Chapter 2

Mathematical Background

In this chapter, we concisely introduce the mathematical background necessary for the subsequent chapters. In Section 2.1, we introduce general optimization problems in Banach spaces and certain Banach spaces which will be of interest in the context of Optimal Control Problems (OCPs). Section 2.2 is dedicated to Nonlinear Programming. We state first-order necessary conditions for Nonlinear Programming Problems (NLPs) and present selected solution methods. In Section 2.3, we consider the optimal control of dynamic systems. We introduce different types of dynamic systems, present a general problem formulation for OCPs, and discuss techniques for problem transformations. For the numerical solution of OCPs, in Section 2.4 we present two so-called direct solution approaches, namely Direct Multiple Shooting and Direct Collocation. Section 2.5 deals with different approaches for the numerical computation of derivatives including sensitivities.

2.1 Optimization in Banach Spaces

In this section, we state a general problem formulation for optimization problems in Banach spaces and introduce certain Banach spaces of interest. We follow the presentation in [60, ch. 2] and [102, ch. 2-3].

In this thesis, we consider optimization problems which fit into the following setting: let $(X, \|\cdot\|)$ be a Banach space over $\mathbb{R}$, $\Sigma \subseteq X$ a non-empty subset and $f : X \to \mathbb{R}$ a functional. A general optimization problem is given by

$$\begin{aligned} &\min_{\mathbf{x}\in X} \; f(\mathbf{x}) \\ &\text{s.t. } \mathbf{x} \in \Sigma. \end{aligned} \tag{2.1}$$

The functional f is called objective function or cost function. The set Σ is called the feasible set, and a vector $\mathbf{x} \in X$ is feasible for Problem (2.1) if and only if $\mathbf{x} \in \Sigma$. We consider X together with the norm topology (i. e., the topology induced by the metric $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|$) as a topological space.

Definition 2.1 (Solutions of Optimization Problems)
We call a vector $\mathbf{x}^* \in \Sigma \subseteq X$

- a local solution (or local minimum) of Problem (2.1) if there is an open neighborhood $U \subseteq X$ of $\mathbf{x}^*$ such that

  $$f(\mathbf{x}^*) \leq f(\mathbf{x}) \text{ for all } \mathbf{x} \in U \cap \Sigma,$$

  and a strict local solution (or strict local minimum) if the above inequality is strictly satisfied for all $\mathbf{x} \neq \mathbf{x}^*$ in $U \cap \Sigma$,

- a global solution (or global minimum) of Problem (2.1) if

  $$f(\mathbf{x}^*) \leq f(\mathbf{x}) \text{ for all } \mathbf{x} \in \Sigma,$$

  and the strict global solution (or strict global minimum) if the above inequality is strictly satisfied for all $\mathbf{x} \neq \mathbf{x}^*$ in Σ,

- the unique solution of Problem (2.1) if $\mathbf{x}^*$ is the only local solution. In this case we call Problem (2.1) uniquely solvable. △

When using the term "a solution" in this thesis, we consider a local solution unless stated otherwise. For results on the existence of solutions of Problem (2.1) and necessary conditions we refer to [60, sec. 2.3].

For the remainder of this section, let $\mathcal{T} = [t_0, t_f] \subset \mathbb{R}$ with $t_0 < t_f$. We now introduce certain Banach spaces which will be of interest later.

Definition 2.2

- Let $1 \leq p < \infty$. The Lebesgue space of equivalence classes of measurable functions $f : \mathcal{T} \to \mathbb{R}$ for which $|f(\cdot)|^p$ is Lebesgue-integrable is denoted by $L^p(\mathcal{T}, \mathbb{R})$. Here, we identify functions which equal each other almost everywhere in $\mathcal{T}$ with respect to the Lebesgue measure. We equip the latter space with the norm

  $$\|f(\cdot)\|_p = \int_{t_0}^{t_f} |f(t)|^p \, \mathrm{d}t.$$

- The Lebesgue space of equivalence classes of essentially bounded measurable functions $f : \mathcal{T} \to \mathbb{R}$ is denoted by $L^\infty(\mathcal{T}, \mathbb{R})$. Again, we identify functions which

equal each other almost everywhere in $\mathcal{T}$ with respect to the Lebesgue measure. We equip $L^\infty(\mathcal{T},\mathbb{R})$ with the norm

$$\|f(\cdot)\|_\infty = \operatorname*{ess\,sup}_{t\in\mathcal{T}} |f(t)|\,.$$

- For $p \in [1,\infty]$ we define

$$L^p\left(\mathcal{T},\mathbb{R}^n\right) \stackrel{\text{def}}{=} \underbrace{L^p(\mathcal{T},\mathbb{R}) \times \cdots \times L^p(\mathcal{T},\mathbb{R})}_{n\text{-times}}$$

and equip $L^p(\mathcal{T},\mathbb{R}^n) \ni \mathbf{f}(\cdot)$ with the norm $\|\mathbf{f}(\cdot)\|_p \stackrel{\text{def}}{=} \max_{j=1,\dots,n} \|\mathbf{f}_j(\cdot)\|_p$. △

For $p \in [1,\infty]$ the space $L^p(\mathcal{T},\mathbb{R})$ together with the stated norm is a Banach space, cf. [91, Theorem 2.8.2] and [91, Theorem 2.11.7], and hence the same holds for the product space $L^p(\mathcal{T},\mathbb{R}^n)$. In this thesis, particularly the case $p = \infty$ is of interest since in OCPs (see Section 2.3) the so-called control functions are typically chosen to be elements of $L^\infty(\mathcal{T},\mathbb{R}^n)$.

Definition 2.3
A function $f : \mathcal{T} = [t_0, t_f] \to \mathbb{R}$ is called absolutely continuous if for all $\varepsilon > 0$, there is a $\delta > 0$ such that for all $m \in \mathbb{N}$ and $t_0 \le a_1 < b_1 \le a_2 < b_2 \le \cdots \le a_n < b_m \le t_f$ the implication

$$\sum_{i=1}^{m} |b_i - a_i| < \delta \implies \sum_{i=1}^{m} |f(b_i) - f(a_i)| < \varepsilon$$

holds. △

We summarize several results on absolutely continuous functions which can be found, e. g., in [109, ch. 9]. An absolutely continuous function $f : \mathcal{T} \to \mathbb{R}$ is continuous, in particular essentially bounded, and differentiable almost everywhere with Lebesgue-integrable derivative. We consider the derivative as an element of $L^1(\mathcal{T},\mathbb{R})$ and denote it by $\frac{\mathrm{d}}{\mathrm{d}t} f(\cdot)$. We have

$$f(t) = f(t_0) + \int_{t_0}^{t} \frac{\mathrm{d}}{\mathrm{d}t} f(\tau)\,\mathrm{d}\tau\,.$$

Furthermore, for any $g \in L^1(\mathcal{T},\mathbb{R})$ the function $G(t) = g(t_0) + \int_{t_0}^{t} g(\tau)\,\mathrm{d}\tau$ is absolutely continuous with $\frac{\mathrm{d}}{\mathrm{d}t} G(t) = g(t)$ as element of $L^1(\mathcal{T},\mathbb{R})$.

