical approaches originating from optimal control theory. The numerical experiments of the presented algorithm with two examples show that the model adaptations can improve the controller performance. The numerical experiments of the presented algorithm on the pendulum-cart system demonstrate that the model adaptations can reduce the control effort compared to the classical `NMPC`. In particular, the quality of state tracking is comparable to that of `optimal NMPC` that knows the true parameter values. The results using a robotic manipulator as an example show an improvement in the objective function value of the OCPs compared to the standard `NMPC`, as well as a reduction in the computation time required for solving the optimal control problems.

7.2. Future Work

In this dissertation, different methods of optimal feedback control and parameter identification are brought together, where the major challenge is the combination, adaptation, and tuning of the different methods. All formulations are always kept as general as possible to obtain an approach independent of the application problem. The contributions of this thesis can be extended as follows:

- Within the parameter identification problems, an objective function formulation with the L_2 norm was used. However, the solution methods are also capable of using other formulations such as the L_1-norm or the L_∞-norm for data fitting.
- Another possibility is to transfer the proposed concepts to other problem classes than those where the dynamical system is defined by ordinary differential equations, for example, problems with partial or algebraic differential equations. A study on problems with constraints involving partial differential equations would be fascinating since these are common in applications, but there are still some open research questions.

In addition, it might be of value to analyze how parameter identification, in the form used here in combination with the adaptive linear quadratic controller, can also serve as an indicator for evaluating the performance of `adLQR`. It is of particular interest to identify when updates of the gain with sensitivity differences are no longer sufficient to achieve stability. In such a case, a recomputation of the feedback gain should be initiated. In this context, the sensitivity-based approximation of the feedback gain can be used as a reasonable initial estimate for the calculations.

In the `adLQR+PI` algorithm, we formulated conditions on the parameter identification process that ensure the further feasible use of the parametric sensitivity analysis methods. It would be highly relevant to work out equivalent statements for the approach presented in Chapter 6. These could similarly be formulated from imposed conditions on the parameter changes by the PI. In this case, it is essential to guarantee that the system, which is asymptotically stable as per Theorem 4.4.4, retains this property despite any modifications in the model parameters.

Another promising aspect is the consideration of the identifiability of the parameters within the `NMPC+PI` control algorithm. The computation of the FIM has been omitted so far due to computational costs in the NMPC setting. Solving optimal control problems already involves significant computational costs, which are increased in our approach by solving identification problems. Perhaps a less accurate approximation of the FIM can be considered as a test of the information content in the data set. Another possible access to this topic is *adaptive dual NMPC*. This approach extends ideas from classical *dual control* for linear systems, [Fel60b; Fel60a; Åst70]. For that, the objective functions of the optimal control problems are modified to get more information from the measurements for the identification. An additional term is added that gives a statement about the information potential of the control trajectory for future parameter estimation and is co-optimized. The Fisher information matrix is often used in this context. This aims to improve the quality of future parameter estimates by allowing small changes in the trajectories to lead to more information content about the influence of the parameters on the system. Transferring these ideas to the approaches used in this work would be most exciting. To this end, the objective function of the optimal control problem in model predictive control, similarly defined as in (4.8), could be adjusted by an additional term:

$$\min_{x,u} \quad \int_0^T (x - x^{ref})^T Q(x - x^{ref}) + (u - u^{ref})^T R(u - u^{ref}) \, \mathrm{d}t + \omega \ \mathrm{tr}\left(\mathcal{I}_p^{-1}(u)\right),$$

where $\omega \in \mathbb{R}$ is a weighting constant. In the context of dual control, the former objective, here the integral term, is called the *performance control term* and the last term the *information gain term*. The new information term can be chosen in different ways. We have formulated the above cost function in terms of the *A-optimality criterion* on the Fisher information matrix, which we already know from Subsection 5.3.2. This approach is also proposed in Feng and Houska [FH18] with a version where each state can be weighted. Wilson et al. [WSM14; WSM15] use the *E-optimality criterion* in a sequential action control approach to get informative trajectories for PI. This criterion deals with the minimization of the inverse of the smallest eigenvalue of the FIM. Implementing these approaches for real-time methods is very challenging since covariance matrices or their derivations with respect to controls are necessary, which can be numerically costly for nonlinear systems.

The additional term in the cost function leads to a multiobjective optimization problem. This would then require further investigation of how to determine the weight ω for an optimal weighting of the different terms, performance control term, and information term. Current research explores this topic, especially this weighting, [La+16; La16; FH18; Hou+17].

The simulations developed for this thesis aim to accurately represent the system behavior and the effects of various control algorithms. Neglecting the influences of computation and transmission times is unavoidable in some areas, particularly in the NMPC approach. However, this has a limited role in the presented results as it primarily relates to the comparison with other NMPC approaches. For future practical applications of the proposed approaches, it would be beneficial to incorporate computation delays into the simulation as an initial endurance test.

Furthermore, the numerical simulations performed within the scope of this work show promising results. As previously mentioned, the next significant step is the transfer of the presented algorithms into a real-world system. A seven-degree-of-freedom robotic manipulator (DENSO VS050) has been studied in the context of this work to investigate various numerical solution approaches for the parameter identification problems using real-world data, Schäfer and Runge et al. [Sch+18]. However, to use the controllers presented here, the system must be equipped with additional sensor technology to measure the true angles of the individual robot links. Another suitable application is autonomously driving rovers or cars, for which the dynamical system models are often challenging to parameterize.

Appendix A.

Fisher Information Matrix

A.1. Computation of the Fisher Information Matrix for the Inverted Pendulum on a Cart

To compute the Fisher information matrix for our example of the pendulum-cart system, we again consider the nonlinear system model of the inverted pendulum on a cart given by (5.13). By using the state $x := (y, \dot{y}, \theta, \dot{\theta})^T \in \mathbb{R}^4$ the equations in (5.13) can be written in state space form as

$$\dot{x} = \begin{pmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \\ \dot{x_4} \end{pmatrix} = \begin{pmatrix} x_2 \\ \frac{1}{M+ms_{x_3}^2}(mls_{x_3}x_4^2 - mgc_{x_3}s_{x_3} + mu) \\ x_4 \\ \frac{1}{M+ms_{x_3}^2}(\frac{(M+m)gs_{x_3}}{l} - mc_{x_3}s_{x_3}x_4^2 - \frac{1}{l}c_{x_3}u) \end{pmatrix} = f(x(t), u, p).$$

We use the abbreviations $s_{x_3} := \sin x_3$ and $c_{x_3} := \cos x_3$. Further in our case the control u is fixed by the feedback law $u = -Kx = \begin{pmatrix} K_1 & K_2 & K_3 & K_4 \end{pmatrix} x$. The original parameter vector for parameter identification $\bar{p} := (M, l)^T \in \mathbb{R}^2$ is extended by the parameters for the initial states to $p := (M, l, x_{0,1}, x_{0,2}, x_{0,3}, x_{0,4})^T \in \mathbb{R}^7$.