Definition 2.4
The space of absolutely continuous functions $f : \mathcal{T} \to \mathbb{R}$ with essentially bounded derivatives is denoted by $W^{1,\infty}(\mathcal{T},\mathbb{R})$. We equip $W^{1,\infty}(\mathcal{T},\mathbb{R})$ with the norm

$$\|f(\cdot)\|_{1,\infty} \overset{\text{def}}{=} \max\left(\|f(\cdot)\|_\infty, \left\|\frac{\mathrm{d}}{\mathrm{d}t} f(\cdot)\right\|_\infty\right).$$

Furthermore, we define

$$W^{1,\infty}\left(\mathcal{T},\mathbb{R}^n\right) \overset{\text{def}}{=} \overbrace{W^{1,\infty}(\mathcal{T},\mathbb{R}) \times \cdots \times W^{1,\infty}(\mathcal{T},\mathbb{R})}^{n\text{-times}}$$

and equip $W^{1,\infty}(\mathcal{T},\mathbb{R}^n) \ni \mathbf{f}(\cdot)$ with the norm $\|\mathbf{f}(\cdot)\|_{1,\infty} \overset{\text{def}}{=} \max_{j=1,\dots,n} \|\mathbf{f}_j(\cdot)\|_{1,\infty}$. △

The space $W^{1,\infty}(\mathcal{T},\mathbb{R})$ together with the stated norm is a Banach space, cf. [102, sec. 2.4] and the references therein, and hence the same holds for the product space $W^{1,\infty}(\mathcal{T},\mathbb{R}^n)$. In this thesis, these spaces are of interest since in OCPs (see Section 2.3) they are a natural choice for the so-called differential states which obey a differential equation.

2.2 Nonlinear Programming

In this section, we give a concise introduction to Nonlinear Programming. We introduce the problem formulation and state first-order necessary conditions for optimality. Subsequently, we present selected solution methods.

2.2.1 Problem Formulation and First-Order Necessary Conditions

This subsection is based on the textbooks [59], [110], and [144]. We consider a general optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \; f(\mathbf{x}) \tag{2.2a}$$
$$\text{s.t. } \mathbf{h}_j(\mathbf{x}) = 0, \quad j = 1,\dots,p, \tag{2.2b}$$
$$\mathbf{g}_i(\mathbf{x}) \le 0, \quad i = 1,\dots,m, \tag{2.2c}$$

with continuous (and potentially nonlinear) functions $f : \mathbb{R}^n \to \mathbb{R}$, $\mathbf{h} : \mathbb{R}^n \to \mathbb{R}^p$, and $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^m$. Such a problem is called an NLP. Note that Problem (2.2) is of Form (2.1). We denote the feasible set by $\mathcal{F}$. For a given $\mathbf{x} \in \mathcal{F}$,

$$\mathcal{A}(\mathbf{x}) = \left\{ i \in \{1,\dots,m\} \,\middle|\, \mathbf{g}_i(\mathbf{x}) = 0 \right\}$$

is the index set of active inequality constraints at $\mathbf{x}$. In the below presentation, we assume $f(\cdot)$, $\mathbf{h}(\cdot)$, and $\mathbf{g}(\cdot)$ to be continuously differentiable. The Lagrangian (function) $L : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ of Problem (2.2) is given by

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \stackrel{\text{def}}{=} f(\mathbf{x}) + \sum_{i=1}^{m} \boldsymbol{\lambda}_i \mathbf{g}_i(\mathbf{x}) + \sum_{j=1}^{p} \boldsymbol{\mu}_j \mathbf{h}_j(\mathbf{x}) .$$

Definition 2.5
We consider Problem (2.2). Let $\boldsymbol{\lambda} \in \mathbb{R}^m$ and $\boldsymbol{\mu} \in \mathbb{R}^p$. The conditions

$$\nabla_{\mathbf{x}} L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \mathbf{0}, \tag{2.3a}$$
$$\mathbf{h}(\mathbf{x}) = \mathbf{0}, \tag{2.3b}$$
$$\boldsymbol{\lambda}_i \geq 0, \quad i = 1, \ldots, m, \tag{2.3c}$$
$$\mathbf{g}_i(\mathbf{x}) \leq 0, \quad i = 1, \ldots, m, \tag{2.3d}$$
$$\boldsymbol{\lambda}_i \mathbf{g}_i(\mathbf{x}) = 0, \quad i = 1, \ldots, m \tag{2.3e}$$

are called Karush-Kuhn-Tucker (KKT) conditions of Problem (2.2). A vector $(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p$ which satisfies the KKT conditions is called a KKT point of Problem (2.2). △

If additional conditions – so-called Constraint Qualifications (CQs) – are satisfied, the KKT conditions are first-order necessary conditions for optimality. We take a look at CQs.

Definition 2.6
A subset $C \subseteq \mathbb{R}^n$ is called a cone, if $r\mathbf{x} \in C$ for all $r > 0$ and $\mathbf{x} \in C$. If $C \subseteq \mathbb{R}^n$ is a non-empty cone, the set

$$C^\circ = \left\{ \mathbf{y} \in \mathbb{R}^n \,\middle|\, \mathbf{y}^T \mathbf{x} \leq 0 \text{ for all } \mathbf{x} \in C \right\}$$

is called the polar cone of C. △

Definition 2.7
Let the feasible set $\mathcal{F}$ of Problem (2.2) be non-empty and $\mathbf{x} \in \mathcal{F}$. The set

$$\mathcal{T}(\mathbf{x}) \stackrel{\text{def}}{=} \left\{ \mathbf{d} \in \mathbb{R}^n \,\middle|\, \exists t_k > 0, \mathbf{x}^k \in \mathcal{F} \text{ with } \lim_{k\to\infty} \mathbf{x}^k = \mathbf{x}, \lim_{k\to\infty} t_k = 0, \text{ and } \lim_{k\to\infty} \frac{\mathbf{x}^k - \mathbf{x}}{t_k} = \mathbf{d} \right\}$$

is called the tangent cone at $\mathbf{x}$ and

$$\mathcal{T}_{\text{lin}}(\mathbf{x}) \overset{\text{def}}{=} \left\{ \mathbf{d} \in \mathbb{R}^n \;\middle|\; \begin{array}{ll} \nabla \mathbf{g}_i(\mathbf{x})^T \mathbf{d} = 0 & \text{for } i \in \mathcal{A}(\mathbf{x}), \\ \nabla \mathbf{h}_j(\mathbf{x})^T \mathbf{d} = 0 & \text{for } j = 1, \ldots, p \end{array} \right\}$$

the linearized tangent cone at $\mathbf{x}$. △

For a given $\mathbf{x} \in \mathcal{F}$, both sets $\mathcal{T}(\mathbf{x})$ and $\mathcal{T}_{\text{lin}}(\mathbf{x})$ are indeed non-empty cones. We have $\mathcal{T}(\mathbf{x}) \subseteq \mathcal{T}_{\text{lin}}(\mathbf{x})$, cf. [144, Lemma 16.5], and consequently $\mathcal{T}_{\text{lin}}(\mathbf{x})^\circ \subseteq \mathcal{T}(\mathbf{x})^\circ$ for the polar cones.

Definition 2.8
Let $\mathbf{x} \in \mathcal{F}$. The condition $\mathcal{T}_{\text{lin}}(\mathbf{x})^\circ = \mathcal{T}(\mathbf{x})^\circ$ is called Guignard Constraint Qualification (GCQ) at $\mathbf{x}$. Any condition which implies GCQ at $\mathbf{x}$ is called a CQ at $\mathbf{x}$. △

Definition 2.9
Let $\mathbf{x} \in \mathcal{F}$ and $\mathcal{A}(\mathbf{x})$ be the corresponding index set of active inequality constraints.