To approximate the Fisher information matrix from (5.10), we have to compute the sensitivities

$$\frac{d}{dp} f(x(t_i; p), \tilde{u}(t_i), p).$$

It is

$$\frac{d}{dp} f(x(t; p), \tilde{u}(t), p) = \nabla_x f(x(t_i; p), \tilde{u}(t_i), p) \nabla_p x(t, p) + \nabla_p f(x(t_i; p), \tilde{u}(t_i), p),$$

with the following partial derivatives:

$$\nabla_x f = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -\frac{K_1 m}{M+ms_{x_3}^2} & -\frac{K_2 m}{M+ms_{x_3}^2} & \frac{lmx_4^2 c_{x_3}}{M+ms_{x_3}^2} - \frac{2lm^2 x_4^2 c_{x_3} s_{x_3}^2}{(M+ms_{x_3}^2)^2} + \frac{gms_{x_3}^2 x_3 - gmc_{x_3}^2 x_3 - K_3 m}{M+ms_{x_3}^2} & \frac{2lms_{x_3} x_4}{M+ms_{x_3}^2} - \frac{K_4 m}{M+ms_{x_3}^2} \\ 0 & 0 & 0 & 1 \\ \frac{K_1 c_{x_3}}{l(M+ms_{x_3}^2)} & \frac{K_2 c_{x_3}}{l(M+ms_{x_3}^2)} & \frac{mx_4^2 s_{x_3}^2 + \frac{us_{x_3} + g(M+m)c_{x_3} + K_3 c_{x_3}}{l} - mx_4^2 c_{x_3}^2}{M+ms_{x_3}^2} - \frac{2mc_{x_3} s_{x_3}(-mx_4^2 c_{x_3} s_{x_3} + \frac{g(M+m)s_{x_3} - uc_{x_3}}{l})}{(M+ms_{x_3}^2)^2} & \frac{\frac{K_4 c_{x_3}}{l} - 2mc_{x_3} s_{x_3} x_4}{M+ms_{x_3}^2} \end{pmatrix}$$

and

$$\nabla_p f = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{lms_{x_3} x_4^2 + mu - gmc_{x_3} s_{x_3}}{(M+ms_{x_3}^2)^2} & \frac{ms_{x_3} x_4^2}{M+ms_{x_3}^2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{gs_{x_3}}{M+ms_{x_3}^2} - \frac{\frac{g(M+m)s_{x_3} - c_{x_3} u}{l} - mc_{x_3} s_{x_3} x_4^2}{(M+ms_{x_3}^2)^2} & \frac{\frac{c_{x_3} u - g(M+m)s_{x_3}}{l^2}}{M+ms_{x_3}^2} & 0 & 0 & 0 & 0 \end{pmatrix}.$$

With the help of these forward sensitivities, we can estimate the Fisher information matrix for the planned control values as soon as they have been determined. We use this for the numerical analysis in Subsection 5.4.6.

Appendix B.

Additional Numerical Results for the Inverted Pendulum on a Cart

B.1. `adLQR+PI+FIM` for the Inverted Pendulum on a Cart for Scenario 0-B

Figure B.1 and B.2 show the states and controls for scenario 0-B with different controllers. The purple plot depicts the results using the proposed Algorithm 5.2 with additional FIM computations. These are additional visualizations for the numerical analysis in Subsection 5.4.6.

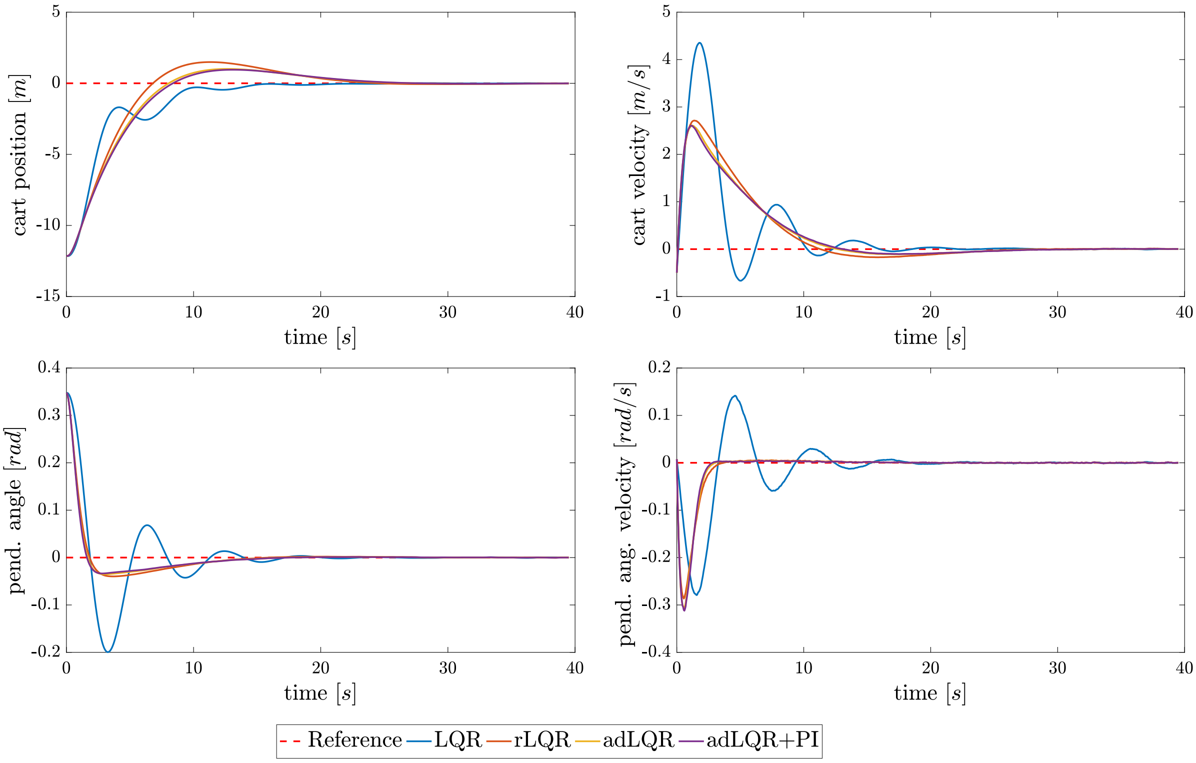

Figure B.1.: *States of the inverted pendulum on a cart for scenario 0-B with additional FIM-check; cart position, cart velocity, pendulum deflection angle, and pendulum angular velocity for a simulation horizon of* $T = 40\,\mathrm{s}$.

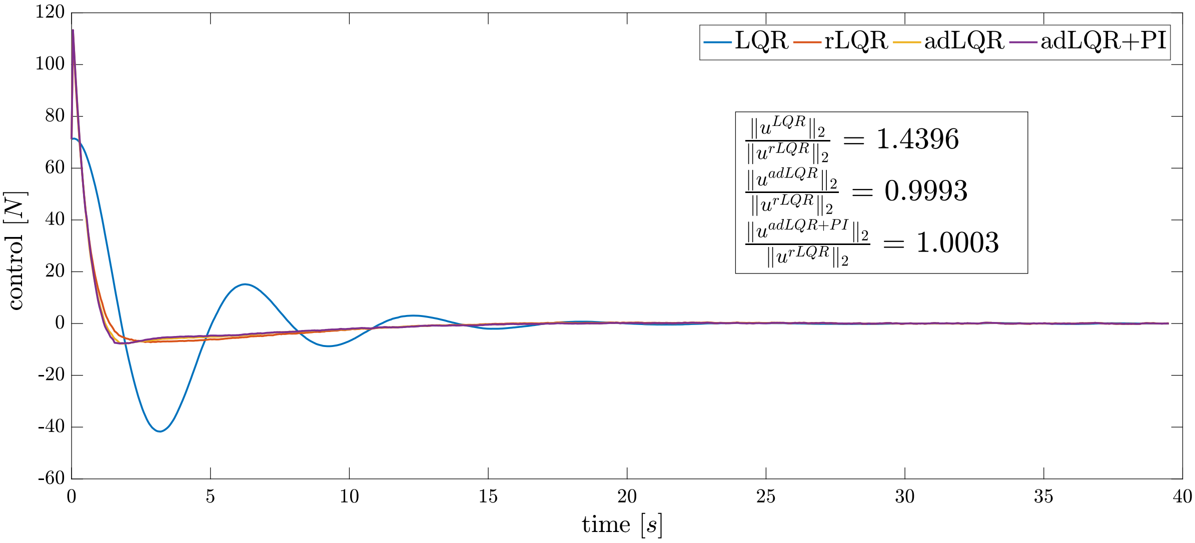

Figure B.2.: *Control of the inverted pendulum on a cart for scenario 0-B with additional FIM-check for a simulation horizon of* $T = 40\,\mathrm{s}$.

Table B.1 gives the accumulated objective function values for the different controllers for all possible tracking scenario combinations and parametric perturbations. From these values, we computed the relative objective function values in Table 5.1 in Subsection 5.4.6.

Table B.1.: *Objective function of the LQR problem for all scenarios for the inverted pendulum-cart system.*

Scenario identifier		**Objective** F_{LQR}				
x_{ref}	Δp	LQR	rLQR	adLQR	adLQR+PI	adLQR+PI+FIM
0	A	5.239×10^3	5.031×10^3	5.075×10^3	5.078×10^3	5.076×10^3
0	B	3.987×10^4	1.949×10^4	1.950×10^4	1.956×10^4	1.956×10^4
0	C	4.941×10^3	4.898×10^3	4.901×10^3	4.901×10^3	4.899×10^3
0	D	4.702×10^3	4.395×10^3	4.404×10^3	4.415×10^3	4.411×10^3
0	E	5.845×10^3	5.623×10^3	5.619×10^3	5.651×10^3	5.648×10^3
0	F	8.563×10^3	6.639×10^3	6.675×10^3	6.735×10^3	6.733×10^3
0	G	1.064×10^4	8.379×10^3	8.404×10^3	8.432×10^3	8.431×10^3
0	H	5.570×10^3	5.395×10^3	5.414×10^3	5.385×10^3	5.387×10^3
0	I	5.065×10^3	3.910×10^3	3.936×10^3	3.924×10^3	3.922×10^3
1	A	4.865×10^4	4.656×10^4	4.661×10^4	4.733×10^4	4.697×10^4
1	B	1.471×10^5	6.376×10^4	6.379×10^4	6.402×10^4	6.368×10^4

Continued on next page

x_{ref}	Δp	LQR	rLQR	adLQR	adLQR+PI	adLQR+PI+FIM
1	C	4.156×10^4	4.040×10^4	4.041×10^4	4.184×10^4	4.093×10^4
1	D	4.459×10^4	4.397×10^4	4.397×10^4	4.522×10^4	4.559×10^4
1	E	4.447×10^4	4.275×10^4	4.273×10^4	4.354×10^4	4.279×10^4
1	F	5.012×10^4	4.256×10^4	4.275×10^4	4.338×10^4	4.333×10^4
1	G	8.075×10^4	5.150×10^4	5.158×10^4	5.192×10^4	5.167×10^4
1	H	6.998×10^4	5.130×10^4	5.141×10^4	5.180×10^4	5.148×10^4
1	I	4.888×10^4	4.364×10^4	4.366×10^4	4.477×10^4	4.737×10^4
2	A	6.584×10^4	6.320×10^4	6.321×10^4	6.364×10^4	6.334×10^4
2	B	2.557×10^5	9.942×10^4	9.984×10^4	1.000×10^5	1.001×10^5
2	C	5.167×10^4	4.974×10^4	4.974×10^4	5.207×10^4	5.091×10^4
2	D	5.881×10^4	5.848×10^4	5.857×10^4	6.001×10^4	5.940×10^4
2	E	5.410×10^4	5.163×10^4	5.171×10^4	5.243×10^4	5.180×10^4
2	F	7.232×10^4	6.318×10^4	6.342×10^4	6.387×10^4	6.461×10^4
2	G	1.233×10^5	7.791×10^4	7.798×10^4	7.862×10^4	7.810×10^4
2	H	1.031×10^5	6.970×10^4	6.981×10^4	7.024×10^4	6.981×10^4
2	I	6.487×10^4	5.796×10^4	5.816×10^4	6.005×10^4	5.930×10^4
3	A	4.327×10^4	4.066×10^4	4.061×10^4	4.096×10^4	4.068×10^4
3	B	1.213×10^5	5.114×10^4	5.129×10^4	5.151×10^4	5.143×10^4
3	C	3.782×10^4	3.7213×10^4	3.718×10^4	3.782×10^4	3.741×10^4
3	D	3.993×10^4	3.870×10^4	3.872×10^4	3.921×10^4	3.888×10^4
3	E	4.055×10^4	3.976×10^4	3.974×10^4	4.016×10^4	3.975×10^4
3	F	4.832×10^4	3.821×10^4	3.830×10^4	3.853×10^4	3.827×10^4
3	G	5.281×10^4	4.043×10^4	4.045×10^4	4.060×10^4	4.039×10^4
3	H	5.084×10^4	4.340×10^4	4.345×10^4	4.374×10^4	4.363×10^4
3	I	4.055×10^4	3.518×10^4	3.517×10^4	3.591×10^4	3.562×10^4
4	A	9.506×10^4	8.862×10^4	8.868×10^4	8.932×10^4	8.865×10^4
4	B	2.975×10^5	1.073×10^5	1.075×10^5	1.084×10^5	1.074×10^5
4	C	7.867×10^4	7.719×10^4	7.723×10^4	7.804×10^4	7.824×10^4
4	D	8.651×10^4	8.406×10^4	8.406×10^4	8.448×10^4	8.430×10^4
4	E	8.312×10^4	8.124×10^4	8.123×10^4	8.181×10^4	8.143×10^4
4	F	9.838×10^4	8.0528×10^4	8.080×10^4	8.117×10^4	8.100×10^4
4	G	1.433×10^5	9.346×10^4	9.334×10^4	9.524×10^4	9.628×10^4
4	H	1.344×10^5	9.666×10^4	9.669×10^4	9.783×10^4	9.684×10^4
4	I	9.218×10^4	8.102×10^4	8.104×10^4	8.207×10^4	$8.186 + 04$
5	A	5.192×10^5	4.688×10^5	4.684×10^5	4.696×10^5	4.693×10^5
5	B	∞	5.437×10^5	5.360×10^5	5.367×10^5	5.370×10^5

Continued on next page

x_{ref}	Δp	LQR	rLQR	adLQR	adLQR+PI	adLQR+PI+FIM
5	C	4.064×10^5	3.972×10^5	3.964×10^5	4.055×10^5	4.052×10^5
5	D	4.514×10^5	4.457×10^5	4.450×10^5	4.470×10^5	4.466×10^5
5	E	4.263×10^5	4.104×10^5	4.096×10^5	4.110×10^5	4.106×10^5
5	F	6.785×10^5	4.5020×10^5	4.489×10^5	4.496×10^5	4.504×10^5
5	G	∞	4.974×10^5	4.941×10^5	4.947×10^5	4.955×10^5
5	H	8.255×10^5	5.065×10^5	5.048×10^5	5.060×10^5	5.065×10^5
5	I	5.195×10^5	4.444×10^5	4.436×10^5	4.453×10^5	4.450×10^5

B.2. NMPC+PI for the Inverted Pendulum on a Cart Scenario 0-B

Figure B.3 shows the states of the pendulum-cart for scenario 0-B. It is presented as a comparison to Scenario 0-A in Subsection 6.3.3 to show that with larger parametric perturbations the classical NMPC can no longer achieve stability, but NMPC+PI stabilizes the system.