- If the set of gradients $\{\nabla \mathbf{g}_i(\mathbf{x}) \mid i \in \mathcal{A}(\mathbf{x})\} \cup \{\nabla \mathbf{h}_j(\mathbf{x}) \mid j = 1, \ldots, p\}$ is linearly independent, we say that the Linear Independence Constraint Qualification (LICQ) holds at $\mathbf{x}$.
- If the set of gradients $\{\nabla \mathbf{h}_j(\mathbf{x}) \mid j = 1, \ldots, p\}$ is linearly independent and there is a $\mathbf{d} \in \mathbb{R}^n$ with

  $$\nabla \mathbf{g}_i(\mathbf{x})^T \mathbf{d} < 0,\ i \in \mathcal{A}(\mathbf{x}) \quad \text{and} \quad \nabla \mathbf{h}_j(\mathbf{x})^T \mathbf{d} = 0,\ j = 1, \ldots, p,$$

  we say that the Mangasarian-Fromovitz Constraint Qualification (MFCQ) is satisfied at $\mathbf{x}$. △

Theorem 2.10
Let $\mathbf{x} \in \mathcal{F}$. We have

$$\text{LICQ holds at } \mathbf{x} \implies \text{MFCQ holds at } \mathbf{x} \implies \text{GCQ holds at } \mathbf{x}.$$

In particular, LICQ and MFCQ are CQs at $\mathbf{x}$.

Proof For the first implication see [110, sec. 12.6] and a proof of the second implication can be found in [144, sec. 16.1-16.2]. □

We can now state first-order necessary conditions for local solutions of Problem (2.2).

Theorem 2.11
Let $\mathbf{x}^* \in \mathcal{F}$ be a local solution of Problem (2.2) such that a CQ is satisfied at $\mathbf{x}$. Then there are $\boldsymbol{\lambda}^* \in \mathbb{R}^m$ and $\boldsymbol{\mu}^* \in \mathbb{R}^p$ such that $(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ is a KKT point of Problem (2.2).

Proof See [144, sec. 16.1]. □

If $f(\cdot), \mathbf{h}(\cdot)$, and $\mathbf{g}(\cdot)$ are twice continuously differentiable, one can further derive second-order (necessary and sufficient) conditions. For according results we refer to [59, sec. 2.2.6] and [110, sec. 12.5], respectively.

2.2.2 Selected Solution Methods

We now present selected solution approaches for NLPs which will be applied in this thesis.

Interior-Point Methods

We sketch the idea of Interior-Point methods following [110, ch. 19] where the interested reader can find details.

The NLP (2.2) can be stated equivalently as

$$\min_{\mathbf{x}\in\mathbb{R}^n, \mathbf{s}\in\mathbb{R}^m} \quad f(\mathbf{x}) \tag{2.4a}$$
$$\text{s.t.} \quad \mathbf{0} = \mathbf{h}(\mathbf{x}), \tag{2.4b}$$
$$\mathbf{0} = \mathbf{g}(\mathbf{x}) - \mathbf{s}, \tag{2.4c}$$
$$\mathbf{s}_i \leq 0, \quad i = 1, \ldots, m, \tag{2.4d}$$

by means of a slack vector $\mathbf{s} \in \mathbb{R}^m$ with non-positive components. Let $L(\mathbf{x}, \mathbf{s}, \boldsymbol{\lambda}, \boldsymbol{\mu})$ denote the Lagrangian of the latter problem. The KKT conditions then read as

$$\mathbf{0} = \nabla_{\mathbf{x},\mathbf{s}} L(\mathbf{x}, \mathbf{s}, \boldsymbol{\lambda}, \boldsymbol{\mu}), \tag{2.5a}$$
$$\mathbf{0} = \mathbf{h}(\mathbf{x}), \tag{2.5b}$$
$$\mathbf{0} = \mathbf{g}(\mathbf{x}) - \mathbf{s}, \tag{2.5c}$$
$$\boldsymbol{\lambda}_i \geq 0, \quad i = 1, \ldots, m, \tag{2.5d}$$
$$\mathbf{s}_i \leq 0, \quad i = 1, \ldots, m, \tag{2.5e}$$
$$\mathbf{s}_i \boldsymbol{\lambda}_i = \varepsilon, \quad i = 1, \ldots, m, \tag{2.5f}$$

if $\varepsilon = 0$. For $\varepsilon = 0$ it is challenging to find a solution $(\mathbf{x}, \mathbf{s}, \boldsymbol{\lambda}, \boldsymbol{\mu})$ of (2.5) due to the Complementarity Conditions (2.5d-2.5f) since they include the determination of the optimal index set of active inequality constraints – a task of combinatorial nature [110, p. 565]. To resolve this issue, the idea of Interior-Point methods is to (approximately) solve System (2.5) for a sequence of strictly negative ε_k in order to generate a sequence $(\mathbf{x}^k, \mathbf{s}^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)$ which ideally converges to a point that satisfies (2.5) with $\varepsilon = 0$ (i. e., is a KKT point of Problem (2.4)) and furthermore is a minimizer of Problem (2.4). For details – in particular on different algorithmic realizations and convergence properties – we refer to [110, ch. 19].

Sequential Quadratic Programming

We sketch the idea of Sequential Quadratic Programming (SQP). In this paragraph, we follow [59, sec. 5.5], [110, ch. 18], and [144, ch. 19] where details can be found, respectively.

We first consider an equality constrained NLP

$$\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x}) \tag{2.6a}$$

$$\text{s.t. } \mathbf{h}(\mathbf{x}) = \mathbf{0}, \tag{2.6b}$$

with twice continuously differentiable functions $f : \mathbb{R}^n \to \mathbb{R}$ and $\mathbf{h} : \mathbb{R}^n \to \mathbb{R}^p$. Let $L(\mathbf{x}, \boldsymbol{\mu})$ denote the Lagrangian of Problem (2.6). The KKT conditions of Problem (2.6) are satisfied if and only if $(\mathbf{x}, \boldsymbol{\mu})$ is a root of the nonlinear function

$$\mathbf{F} : \mathbb{R}^{n+p} \to \mathbb{R}^{n+p}, \quad (\mathbf{x}, \boldsymbol{\mu}) \longmapsto \begin{pmatrix} \nabla_{\mathbf{x}} L(\mathbf{x}, \boldsymbol{\mu}) \\ \mathbf{h}(\mathbf{x}) \end{pmatrix}.$$

Determining a KKT point of Problem (2.6) by finding a root of $\mathbf{F}(\cdot)$ with Newton's method is known as the Lagrange-Newton method, see [59, sec. 5.5.2].

Let $(\mathbf{x}^k, \boldsymbol{\mu}^k)$ be an iterate of the Lagrange-Newton method. We determine the next iterate $(\mathbf{x}^{k+1}, \boldsymbol{\mu}^{k+1}) = (\mathbf{x}^k, \boldsymbol{\mu}^k) + (\Delta\mathbf{x}, \Delta\boldsymbol{\mu})$ by solving the linear system of equations

$$\begin{pmatrix} \nabla^2_{\mathbf{xx}} L(\mathbf{x}^k, \boldsymbol{\mu}^k) & \nabla \mathbf{h}(\mathbf{x}^k) \\ \nabla \mathbf{h}(\mathbf{x}^k)^T & \mathbf{0} \end{pmatrix} \begin{pmatrix} \Delta\mathbf{x} \\ \Delta\boldsymbol{\mu} \end{pmatrix} = - \begin{pmatrix} \nabla_{\mathbf{x}} L(\mathbf{x}^k, \boldsymbol{\mu}^k) \\ \mathbf{h}(\mathbf{x}) \end{pmatrix}. \tag{2.7}$$

On the other hand, we consider the quadratic problem

$$\min_{\Delta\mathbf{x}\in\mathbb{R}^n} \nabla f(\mathbf{x}^k)^T\Delta\mathbf{x} + \tfrac{1}{2}\Delta\mathbf{x}^T\nabla^2_{\mathbf{xx}}L(\mathbf{x}^k,\boldsymbol{\mu}^k)\Delta\mathbf{x}^T \tag{2.8a}$$

$$\text{s.t. } 0 = \mathbf{h}_j(\mathbf{x}^k) + \nabla\mathbf{h}_j(\mathbf{x}^k)^T\Delta\mathbf{x}, \qquad j = 1,\dots,p. \tag{2.8b}$$

One can show that $(\Delta\mathbf{x},\Delta\boldsymbol{\mu})$ solves (2.7) if and only if $(\Delta\mathbf{x},\boldsymbol{\mu}^k+\Delta\boldsymbol{\mu})$ is a KKT point of Problem (2.8), cf. [144, Lemma 19.4]. Thus, simply put we can determine the iterates of the Lagrange-Newton method by solving a sequence of quadratic problems and in this way find a KKT point of Problem (2.6). An introduction to Quadratic Programming can be found in [110, ch. 16].