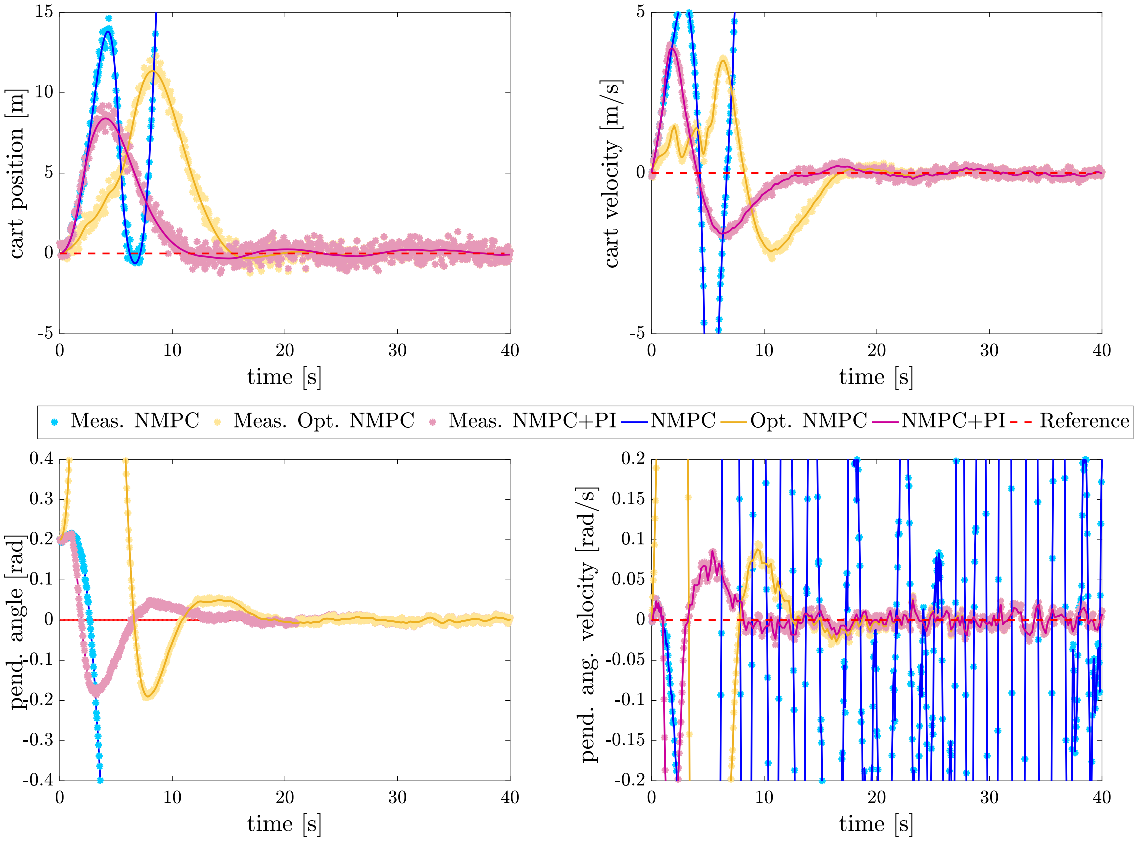

Figure B.3.: *States of the inverted pendulum on a cart for scenario 0-B controlled by classic NMPC (blue), NMPC+PI (violet), optimal NMPC (yellow) and the corresponding noisy measurements (dots in similar color); cart position, cart velocity, pendulum deflection angle, and pendulum angular velocity for a simulation horizon of $T = 40s$.*

B.3. NMPC+PI for the Inverted Pendulum on a Cart Scenario 2-G

In addition to the plots in Subsection 6.3.4, the following Figure B.4 shows the states for the classical NMPC for scenario 2-G. The controller is not capable to stabilize the system.

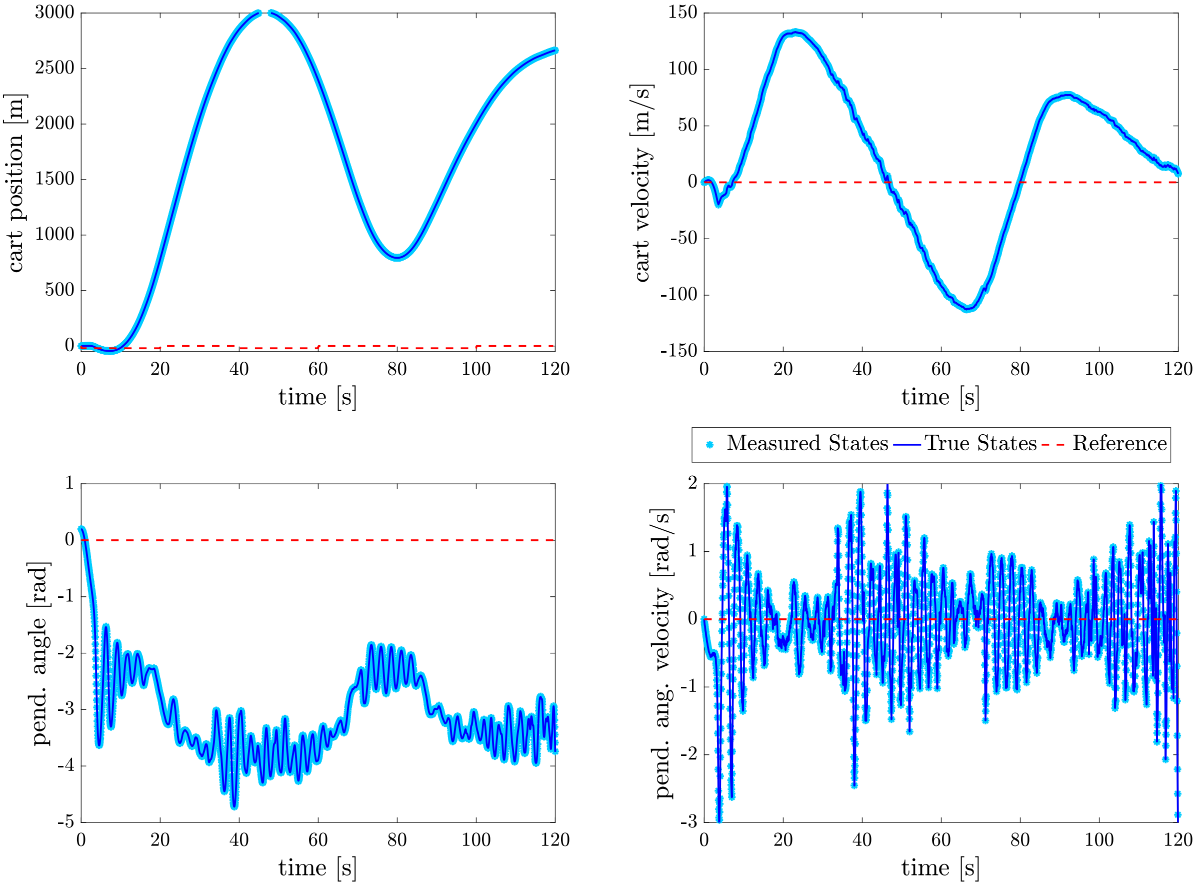

Figure B.4.: *States of the inverted pendulum on a cart for scenario 2-G controlled by classical NMPC and the corresponding noisy measurements (dots); cart position, cart velocity, pendulum deflection angle, and pendulum angular velocity for a simulation horizon of* $T = 120\,\mathrm{s}$.

Bibliography

[AA00] P. Apkarian and R. J. Adams. “Advanced gain-scheduling techniques for uncertain systems”. In: *Advances in linear matrix inequality methods in control.* SIAM, 2000, pp. 209–228.

[ADG09] V. Adetola, D. Dehaan, and M. Guay. “Adaptive model predictive control for constrained nonlinear systems”. In: *Systems & Control Letters* 58 (May 2009), pp. 320–326.

[ADT07] A. Atkinson, A. Donev, and R. Tobias. *Optimum Experimental Designs, With SAS.* Oxford Statistical Science Series Vol. 34. OUP Oxford, 2007.

[AG07] V. Adetola and M. Guay. “Parameter convergence in adaptive extremum-seeking control”. In: *Automatica* 43.1 (2007), pp. 105–110.

[AM12] B. D. Anderson and J. B. Moore. *Optimal filtering.* Courier Corporation, 2012.

[APD71] B. D. Anderson, J. B. Pren, and S. L. Dickerson. *Linear optimal control.* 1971.

[Arm66] L. Armijo. “Minimization of functions having Lipschitz continuous first partial derivatives”. In: *Pacific Journal of mathematics* 16.1 (1966), pp. 1–3.

[Åst70] K. J. Åström. *Introduction to stochastic control theory.* Mathematics in science and engineering, v. 70. xi, 299. Academic Press, 1970.

[Asw+13] A. Aswani, H. González, S. S. Sastry, and C. J. Tomlin. “Provably safe and robust learning-based model predictive control”. In: *Automatica* 49 (2013), pp. 1216–1226.

[ÅW09] K. J. Åström and B. Wittenmark. *Adaptive control.* Vol. 2. Pearson Education, 2009.

[ÅW73] K. J. Åström and B. Wittenmark. “On self tuning regulators”. In: *Automatica* 9.2 (1973), pp. 185–199.

[BÅ70] R. Bellman and K. J. Åström. “On structural identifiability”. In: *Mathematical Biosciences* 7.3 (1970), pp. 329–339.

[BC12] A. Badinski and D. Corbett. “Model-Based Design of Experiments for Estimating Heat-Transport Parameters in Tubular Reactors”. In: *Model Based Parameter Estimation: Theory and Applications.* Springer, 2012, pp. 251–266.

[Ben09] P. Benner. *Mathematische System- und Regelungstheorie.* lecture notes, Technische Universität Chemnitz. 2009.

[Ber+22] J. Berberich, J. Köhler, M. A. Müller, and F. Allgöwer. “Stability in data-driven MPC: an inherent robustness perspective”. In: *61st Conference on Decision and Control (CDC).* IEEE. 2022, pp. 1105–1110.

[Bil+22] D. Bilgic, A. Koch, G. Pan, and T. Faulwasser. “Toward data-driven predictive control of multi-energy distribution systems”. In: *Electric Power Systems Research* 212 (2022), p. 108311.

[BLK01] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan. *Estimation with applications to tracking and navigation: theory algorithms and software.* John Wiley & Sons, Inc., 2001.

[BLM09] C. L. Bottasso, F. Luraghi, and G. Maisano. “A unified approach to trajectory optimization and parameter estimation in vehicle dynamics”. In: *Proceedings of the CMND 2009, International Symposium on Coupled Methods in Numerical Dynamics, Split, Croatia.* 2009.

[Boc+13] H. G. Bock, T. Carraro, W. Jäger, S. Körkel, R. Rannacher, and J. P. Schlöder. *Model Based Parameter Estimation: Theory and Applications.* Vol. 4. Springer Science & Business Media, 2013.

[Bro70] C. G. Broyden. “The convergence of a class of double-rank minimization algorithms 1. general considerations”. In: *IMA Journal of Applied Mathematics* 6.1 (1970), pp. 76–90.

[Brü+21] T. Brüdigam, V. Gaßmann, D. Wollherr, and M. Leibold. “Minimization of constraint violation probability in model predictive control”. In: *International Journal of Robust and Nonlinear Control* 31.14 (2021), pp. 6740–6772.

[BS05] R. Bulirsch and J. Stoer. *Numerische Mathematik 2.* Vol. 7. Springer-Verlag Berlin Heidelberg, 2005.

[Bun89] A. Bunse-Gerstner. “Numerical methods for algebraic Riccati equations”. In: *Proc. Workshop on the Riccati Equation in Control, Systems, and Signals.* Pitagora Editrice, Bologna. 1989, pp. 107–116.

[Büs02] C. Büskens. *Echtzeitoptimierung und Echtzeitoptimalsteuerung parametergestörter Probleme.* Habilitation. Bayreuth, 2002.

[Büs09] C. Büskens. “Echtzeitanpassung des klassischen Riccati-Reglers, Real-Time Adaption of the Classical Riccati-Controller”. In: *at-Automatisierungstechnik Methoden und Anwendungen der Steuerungs-, Regelungs- und Informationstechnik* 57.6 (2009), pp. 269–278.

[Büs19] C. Büskens. *Optimale Regelung.* lecture notes, University of Bremen. 2018/2019.

[Büs98] C. Büskens. “Optimierungsmethoden und Sensitivitätsanalyse für optimale Steuerprozesse mit Steuer- und Zustands-Beschränkungen”. PhD thesis. 1998.

[BW13] C. Büskens and D. Wassel. “The ESA NLP Solver WORHP”. In: *Modeling and Optimization in Space Engineering*. Ed. by G. Fasano and J. D. Pintér. New York, NY: Springer New York, 2013, pp. 85–110.

[BZB18] M. Bujarbaruah, X. Zhang, and F. Borrelli. “Adaptive MPC with Chance Constraints for FIR Systems”. In: *2018 Annual American Control Conference (ACC)* (2018), pp. 2312–2317.

[CA98] H. Chen and F. Allgöwer. “A Quasi-Infinite Horizon Nonlinear Model Predictive Control Scheme with Guaranteed Stability.” In: *Automatica* 34.10 (1998), pp. 1205–1217.

[Cat89] D. E. Catlin. *Estimation, control, and the discrete Kalman filter*. Applied mathematical sciences; 71. New York: Springer, 1989.

[CB07] E. F. Camacho and C. Bordons. “Nonlinear model predictive control”. In: *Model predictive control*. Second Edition. Springer, London, 2007. Chap. Nonlinear Model Predictive Control. Pp. 249–288.

[CN19] A. de Cheveigné and I. Nelken. “Filters: When, Why, and How (Not) to Use Them”. In: *Neuron* 102.2 (2019), pp. 280–293.

[Die+05] M. Diehl, R. Findeisen, F. Allgöwer, H. G. Bock, and J. P. Schlöder. “Nominal stability of real-time iteration scheme for nonlinear model predictive control”. In: *IEE Proceedings-Control Theory and Applications* 152.3 (2005), pp. 296–308.

[Die+06] M. Diehl, P. Kühl, H. G. Bock, and J. P. Schlöder. “Schnelle Algorithmen für die Zustands-und Parameterschätzung auf bewegten Horizonten (Fast Algorithms for State and Parameter Estimation on Moving Horizons)”. In: *at–Automatisierungstechnik* 54.12 (2006), pp. 602–613.

[DS14] M. Durea and R. Strugariu. *An Introduction to Nonlinear Optimization Theory*. Warschau/Berlin: De Gruyter Open, 2014.

[Ech14] M. Echim. “Modellbasierte optimale Mehrgrößenregelung eines aufgeladenen Dieselmotors mittels Methoden der nichtlinearen Optimierung .” PhD thesis. University Bremen, 2014.

[FA02] R. Findeisen and F. Allgöwer. “An introduction to nonlinear model predictive control”. In: *21st Benelux meeting on systems and control*. Vol. 11. Citeseer. 2002, pp. 119–141.

[FA04] R. Findeisen and F. Allgöwer. “Computational Delay in Nonlinear Model Predictive Control”. In: *7th International Symposium on Advanced Control of Chemical Processes (ADCHEM 2003)*. Vol. 37. 1. IFAC. 2004, pp. 427–432.

[Fel60a] A. A. Feldbaum. “Dual control theory. II”. In: *Avtomatika i Telemekhanika* 21.11 (1960), pp. 1453–1464.

[Fel60b] A. A. Feldbaum. “Dual control theory. I”. In: *Avtomatika i Telemekhanika* 21.9 (1960), pp. 1240–1249.