This idea can be transferred to equality *and* inequality constrained NLPs of Form (2.2). The result is a so-called local SQP method which – under appropriate assumptions – can be shown to be locally convergent with quadratic convergence rate, cf. [59, sec. 5.5.3]. Furthermore, the (local) method can be modified to achieve global convergence. For details – in particular on algorithmic realizations and convergence properties – we refer to [59, sec. 5.5] and [110, ch. 18].

Derivative-Free Optimization – Model-Based Approach

Interior-Point methods and SQP methods both rely on the availability of derivative information. In this paragraph however, we consider optimization problems for which reliable derivatives are *not* available in practice (at least at acceptable computational costs) or do not exist everywhere. This raises the need for Derivative-Free Optimization (DFO) methods. An introduction to DFO is given in [34] and, more concisely, in [110, ch. 9].

Various DFO methods exist. In this thesis, we make use of a so-called model-based DFO approach for box-constrained optimization problems of the form

$$\begin{aligned}\min_{\mathbf{x}\in\mathbb{R}^n}\ & f(\mathbf{x})\\ \text{s.t. }& \mathbf{a}_i \le \mathbf{x}_i \le \mathbf{b}_i \quad i = 1,\dots,n,\end{aligned}$$

see [124]. For an introduction to model-based DFO methods for unconstrained optimization problems, we refer to [110, sec. 9.2]. We give a short outline of a class of solution methods which includes the ones described in [124] and in [110, sec. 9.2]. Until satisfaction of termination conditions, in each iteration one proceeds as follows. The objective function $f(\cdot)$ is locally approximated by a linear or quadratic model

function $m^k(\cdot)$ which interpolates $f(\cdot)$ at a set of interpolation points comprising the current iterate $\mathbf{x}^k$. The model $m^k(\cdot)$ is used to determine a (feasible) iterate $\mathbf{x}^{k+1}$, e. g., by solving a trust region subproblem. In addition, the set of interpolation points is updated.

2.3 Optimal Control of Dynamic Systems

The present thesis deals with Optimal Control of dynamic systems, i. e., the optimization of dynamic processes which are described by means of dynamic systems that can be influenced or steered by so-called control functions. Comprehensive introductions to the topic of Optimal Control can be found, e. g., in [60] and [83], respectively. In this section, we introduce different types of dynamic systems, present a general problem formulation for OCPs, and discuss techniques for problem transformations.

2.3.1 Dynamic Systems

Let $\mathcal{T} = [t_0, t_f] \subset \mathbb{R}$ with $t_0 < t_f$. In this thesis, we consider dynamic systems which can be steered by a so-called control function and that are governed by Ordinary Differential Equations (ODEs) of first order. More precisely, the systems can be described by means of differential states $\mathbf{x} : \mathcal{T} \to \mathbb{R}^{n_x}$, and for a given control function $\mathbf{u} : \mathcal{T} \to \mathbb{R}^{n_u}$ the states $\mathbf{x}(\cdot)$ satisfy a differential equation of the form

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}(t)) \,, \quad t \in \mathcal{T},$$

with $\mathbf{f} : \mathcal{T} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$. If the initial state of a system is given we consider a so-called Initial Value Problem (IVP): for a given control function $\mathbf{u}(\cdot)$ and an initial value $\mathbf{x}_0$, find differential states $\mathbf{x}(\cdot)$ such that

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}(t)) \,, \quad t \in \mathcal{T}, \tag{2.9a}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \,. \tag{2.9b}$$

If $\mathbf{f}(\cdot)$ is continuous and globally Lipschitz continuous with respect to the second argument for the given $\mathbf{u}(\cdot)$, the solution $\mathbf{x}(\cdot)$ of the IVP is unique and depends continuously on the initial value $\mathbf{x}_0$. Furthermore, under additional requirements on the functional matrix $\frac{\partial \mathbf{f}}{\partial \mathbf{x}}$, for every $t \in \mathcal{T}$ the solution $\mathbf{x}(\cdot)$ of the IVP is continuously differentiable with respect to $\mathbf{x}_0$. For the mentioned results, see [141, sec. 7.1]. If we

replace the Initial Condition (2.9b) by a boundary condition of form

$$\mathbf{0} = \mathbf{r}\big(\mathbf{x}(t_0), \mathbf{x}(t_f)\big), \quad \text{or} \quad \mathbf{0} = \mathbf{r}\big(\mathbf{x}(t_0), \mathbf{x}(t_1), \ldots, \mathbf{x}(t_f)\big) \quad \text{with } t_0 < t_1 < \cdots < t_f,$$

we obtain a so-called boundary value problem or multi-point boundary value problem, respectively. Details on ODEs and their numerical treatment can be found, e. g., in [141, ch. 7].

More generally, we consider dynamic systems which are governed by Differential Algebraic Equations (DAEs). Here, the systems can be described by means of states $\mathbf{z} : \mathcal{T} \to \mathbb{R}^{n_z}$, and for a given control function $\mathbf{u} : \mathcal{T} \to \mathbb{R}^{n_u}$ the states $\mathbf{z}(\cdot)$ satisfy a differential equation of the form

$$\mathbf{F}(t, \mathbf{z}(t), \dot{\mathbf{z}}(t), \mathbf{u}(t)) = \mathbf{0}, \quad t \in \mathcal{T},$$

with $\mathbf{F} : \mathcal{T} \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u}$. Assuming that $\mathbf{u}(\cdot)$ is sufficiently smooth, we say that the above DAE is of differential index $k \in \mathbb{N}$ if k is the smallest number of differentiations with respect to time that is needed to transfer the equation system

$$\left(\frac{\mathrm{d}}{\mathrm{d}t}\right)^i \mathbf{F}(t, \mathbf{z}(t), \dot{\mathbf{z}}(t), \mathbf{u}(t)) = \mathbf{0}, \quad i = 0, \ldots, k$$

into an ODE system of form

$$\dot{\mathbf{z}}(t) = \tilde{\mathbf{f}}\left(t, \mathbf{z}(t), \mathbf{u}(t), \ldots, \left(\tfrac{\mathrm{d}}{\mathrm{d}t}\right)^k \mathbf{u}(t)\right)$$

by means of algebraic manipulations (cf. [60, p. 28] and [68, p. 455]). For instance, the DAEs we consider in this thesis arise from the dynamics of constrained Multi-Body Systems (see Section 4.1.1) and are of differential index 3. A comprehensive introduction to the numerical treatment of DAEs can be found, e. g., in [68, ch. VI & VII].