[FH18] X. Feng and B. Houska. “Real-time algorithm for self-reflective model predictive control”. In: *Journal of Process Control* 65 (2018), pp. 68–77.

[Fia+83] A. V. Fiacco et al. “Introduction to sensitivity and stability analysis in nonlinear programming”. In: *Mathematics in Science and Engineering* 165 (1983).

[Fla13] K. Flaßkamp. “On the optimal control of mechanical systems-hybrid control strategies and hybrid dynamics”. PhD thesis. University of Paderborn, 2013.

[Fle70] R. Fletcher. “A new approach to variable metric algorithms”. In: *The computer journal* 13.3 (1970), pp. 317–322.

[FM08] G. Franceschini and S. Macchietto. “Model-based design of experiments for parameter precision: State of the art”. In: *Chemical Engineering Science* 63.19 (2008). Model-Based Experimental Analysis, pp. 4846–4872.

[Fra+12] J. V. Frasch, L. Wirsching, S. Sager, and H. G. Bock. “Mixed-Level Iteration Schemes for Nonlinear Model Predictive Control”. In: *IFAC Proceedings Volumes* 45.17 (2012). 4th IFAC Conference on Nonlinear Model Predictive Control, pp. 138–144.

[Fri+15] G. Frison, M. Vukov, N. K. Poulsen, M. Diehl, and J. B. Jørgensen. “High-Performance Small-Scale Solvers for Moving Horizon Estimation”. In: *IFAC-PapersOnLine* 48.23 (2015), pp. 80–86.

[Gef17] S. Geffken. “Effizienzsteigerung numerischer Verfahren der nichtlinearen Optimierung”. PhD thesis. University of Bremen, 2017.

[Ger18] M. Gerdts. “Numerical experiments with multistep model-predictive control approaches and sensitivity updates for the tracking control of cars”. In: *arXiv preprint arXiv:1809.00577* (2018).

[Ges+17] J. Gesenhues, M. Hein, M. Ketelhut, T. Albin, and D. Abel. “Towards Medical Cyber-Physical Systems: Modelica and FMI based Online Parameter Identification of the Cardiovascular System.” In: *Modelica*. 2017, pp. 132–068.

[GJ16] L. Grüne and O. Junge. *Gewöhnliche Differentialgleichungen: eine Einführung aus der Perspektive der dynamischen Systeme. 2. Auflage*. Springer Spektrum, 2016.

[GK02] C. Geiger and C. Kanzow. *Theorie und Numerik restringierter Optimierungsaufgaben*. Berlin, Heidelberg: Springer, 2002.

[Gle23] C. J. E. Gletter. “Untersuchung datenbasierter, nichtlinearer Regelungsmethoden für die Entwicklung individuell-optimaler Hybridfahrzeug-Betriebsstrategien”. PhD thesis. Universität Ulm, 2023.

[GMW81] P. E. Gill, W. Murray, and M. H. Wright. *Practical optimization.* Academic Press, 1981.

[Gol70] D. Goldfarb. “A family of variable-metric methods derived by variational means”. In: *Mathematics of computation* 24.109 (1970), pp. 23–26.

[GP15] L. Grüne and V. G. Palma. “Robustness of performance and stability for multistep and updated multistep MPC schemes”. In: *Discrete & Continuous Dynamical Systems* 35.9 (2015), p. 4385.

[GP17] L. Grüne and J. Pannek. *Nonlinear model predictive control, Theory and Algorithms, Second Edition.* Springer, 2017.

[Gra12] K. Graichen. *Methoden der Optimierung und optimalen Steuerung.* lecture notes, University of Ulm. 2012.

[Grü+10] L. Grüne, J. Pannek, M. Seehafer, and K. Worthmann. “Analysis of unconstrained nonlinear MPC schemes with time varying control horizon”. In: *SIAM Journal on Control and Optimization* 48.8 (2010), pp. 4938–4962.

[Grü18] L. Grüne. *Mathematische Kontrolltheorie.* lecture notes, University of Bayreuth. 2018.

[Han+22] D. Hanover, P. Foehn, S. Sun, E. Kaufmann, and D. Scaramuzza. “Performance, Precision, and Payloads: Adaptive Nonlinear MPC for Quadrotors”. In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 690–697.

[Hed13] R. Hedjar. “Adaptive neural network model predictive control”. In: *International Journal of Innovative Computing, Information and Control* 9.3 (2013), pp. 1245–1257.

[Her+18] M. Hertneck, J. Koehler, S. Trimpe, and F. Allgöwer. “Learning an Approximate Model Predictive Controller with Guarantees”. In: *IEEE Control Systems Letters* 2.3 (2018), pp. 543–548.

[Hes09] J. P. Hespanha. *Linear systems theory.* Princeton university press, 2009.

[Heu95] H. Heuser. *Gewöhnliche Differentialgleichungen : Einführung in Lehre und Gebrauch.* 3. Auflage. Mathematische Leitfäden. Wiesbaden: Vieweg+Teubner Verlag, 1995.

[Hew+20] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger. “Learning-based model predictive control: Toward safe learning in control”. In: *Annual Review of Control, Robotics, and Autonomous Systems* 3 (2020), pp. 269–296.

[Hou+17] B. Houska, D. Telen, F. Logist, and J. V. Impe. “Self-reflective model predictive control”. In: *SIAM Journal on Control and Optimization* 55.5 (2017), pp. 2959–2980.

[HRS06] C. Heij, A. C. Ran, and F. van Schagen. *Introduction to mathematical systems theory: linear systems, identification and control.* Springer Science & Business Media, 2006.

[Hua+21] L. Huang, J. Coulson, J. Lygeros, and F. Dörfler. "Decentralized data-enabled predictive control for power system oscillation damping". In: *IEEE Transactions on Control Systems Technology* 30.3 (2021), pp. 1065–1077.

[HV19] B. Houska and M. E. Villanueva. "Robust optimization for MPC". In: *Handbook of model predictive control*. Springer, 2019, pp. 413–443.

[JUD95] S. J. Julier, J. K. Uhlmann, and H. F. Durrant-Whyte. "A new approach for filtering nonlinear systems". In: *Proceedings of American Control Conference - ACC'95*. Vol. 3. IEEE. 1995, pp. 1628–1632.

[Jun15] D. Jungnickel. *Optimierungsmethoden: Eine Einführung*. 3. Auflage. Springer-Lehrbuch. Berlin: Springer Spektrum, 2015.

[Jun19] F. Jung. "Entwicklung robuster Prognosen für ein Energiemanagementsystem anhand datenbasierter Modellierungsverfahren unter Berücksichtigung von Unsicherheiten". PhD thesis. University of Bremen, 2019.

[Kal60] R. E. Kalman. "A New Approach to Linear Filtering and Prediction Problems". In: *Journal of Basic Engineering* 82.1 (Mar. 1960), pp. 35–45.

[KB12] M. Knauer and C. Büskens. "From WORHP to TransWORHP". In: *Proceedings of the 5th International Conference on Astrodynamics Tools and Techniques*. 2012.

[KB19] M. Knauer and C. Büskens. "Real-Time Optimal Control Using TransWORHP and WORHP Zen". In: *Modeling and Optimization in Space Engineering*. Springer, 2019, pp. 211–232.

[KB60] R. E. Kalman and R. S. Bucy. "New Results in Linear Filtering and Prediction Theory". In: *J. Basic Eng., ASME Trans., ser. D.* 83 (1960), pp. 95–107.

[Kem15] A. E. Kemper. "Modellbasierte optimale Mehrgrößenregelung und optimale Reglerparametrisierung für Luftsysteme von Pkw-Dieselmotoren". PhD thesis. University of Bremen, 2015.