Switched Dynamic Systems

In this thesis, we consider switched (dynamic) systems. A concise introduction to switched systems can be found in [102, ch. 1] and for more details we refer to [63, 97, 145]. A switched system consists of a set of dynamic subsystems which represent the *operation modes* or simply *modes* the system can run in. A switch denotes a change of modes, and the timed sequence of modes is called switching sequence. A switch from one mode into another is triggered by a signal which can be caused

internally, i. e., by the value of the differential states or a time event, or externally, i. e., by a control function. Accordingly, an internally switched system is a switched system in which all switches are caused internally, and the term externally switched system is defined analogously.

In this thesis, we take a look at the ODE case in which the dimension of the differential states is constant throughout all modes. Hence, the modes can be identified with the set of possible right-hand side functions of the differential equation, and a switch can be seen as a change of the right-hand side. We are interested in systems which can run in a finite number of modes. For internally switched systems – also called implicitly switched systems – the differential equation can be described by means of a switching function $\boldsymbol{\sigma} : \mathcal{T} \times \mathbb{R}^{n_x} \to \mathbb{R}^{n_\sigma}$ in the form

$$\dot{\mathbf{x}}(t) = \mathbf{f}\big(t, \mathbf{x}(t), \mathbf{u}(t), \operatorname{sgn}(\boldsymbol{\sigma}(t, \mathbf{x}(t)))\big) .$$

Here, the current mode of the system depends on the sign of the components of $\boldsymbol{\sigma}(t, \mathbf{x}(t))$, and the switching sequence depends on the initial value $\mathbf{x}(t_0)$ as well as on the control function $\mathbf{u}(\cdot)$. On the other hand, for externally switched systems the current mode of the system can be encoded in the value a discrete valued control function $\mathbf{v} : \mathcal{T} \to \mathfrak{D}$ with $|\mathfrak{D}| < \infty$ (e. g., $\mathfrak{D} \subset \mathbb{N}$ finite subset), and the differential equation can be expressed in the form

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}(t), \mathbf{v}(t)) .$$

Here, the overall control function comprises a continuously valued part $\mathbf{u}(\cdot)$ and a discrete valued part $\mathbf{v}(\cdot)$ where the latter determines the switching sequence.

In certain applications, a switch can induce a discontinuity of the differential states $\mathbf{x}(\cdot)$ which we call a jump in the following. At switching time t_s, the transition from the states before the jump, $\mathbf{x}(t_s^-)$, to the states after jump, $\mathbf{x}(t_s^+)$, can be expressed by means of a jump function $\boldsymbol{\Delta}(\cdot)$. Depending on the particular application, the jump function depends on the mode before the switch, after the switch, or both. For externally switched systems, the jump condition can be stated in the form

$$\mathbf{x}(t_s^+) = \boldsymbol{\Delta}\left(t_s, \mathbf{x}(t_s^-), \mathbf{v}(t_s^-), \mathbf{v}(t_s^+)\right) ,$$

where $\mathbf{v}(t_s^-)$ and $\mathbf{v}(t_s^+)$ encode the mode of the system before and after the switch, respectively.

2.3.2 Problem Formulation

We present a general problem formulation for OCPs. In words, we seek for so-called controls that steer a given dynamic process from an initial state to a terminal state in an optimal manner with respect to a performance criterion while satisfying constraints which are imposed on the process. More precisely, we consider optimization problems, e. g. (and w. l. o. g.) of a standardized form

$$\min_{\mathfrak{u},\mathbf{u}(\cdot),\mathbf{x}(\cdot)} \Phi^M(\mathbf{x}(1)) + \int_0^1 \Phi^L(\mathbf{x}(t),\mathbf{u}(t))\,\mathrm{d}t \tag{2.10a}$$

$$\text{s.t.} \quad \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t),\mathbf{u}(t)), \qquad t \in [0,1], \tag{2.10b}$$

$$\mathbf{0} \le \mathbf{c}(\mathbf{x}(t),\mathbf{u}(t)), \qquad t \in [0,1], \tag{2.10c}$$

$$\mathbf{0} \le \mathbf{r}(\mathbf{x}(0),\mathbf{x}(1),\mathfrak{u},\mathbf{p}), \tag{2.10d}$$

$$\mathfrak{u} \in \mathcal{P} \subseteq \mathbb{R}^{n_\mathfrak{u}}, \tag{2.10e}$$

$$\mathbf{u}(t) \in \mathcal{U} \subseteq \mathbb{R}^{n_u}, \qquad t \in [0,1], \tag{2.10f}$$

with

- so-called controllable parameters $\mathfrak{u} \in \mathbb{R}^{n_\mathfrak{u}}$,
- control function $\mathbf{u} : [0,1] \to \mathbb{R}^{n_u}$,
- differential states $\mathbf{x} : [0,1] \to \mathbb{R}^{n_x}$,
- (non-controllable) parameters $\mathbf{p} \in \mathbb{R}^{n_p}$,
- model functions $\Phi^M : \mathbb{R}^{n_x} \to \mathbb{R}$, $\Phi^L : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$, $\mathbf{f} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$, $\mathbf{c} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_c}$, and $\mathbf{r} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_\mathfrak{u}} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_r}$,

where all inequalities are assumed to hold component-wise. Remark, that in contrast to the non-controllable parameters $\mathbf{p}$, the controllable parameters $\mathfrak{u}$ are subject to optimization. Optimization problems of this kind are called Optimal Control Problems (OCPs). The above problem formulation covers many situations, such as externally switched systems, free time horizons, so-called multi-stage problems with possible jumps in the differential states (cf. Problem (2.14)), etc., as we will we see in the subsequent paragraphs. The terms $\Phi^M(\mathbf{x}(1))$ and $\int_0^1 \Phi^L(\mathbf{x}(t),\mathbf{u}(t))\,\mathrm{d}t$ in the Objective Function (2.10a) are called Mayer term and Lagrange term, respectively. Furthermore, we refer to the Constraints (2.10c) as path constraints and to (2.10d) as boundary constraints. Note that the above problem formulation formally includes

equality constraints as well, as they can be expressed by means of two opposing inequality constraints. We consider Problem (2.10) as an optimization problem in the Banach space

$$\mathbb{R}^{n_{\mathbf{u}}} \times L^{\infty}\left([0,1], \mathbb{R}^{n_u}\right) \times W^{1,\infty}\left([0,1], \mathbb{R}^{n_x}\right) \ni (\mathbf{u}, \mathbf{u}(\cdot), \mathbf{x}(\cdot)) \,,$$

see Section 2.1. Thus, Problem (2.10) is an infinite dimensional optimization problem of Form (2.1).

Mixed-Integer Optimal Control Problems

In Problem Formulation (2.10) we do not make assumptions on the sets $\mathcal{P} \ni \mathbf{u}$ and $\mathcal{U} \ni \mathbf{u}(t)$. If one of the sets is discrete, we speak of a Mixed-Integer Optimal Control Problem (MIOCP) – an OCP which involves continuously valued and discrete valued control variables. MIOCPs arise, e. g., in connection with switched systems in which the switching sequence is free and subject to optimization by encoding the switching sequence in the time course of the control function (see, e. g., Chapter 5). Due to the discrete valued optimization variables, MIOCPs demand for a different treatment than continuous OCPs. As entry points to the topic we refer to [60, ch. 7], [81, ch. 2], and [127].

2.3.3 Transformation Techniques

OCPs can be formulated in many different ways. In the following, we present techniques to transfer frequently arising problem formulations to Form (2.10). Though this is important for theoretical considerations, in practice it is often beneficial to use the original problem formulation together with tailored solution methods.

Maximization. As the maximization of a real valued function $\Phi(\cdot)$ is equivalent to the minimization of $-\Phi(\cdot)$, OCPs in which the objective function is maximized can be equivalently transferred to a minimization problem by changing the sign of the cost function.

Time Horizon (see [60, sec. 1.2.1]). In Problem (2.10), we consider the fixed normalized time horizon $[0,1]$. However, we are also interested in similar problems with a time horizon $[t_0, t_f]$ where possibly both t_0 and t_f can either be fixed or free, respectively. In this situation, we employ a linear time transformation

$$t : [0,1] \to [t_0, t_f], \quad \tau \mapsto t_0 + \tau(t_f - t_0)$$

and define time transformed differential states $\tilde{\mathbf{x}} : [0,1] \to \mathbb{R}^{n_x}$ and control functions $\tilde{\mathbf{u}} : [0,1] \to \mathbb{R}^{n_u}$ by $\tilde{\mathbf{x}}(\tau) = \mathbf{x}(t(\tau))$ and $\tilde{\mathbf{u}}(\tau) = \mathbf{u}(t(\tau))$, respectively. According to the chain rule, we obtain for the differential equation

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\tilde{\mathbf{x}}(\tau) = \frac{\mathrm{d}}{\mathrm{d}\tau}\mathbf{x}(t(\tau)) = \dot{\mathbf{x}}(t(\tau))\frac{\mathrm{d}}{\mathrm{d}\tau}t(\tau) = (t_f - t_0)\mathbf{f}(\tilde{\mathbf{x}}(\tau), \tilde{\mathbf{u}}(\tau))\,. \tag{2.11}$$

Path constraints as well as boundary constraints can be directly expressed in the Form (2.10c) and (2.10d), respectively, by means of $\tilde{\mathbf{x}}(\cdot)$ and $\tilde{\mathbf{u}}(\cdot)$, and the objective function is transformed to

$$\Phi^M(\tilde{\mathbf{x}}(1)) + \int_0^1 (t_f - t_0)\Phi^L(\tilde{\mathbf{x}}(\tau), \tilde{\mathbf{u}}(\tau))\,\mathrm{d}\tau\,.$$

If t_0 or t_f are free variables, we add them to the vector of controllable parameters. Altogether, we obtain a problem of Form (2.10).

Parameter-Dependence of Model Functions. In Problem (2.10), the parameters $\mathbf{u}$ and $\mathbf{p}$ do not enter $\Phi^M(\cdot)$, $\Phi^L(\cdot)$, $\mathbf{f}(\cdot)$, $\mathbf{c}(\cdot)$, and solely appear in the Boundary Constraints (2.10d). In the following, we focus on $\mathbf{p}$ since $\mathbf{u}$ can be treated similarly. If we consider a more general OCP with the parameters $\mathbf{p}$ entering the objective function, $\mathbf{f}(\cdot)$, or $\mathbf{c}(\cdot)$, we introduce additional constant differential states $\tilde{\mathbf{p}}(\cdot)$ with the initial value $\mathbf{p}$. The augmented differential equation is given by

$$\begin{pmatrix}\dot{\mathbf{x}}(t)\\ \dot{\tilde{\mathbf{p}}}(t)\end{pmatrix} = \begin{pmatrix}\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p})\\ \mathbf{0}\end{pmatrix} = \begin{pmatrix}\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), \tilde{\mathbf{p}}(t))\\ \mathbf{0}\end{pmatrix} \tag{2.12}$$

and can be stated in the form $\dot{\mathbf{y}} = \tilde{\mathbf{f}}(\mathbf{y}(t), \mathbf{u}(t))$ with the augmented states $\mathbf{y}(\cdot) = \begin{pmatrix}\mathbf{x}(\cdot)\\ \tilde{\mathbf{p}}(\cdot)\end{pmatrix}$. Path constraints and the objective function can be expressed in terms of $\mathbf{y}(\cdot)$ and $\mathbf{u}(\cdot)$. Furthermore, the boundary constraints are replaced by suitable constraints of the form

$$\mathbf{0} \leq \tilde{\mathbf{r}}(\mathbf{y}(0), \mathbf{y}(1), \mathbf{u}, \mathbf{p}) \tag{2.13}$$

which include the condition $\tilde{\mathbf{p}}(0) = \mathbf{p}$. Altogether, we get a problem of Form (2.10).

Non-Autonomous Problems (see [60, sec. 1.2.2]). In Problem (2.10), the model functions are not explicitly time-dependent which renders the problem a so-called autonomous problem. If one of the functions $\Phi^L(\cdot)$, $\mathbf{f}(\cdot)$, $\mathbf{c}(\cdot)$, $\mathbf{r}(\cdot)$ does explicitly depend on t, we introduce an additional differential state $\tilde{t} : [0,1] \to [0,1]$ with $\frac{\mathrm{d}}{\mathrm{d}t}\tilde{t}(t) = 1$ and

$\tilde{t}(0) = 0$. We augment the differential equation and the boundary constraints as in (2.12) and (2.13) and replace all explicit time dependencies by $\tilde{t}(t)$ to obtain a problem of Form (2.10).

Transformations of Objective Functions. In Problem (2.10), the Mayer term solely depends on $\mathbf{x}(1)$. If $\Phi^M(\cdot)$ depends on $\mathbf{x}(0)$ as well, we introduce an additional constant differential state with initial value $\mathbf{x}(0)$ which enables us to formulate the Problem in Form (2.10) again. Furthermore, we can eliminate the Lagrange term and focus on Mayer type objective functions by introducing an additional differential state $l(\cdot)$ with $\dot{l}(t) = \Phi^L(\mathbf{x}(t), \mathbf{u}(t))$ and $l(0) = 0$. The objective function is then given by the term $\Phi^M(\mathbf{x}(1)) + l(1)$ which can be brought in form $\tilde{\Phi}^M(\tilde{\mathbf{x}}(1))$ by introducing an additional differential state encoding the time, see the previous paragraph.

Multi-Stage OCPs (compare [60, sec. 1.2.5]). In this thesis, we deal with problems of the form

$$\min_{\substack{\mathbf{x}^1(\cdot),\dots,\mathbf{x}^n(\cdot),\\ \mathbf{u}^1(\cdot),\dots,\mathbf{u}^n(\cdot),\\ T_1,\dots,T_n}} \quad \sum_{j=1}^{n} \left[\Phi_j^M\left(T_j, \mathbf{x}^j(T_j), \mathbf{p}\right) + \int_{T_{j-1}}^{T_j} \Phi_j^L\left(\mathbf{x}^j(t), \mathbf{u}^j(t), \mathbf{p}\right) \mathrm{d}t \right] \tag{2.14a}$$

$$\text{s.t.} \quad \dot{\mathbf{x}}^j(t) = \mathbf{f}^j\left(\mathbf{x}^j(t), \mathbf{u}^j(t), \mathbf{p}\right), \quad t \in \mathcal{T}_j, \qquad j = 1,\dots,n, \tag{2.14b}$$

$$T_{j-1} \le T_j, \qquad j = 1,\dots,n, \tag{2.14c}$$

$$\mathbf{x}^{j+1}(T_j) = \mathbf{\Delta}^j\left(\mathbf{x}^j(T_j), \mathbf{p}\right), \qquad j = 1,\dots,n, \tag{2.14d}$$

$$\mathbf{0} \le \mathbf{c}^j\left(\mathbf{x}^j(t), \mathbf{u}^j(t), \mathbf{p}\right), \quad t \in \mathcal{T}_j, \tag{2.14e}$$