[KGB18] R. Kuhlmann, S. Geffken, and C. Büskens. "WORHP Zen: Parametric Sensitivity Analysis for the Nonlinear Programming Solver WORHP". In: *Operations Research Proceedings 2017*. Ed. by N. Kliewer, J. F. Ehmke, and R. Borndörfer. Springer International Publishing, 2018, pp. 649–654.

[Kir+10] C. Kirches, L. Wirsching, S. Sager, and H. G. Bock. "Efficient numerics for nonlinear model predictive control". In: *Recent Advances in Optimization and its Applications in Engineering*. Springer, 2010, pp. 339–357.

[Kna01] M. Knauer. "Sensitivitätsanalyse verschiedener Gütekriterien bei der optimalen Bahnplanung von Industrierobotern". Diplomarbeit. University of Bayreuth, 2001.

[Kol+18] T. Koller, F. Berkenkamp, M. Turchetta, and A. Krause. "Learning-based model predictive control for safe exploration". In: *2018 IEEE conference on decision and control (CDC)*. IEEE. 2018, pp. 6059–6066.

[KS72] H. Kwakernaak and R. Sivan. *Linear optimal control systems*. Vol. 1. Wiley-interscience New York, 1972.

[Küp+09] A. Küpper, M. Diehl, J. P. Schlöder, H. G. Bock, and S. Engell. “Efficient moving horizon state and parameter estimation for the varicol smb process”. In: *IFAC Proceedings Volumes* 42.11 (2009), pp. 590–595.

[La+16] H. La, A. Potschka, J. Schlöder, and H. Bock. “Dual control and information gain in controlling uncertain processes”. In: *IFAC-PapersOnLine* 49.7 (2016), pp. 139–144.

[La16] H. C. La. “Dual control for nonlinear model predictive control”. PhD thesis. Ruprecht-Karls-Universität Heidelberg, 2016.

[Lan74] I. Landau. “A survey of model reference adaptive techniques—Theory and applications”. In: *Automatica* 10.4 (1974), pp. 353–379.

[LCA19] M. Lorenzen, M. Cannon, and F. Allgöwer. “Robust MPC with recursive model update”. In: *Automatica* 103 (2019), pp. 461–471.

[Lee11] J. H. Lee. “Model predictive control: Review of the three decades of development”. In: *International Journal of Control, Automation and Systems* 9.3 (2011), pp. 415–424.

[Lib11] D. Liberzon. *Calculus of variations and optimal control theory: a concise introduction*. Princeton University Press, 2011.

[Liu+22] X. Liu, D. Jiang, B. Tao, G. Jiang, Y. Sun, J. Kong, X. Tong, G. Zhao, and B. Chen. “Genetic algorithm-based trajectory optimization for digital twin robots”. In: *Frontiers in Bioengineering and Biotechnology* 9:793782 (2022).

[LK18] S. Lucia and B. Karg. “A deep learning-based approach to robust nonlinear model predictive control”. In: *IFAC-PapersOnLine* 51.20 (2018). 6th IFAC Conference on Nonlinear Model Predictive Control NMPC 2018, pp. 511–516.

[LP14] S. Lucia and R. Paulen. “Robust nonlinear model predictive control with reduction of uncertainty via robust optimal experiment design”. In: *IFAC Proceedings Volumes* 47.3 (2014), pp. 1904–1909.

[Lun16] J. Lunze. *Regelungstechnik 1: Systemtheoretische Grundlagen, Analyse und Entwurf einschleifiger Regelungen*. 11. Berlin: Springer, 2016.

[Mag+02] L. Magni, G. D. Nicolao, R. Scattolini, and F. Allgöwer. “Robust receding horizon control for nonlinear discrete-time systems”. In: *IFAC Proceedings Volumes* 35.1 (2002). 15th IFAC World Congress, pp. 403–408.

[May14] D. Q. Mayne. “Model predictive control: Recent developments and future promise”. In: *Automatica* 50.12 (2014), pp. 2967–2986.

[Mes+14] A. Mesbah, S. Streif, R. Findeisen, and R. D. Braatz. “Stochastic nonlinear model predictive control with probabilistic constraints”. In: *2014 American Control Conference*. 2014, pp. 2413–2419.

[Mes16] A. Mesbah. “Stochastic model predictive control: An overview and perspectives for future research”. In: *IEEE Control Systems Magazine* 36.6 (2016), pp. 30–44.

[Mes18] A. Mesbah. “Stochastic model predictive control with active uncertainty learning: A survey on dual control”. In: *Annual Reviews in Control* 45 (2018), pp. 107–117.

[ML99] M. Morari and J. H. Lee. “Model predictive control: past, present and future”. In: *Computers & Chemical Engineering* 23.4-5 (1999), pp. 667–682.

[MM17] R. Miranda-Colorado and J. Moreno-Valenzuela. “An efficient on-line parameter identification algorithm for nonlinear servomechanisms with an algebraic technique for state estimation”. In: *Asian Journal of Control* 19.6 (2017), pp. 2127–2142.

[MM93] D. Mayne and H. Michalska. “Adaptive receding horizon control for constrained nonlinear systems”. In: *Proceedings of 32nd IEEE Conference on Decision and Control.* 1993, 1286–1291 vol.2.

[Mur+94] R. M. Murray, Z. Li, S. S. Sastry, and S. S. Sastry. *A mathematical introduction to robotic manipulation.* CRC press, 1994.

[NAD19] A. Nurkanović, S. Albrecht, and M. Diehl. “Multi-level iterations for economic nonlinear model predictive control”. In: *Recent Advances in Model Predictive Control.* Ed. by T. Faulwasser, M. A. Müller, and K. Worthmann. 2019.

[Nel01] O. Nelles. *Nonlinear system identification. From classical approaches to neural networks and fuzzy models.* Engineering online library. Berlin: Springer, 2001.

[NW06] J. Nocedal and S. J. Wright. *Numerical optimization.* Second edition. Springer, 2006.

[Pon+62] L. S. Pontryagin, E. Mishchenko, V. Boltyanskii, and R. Gamkrelidze. *The mathematical theory of optimal processes.* Wiley, 1962.

[Pow78] M. J. Powell. “A fast algorithm for nonlinearly constrained optimization calculations”. In: *Numerical analysis.* Springer, 1978, pp. 144–157.

[Rau14] S. Rauški. “Limited Memory BFGS method for Sparse and Large-Scale Nonlinear Optimization”. PhD thesis. University of Bremen, 2014.

[Rec19] B. Recht. “A tour of reinforcement learning: The view from continuous control”. In: *Annual Review of Control, Robotics, and Autonomous Systems* 2 (2019), pp. 253–279.

[RFB18] M. Runge, K. Flaßkamp, and C. Büskens. “Sequential solution of parameter identification and optimal control problems for robotic systems”. In: *PAMM* 18.1 (2018), e201800099.

[RFB20] M. Runge, K. Flaßkamp, and C. Büskens. “Model Predictive Control with Online Nonlinear Parameter Identification for a Robotic System.” In: *2020 7th International Conference on Control, Decision and Information Technologies (CoDIT)*. 2020, pp. 312–318.

[RFB21] M. Runge, K. Flaßkamp, and C. Büskens. “Real-time parameter estimation for sensitivity-based LQ regulator adaptation”. In: *PAMM* 20.1 (2021), e202000292.

[Ric+19] M. Rick, J. Clemens, L. Sommer, A. Folkers, K. Schill, and C. Büskens. “Autonomous Driving Based on Nonlinear Model Predictive Control and Multi-Sensor Fusion.” In: *IFAC-PapersOnLine* 52.8 (2019). 10th IFAC Symposium on Intelligent Autonomous Vehicles IAV 2019, pp. 182–187.

[RL19] S. V. Raković and W. S. Levine. *Handbook of Model Predictive Control.* Control Engineering. Springer International Publishing, 2019.