$$\mathbf{0} \le \mathbf{r}\left(\mathbf{x}^1(T_0), \mathbf{x}^1(T_1), \dots, \mathbf{x}^n(T_n), \mathbf{p}\right), \tag{2.14f}$$

with fixed $T_0 \in \mathbb{R}$ and $\mathcal{T}_j = [T_{j-1}, T_j]$. Problems of this kind are called multi-stage OCPs. They include $j = 1,\dots,n$ consecutive so-called model stages, also denoted by the term phases. Each model stage is assigned a time horizon $\mathcal{T}_j$, optimization variables $\mathbf{x}^j(\cdot) : \mathcal{T}_j \to \mathbb{R}^{n_{x,j}}$, $\mathbf{u}^j(\cdot) : \mathcal{T}_j \to \mathbb{R}^{n_{u,j}}$, $T_j \in \mathbb{R}$, a set of constraints and an objective function contribution. Furthermore, the Constraints (2.14d) describe the transition of the values of the differential states $\mathbf{x}^j(\cdot)$ at phase transition. Multi-stage OCPs arise in connection with switched dynamic systems with predefined sequences of modes and will be used for gait modeling in this thesis, see Section 4.1.2. If $n = 1$ and $\mathbf{\Delta}^1(\cdot) = \mathbf{Id}(\cdot)$, we obtain a so-called single-stage Problem.

We explain how Problem (2.14) can be reformulated into an equivalent problem of Form (2.10) by means of the techniques described in the previous paragraphs. To

this end, we introduce time transformations

$$t^j : [0,1] \to \mathcal{T}_j, \quad \tau \mapsto T_{j-1} + (T_j - T_{j-1})\tau, \quad j = 1, \dots, n,$$

as well as differential states $\tilde{\mathbf{x}}^j : [0,1] \to \mathbb{R}^{n_{x,j}}$ and control functions $\tilde{\mathbf{u}}^j : [0,1] \to \mathbb{R}^{n_{u,j}}$ which are given by $\tilde{\mathbf{x}}^j(\tau) = \mathbf{x}\big(t^j(\tau)\big)$ and $\tilde{\mathbf{u}}^j(\tau) = \mathbf{u}\big(t^j(\tau)\big)$, respectively. Similar to (2.11), we obtain the differential equations

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\tilde{\mathbf{x}}^j(\tau) = (T_j - T_{j-1})\mathbf{f}\big(\tilde{\mathbf{x}}^j(\tau), \tilde{\mathbf{u}}^j(\tau), \mathbf{p}\big).$$

We set

$$\mathfrak{u} = \begin{pmatrix} T_1 \\ \vdots \\ T_n \end{pmatrix}, \ \tilde{\mathbf{x}}(\tau) = \begin{pmatrix} \tilde{\mathbf{x}}^1(\tau) \\ \vdots \\ \tilde{\mathbf{x}}^n(\tau) \end{pmatrix}, \ \tilde{\mathbf{u}}(\tau) = \begin{pmatrix} \tilde{\mathbf{u}}^1(\tau) \\ \vdots \\ \tilde{\mathbf{u}}^n(\tau) \end{pmatrix}, \text{ and } \tilde{\mathbf{c}}(\tilde{\mathbf{x}}(\tau), \tilde{\mathbf{u}}(\tau), \mathbf{p}) = \begin{pmatrix} \mathbf{c}^1(\tilde{\mathbf{x}}^1(\tau), \tilde{\mathbf{u}}^1(\tau), \mathbf{p}) \\ \vdots \\ \mathbf{c}^n(\tilde{\mathbf{x}}^n(\tau), \tilde{\mathbf{u}}^n(\tau), \mathbf{p}) \end{pmatrix}.$$

The Conditions (2.14c), (2.14d), and (2.14f) can be expressed together equivalently by a boundary constraint of the form $\mathbf{0} \le \tilde{\mathbf{r}}\big(\tilde{\mathbf{x}}(0), \tilde{\mathbf{x}}(1), \mathfrak{u}, \mathbf{p}\big)$. Thus, Problem (2.14) can be transformed into a problem of the form

$$\min_{\mathfrak{u}, \tilde{\mathbf{u}}(\cdot), \tilde{\mathbf{x}}(\cdot)} \ \tilde{\Phi}^M\big(\tilde{\mathbf{x}}(1), \mathfrak{u}, \mathbf{p}\big) + \int_0^1 \tilde{\Phi}^L\big(\tilde{\mathbf{x}}(\tau), \tilde{\mathbf{u}}(\tau), \mathfrak{u}, \mathbf{p}\big)\,\mathrm{d}\tau \tag{2.15a}$$

$$\text{s.t. } \dot{\tilde{\mathbf{x}}}(\tau) = \tilde{\mathbf{f}}\big(\tilde{\mathbf{x}}(\tau), \tilde{\mathbf{u}}(t), \mathfrak{u}, \mathbf{p}\big), \qquad \tau \in [0,1], \tag{2.15b}$$

$$\mathbf{0} \le \tilde{\mathbf{c}}\big(\tilde{\mathbf{x}}(\tau), \tilde{\mathbf{u}}(\tau), \mathbf{p}\big), \qquad \tau \in [0,1], \tag{2.15c}$$

$$\mathbf{0} \le \tilde{\mathbf{r}}\big(\tilde{\mathbf{x}}(0), \tilde{\mathbf{x}}(1), \mathfrak{u}, \mathbf{p}\big). \tag{2.15d}$$

Finally, we use the technique described in paragraph “Parameter-Dependence of Model Functions” (see Page 23) to transfer all parameter dependencies to the boundary constraints which yields a problem of Form (2.10), as desired.

2.4 Direct Solution Approaches to Optimal Control Problems

We present solution approaches to continuous OCPs, i. e., OCPs with continuously valued optimization variables. For solution approaches to MIOCPs we refer to [60, ch. 7], [81, ch. 2], and [127]. As stated above, OCPs are infinite dimensional optimization problems. In this section, we concentrate on so-called *direct approaches*. Here, an OCP is transcribed into an NLP in a first step, and the resulting finite dimensional optimization problem is solved subsequently. For this reason, direct approaches are also referred to as “first discretize, then optimize” approaches. In contrast, in *indirect*

approaches one establishes necessary conditions, also called maximum or minimum principles, for continuous OCPs. This results in boundary value problems which have to be solved numerically in a second step. Therefore, indirect approaches are also known as "first optimize, then discretize" approaches. A further prominent solution approach for OCPs is *Dynamic Programming*. For indirect approaches we refer to [60] and [83, part III], and for Dynamic Programming to [83, ch. 3].

In this thesis we make use of two direct approaches, namely Direct Multiple Shooting and Direct Collocation, both of which we introduce in the following. To this end, we consider a continuous single-stage OCP of the form

$$\min_{\mathbf{u},\mathbf{u}(\cdot),\mathbf{x}(\cdot)} \Phi^M(\mathbf{x}(1)) \tag{2.16a}$$

$$\text{s.t.} \quad \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t),\mathbf{u}(t)), \qquad t \in [0,1], \tag{2.16b}$$

$$\mathbf{0} \leq \mathbf{c}(\mathbf{x}(t),\mathbf{u}(t)), \qquad t \in [0,1], \tag{2.16c}$$

$$\mathbf{0} \leq \mathbf{r}(\mathbf{x}(0),\mathbf{x}(1),\mathbf{u},\mathbf{p}), \tag{2.16d}$$

with controllable parameters $\mathbf{u} \in \mathbb{R}^{n_\mathbf{u}}$, control function $\mathbf{u}(\cdot) \in L^\infty([0,1],\mathbb{R}^{n_u})$, and differential states $\mathbf{x}(\cdot) \in W^{1,\infty}([0,1],\mathbb{R}^{n_x})$. We assume all model functions to be continuously differentiable.