[RMD17] J. B. Rawlings, D. Q. Mayne, and M. Diehl. *Model predictive control: theory, computation, and design.* Vol. 2. Nob Hill Publishing Madison, WI, 2017.

[RRM03] C. V. Rao, J. B. Rawlings, and D. Q. Mayne. “Constrained state estimation for nonlinear discrete-time systems: Stability and moving horizon approximations”. In: *IEEE transactions on automatic control* 48.2 (2003), pp. 246–258.

[RS00] W. J. Rugh and J. S. Shamma. “Research on gain scheduling”. In: *Automatica* 36.10 (2000), pp. 1401–1425.

[RWB17] S. Roy, H. Wernsing, and C. Büskens. “Optimization of ship manoeuvring within the project GALILEOnautic”. In: *PAMM* 17.1 (2017), pp. 813–814.

[Sch+18] K. Schäfer, M. Runge, K. Flaßkamp, and C. Büskens. “Parameter identification for dynamical systems using optimal control techniques”. In: *2018 European Control Conference (ECC)*. IEEE. 2018, pp. 137–142.

[Sch+22] P. Schmitz, A. Engelmann, T. Faulwasser, and K. Worthmann. “Data-driven MPC of descriptor systems: A case study for power networks”. In: *IFAC-PapersOnLine* 55.30 (2022), pp. 359–364.

[Sch02] K. Schittkowski. *Numerical data fitting in dynamical systems: a practical introduction with applications and software.* Vol. 77. Springer Science & Business Media, 2002.

[Sch08] K. Schittkowski. “Parameter identification and model verification in systems of partial differential equations applied to transdermal drug delivery”. In: *Mathematics and Computers in Simulation* 79.3 (2008), pp. 521–538.

[Sch14] C. Schomakers. “Nichtlineare modellprädiktive Regelung für die Bahnplanung eines Wiedereintrittsproblems”. MA thesis. University of Bremen, 2014.

[Sch17] D. Schröder. *Intelligente Verfahren.* Berlin Heidelberg: Springer, 2017.

[See17] D. Seelbinder. “On-board trajectory computation for Mars atmospheric entry based on parametric sensitivity analysis of optimal control problems”. PhD thesis. University of Bremen, 2017.

[SFH18] P. Stapor, F. Fröhlich, and J. Hasenauer. “Optimization and profile calculation of ODE models using second order adjoint sensitivity analysis”. In: *Bioinformatics* 34.13 (2018), pp. i151–i159.

[SFW22] P. Schmitz, T. Faulwasser, and K. Worthmann. “Willems’ fundamental lemma for linear descriptor systems and its use for data-driven output-feedback MPC”. In: *IEEE Control Systems Letters* 6 (2022), pp. 2443–2448.

[Sha08] D. Sha. “A new neural networks based adaptive model predictive control for unknown multiple variable non-linear systems”. In: *International Journal of Advanced Mechatronic Systems* 1.2 (2008), pp. 146–155.

[Sha70] D. F. Shanno. “Conditioning of quasi-Newton methods for function minimization”. In: *Mathematics of computation* 24.111 (1970), pp. 647–656.

[Sic08] B. Siciliano. *Springer handbook of robotics : with 84 tables*. Online-Ressource (LX, 1611 S.) : Ill., graph. Darst. Berlin [u.a.]: Springer, 2008.

[SK11] H. R. Schwarz and N. Köckler. “Numerische Integration”. In: *Numerische Mathematik*. Wiesbaden: Vieweg+Teubner Verlag, 2011, pp. 307–341.

[Sor70] H. W. Sorenson. “Least-squares estimation: from Gauss to Kalman”. In: *IEEE spectrum* 7.7 (1970), pp. 63–68.

[SVD07] J. Swevers, W. Verdonck, and J. De Schutter. “Dynamic model identification for industrial robots”. In: *IEEE control systems magazine* 27.5 (2007), pp. 58–71.

[Tan+14] M. Tanaskovic, L. Fagiano, R. Smith, and M. Morari. “Adaptive receding horizon control for constrained MIMO systems”. In: *Automatica* 50.12 (2014), pp. 3019–3029.

[Tao+22] F. Tao, B. Xiao, Q. Qi, J. Cheng, and P. Ji. “Digital twin modeling”. In: *Journal of Manufacturing Systems* 64 (2022), pp. 372–389.

[Ter+19] E. Terzi, L. Fagiano, M. Farina, and R. Scattolini. “Learning-based predictive control for linear systems: a unitary approach”. In: *Automatica* 108 (2019).

[Tie12] J. Tietjen. “Optimalitätsbedingungen, parametrische Sensitivitätsanalyse und Echtzeitanpassung optimaler Regel- und Schätzverfahren”. PhD thesis. University of Bremen, 2012.

[VP19] J. Valluru and S. C. Patwardhan. “An Integrated Frequent RTO and Adaptive Nonlinear MPC Scheme Based on Simultaneous Bayesian State and Parameter Estimation”. In: *Industrial & Engineering Chemistry Research* 58.18 (2019), pp. 7561–7578.

[VTK04] H. U. Voss, J. Timmer, and J. Kurths. “Nonlinear dynamical system identification from uncertain and indirect measurements”. In: *International Journal of Bifurcation and Chaos* 14.06 (2004), pp. 1905–1933.

[Wab+21] K. P. Wabersich, L. Hewing, A. Carron, and M. N. Zeilinger. “Probabilistic model predictive safety certification for learning-based control”. In: *IEEE Transactions on Automatic Control* (2021).

[Was13] D. Wassel. “Exploring novel designs of NLP solvers: Architecture and Implementation of WORHP”. PhD thesis. University of Bremen, 2013.

[Wer18] H. Wernsing. “PDE-restringierte Optimierung in Anwendungen der spanenden Trockenbearbeitung”. PhD thesis. University of Bremen, 2018.

[WG09] A. Walther and A. Griewank. “Getting Started with ADOL-C.” In: *Combinatorial scientific computing* 09061 (2009), pp. 181–202.

[Wil+05] J. C. Willems, P. Rapisarda, I. Markovsky, and B. L. De Moor. “A note on persistency of excitation”. In: *Systems & Control Letters* 54.4 (2005), pp. 325–329.

[Wit75] B. Wittenmark. “Stochastic adaptive control methods: a survey”. In: *International Journal of Control* 21.5 (1975), pp. 705–730.

[WSM14] A. D. Wilson, J. A. Schultz, and T. D. Murphey. “Trajectory synthesis for Fisher information maximization”. In: *IEEE Transactions on Robotics* 30.6 (2014), pp. 1358–1370.

[WSM15] A. D. Wilson, J. A. Schultz, and T. D. Murphey. “Trajectory optimization for well-conditioned parameter estimation”. In: *IEEE Transactions on Automation Science and Engineering* 12.1 (2015), pp. 28–36.

[YB13] X. Yang and L. T. Biegler. “Advanced-multi-step nonlinear model predictive control”. In: *Journal of process control* 23.8 (2013), pp. 1116–1128.

[YG18] Y. Yoon and B. Gur. “Parameter Estimation and Control of Nonholonomic Mobile Robots: A Model-Based Approach”. In: *2018 6th International Conference on Control Engineering & Information Technology (CEIT)*. IEEE. 2018, pp. 1–6.

[YLP20] X. Yang, Y. Liu, and G.-K. Park. “Parameter estimation of uncertain differential equation with application to financial market”. In: *Chaos, Solitons & Fractals* 139 (2020), p. 110026.

[ZB09] V. M. Zavala and L. T. Biegler. “The advanced-step NMPC controller: Optimality, stability and robustness”. In: *Automatica* 45.1 (2009), pp. 86–93.

[ZD98] K. Zhou and J. C. Doyle. *Essentials of robust control.* Prentice Hall, 1998.