2.4.1 Direct Multiple Shooting

The Direct Multiple Shooting Approach for solving OCPs was introduced in [25, 120] for OCPs constrained by ODEs. The software package `MUSCOD-II` [95] provides an implementation. In the following, we describe a so-called Multiple Shooting discretization for an OCP of Form (2.16) which transcribes the problem into an NLP. The resulting NLP is then solved, e. g., with a structure exploiting SQP method. For the discretization of multi-stage problems we refer to [93].

As a first step, we introduce a shooting grid of size $N+1$:

$$0 = \tau_0 < \tau_1 < \cdots < \tau_N = 1$$

Control Function Discretization

For each shooting interval $[\tau_i,\tau_{i+1}]$, $i = 0,\ldots,N-1$, the control function $\mathbf{u}(\cdot)$ is approximated by elements $\mathbf{U}^i(\cdot)$ of finite dimensional subspaces of $L^\infty([\tau_i,\tau_{i+1}],\mathbb{R}^{n_u})$. Popular choices for the interval-wise discretization of the components $\mathbf{u}_j(\cdot)$,

$j = 1, \dots, n_u$, of $\mathbf{u}(\cdot)$ are piecewise constant functions,

$$\mathbf{U}_j^i(t) = q^{i,j} \quad \text{for } t \in [\tau_i, \tau_{i+1}]$$

with $q^{i,j} \in \mathbb{R}$, and piecewise linear functions,

$$\mathbf{U}_j^i(t) = \frac{t-\tau_{i+1}}{\tau_i - \tau_{i+1}} \mathbf{q}_1^{i,j} + \frac{t-\tau_i}{\tau_{i+1}-\tau_i} \mathbf{q}_2^{i,j} \quad \text{for } t \in [\tau_i, \tau_{i+1}]$$

with $\mathbf{q}^{i,j} \in \mathbb{R}^2$. In any case, for each shooting interval $[\tau_i, \tau_{i+1}]$ and each component $\mathbf{U}_j^i(\cdot)$, $j = 1, \dots, n_u$, of $\mathbf{U}^i(\cdot)$ there are $\mathbf{q}^{i,j} \in \mathbb{R}^{n_{i,j}}$ and functions $\boldsymbol{\xi}_j^i : [\tau_i, \tau_{i+1}] \times \mathbb{R}^{n_{i,j}} \to \mathbb{R}$ such that

$$\mathbf{U}_j^i(t) = \boldsymbol{\xi}_j^i\left(t, \mathbf{q}^{i,j}\right) \quad \text{for } t \in [\tau_i, \tau_{i+1}].$$

Setting

$$\mathbf{q}^i = \begin{pmatrix} \mathbf{q}^{i,1} \\ \vdots \\ \mathbf{q}^{i,n_u} \end{pmatrix} \in \mathbb{R}^{\sum_j n_{i,j}} \quad \text{and} \quad \mathbf{q} = \begin{pmatrix} \mathbf{q}^0 \\ \vdots \\ \mathbf{q}^{N-1} \end{pmatrix} \in \mathbb{R}^{\sum_{i,j} n_{i,j}},$$

we obtain an approximation of $\mathbf{u}(\cdot)$ by the parameterized function

$$\mathbf{U}(t, \mathbf{q}) = \begin{cases} \mathbf{U}^i\left(t, \mathbf{q}^i\right) & \text{if } t \in [\tau_i, \tau_{i+1}) \text{ for } i = 0, \dots, N-2, \\ \mathbf{U}^{N-1}\left(t, \mathbf{q}^{N-1}\right) & \text{if } t \in [\tau_{N-1}, \tau_N]. \end{cases}$$

If required, one can enforce continuity of a component $\mathbf{U}_j(\cdot)$ by imposing additional constraints of the form

$$\mathbf{U}_j^i\left(\tau_{i+1}, \mathbf{q}^{i,j}\right) = \mathbf{U}_j^{i+1}\left(\tau_{i+1}, \mathbf{q}^{i+1,j}\right), \quad i = 0, \dots, N-2.$$

State Parametrization

For the differential states $\mathbf{x}(\cdot)$, we introduce variables $\mathbf{s}^i \in \mathbb{R}^{n_x}$, $i = 0, \dots, N$, which represent the values of $\mathbf{x}(\cdot)$ at the shooting grid points and set

$$\mathbf{s} = \begin{pmatrix} \mathbf{s}^0 \\ \vdots \\ \mathbf{s}^N \end{pmatrix}.$$

For each shooting interval we consider an IVP

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{f}\Big(\mathbf{x}(t), \mathbf{U}^i\big(t, \mathbf{q}^i\big)\Big), \quad t \in [\tau_i, \tau_{i+1}], \\ \mathbf{x}(\tau_i) &= \mathbf{s}^i, \end{aligned} \tag{2.17}$$

and assume that for given initial values $\mathbf{s}^i$ and parameters $\mathbf{q}^i$ the solution of the IVP – which we denote by $\mathbf{x}\big(t; \mathbf{s}^i, \mathbf{q}^i\big)$ – exists, is unique, and continuously differentiable with respect to $\mathbf{s}^i$ and $\mathbf{q}^i$ for all $t \in [\tau_i, \tau_{i+1}]$, respectively. In practice, we solve the IVPs by numerical integration. In order to get a continuous trajectory we impose additional constraints, the so-called matching conditions

$$\mathbf{0} = \mathbf{x}\big(\tau_{i+1}; \mathbf{s}^i, \mathbf{q}^i\big) - \mathbf{s}^{i+1} \quad \text{for } i = 0, \dots, N-1.$$

In particular, we obtain $\mathbf{x}\big(1; \mathbf{s}^{N-1}, \mathbf{q}^{N-1}\big) = \mathbf{s}^N$.

Discretization of Objective Function, Path Constraints and Boundary Constraints

The objective function for the discretized problem is given by

$$\Phi^M\big(\mathbf{s}^N\big)$$

and the Boundary Constraints (2.16d) are transformed into

$$\mathbf{0} \leq \mathbf{r}\big(\mathbf{s}^0, \mathbf{s}^N, \mathbf{u}, \mathbf{p}\big).$$

The Path Constraints (2.16c) are enforced to hold at the shooting grid points which yields

$$\begin{aligned} \mathbf{0} &\leq \mathbf{c}\Big(\mathbf{s}^i, \mathbf{U}^i\big(\tau_i, \mathbf{q}^i\big)\Big), \quad i = 0, \dots, N-1, \\ \mathbf{0} &\leq \mathbf{c}\big(\mathbf{s}^N, \mathbf{U}^{N-1}\big(\tau_N, \mathbf{q}^{N-1}\big)\big). \end{aligned} \tag{2.18}$$

Though this is a relaxation of (2.16c), in many real world application it is sufficient to demand (2.18). However, if critical violations of the Constraints (2.16c) are observed, a straightforward approach is to adapt or refine the shooting grid. A more sophisticated method to overcome the issue can be found in [122, 123].

The Resulting Nonlinear Programming Problem

The NLP resulting from the control discretization and the state parametrization is given by

$$\min_{\mathbf{u},\mathbf{s},\mathbf{q}} \Phi^M\left(\mathbf{s}^N\right) \tag{2.19a}$$

$$\text{s.t.} \quad \mathbf{0} = \mathbf{x}\left(\tau_{i+1};\mathbf{s}^i,\mathbf{q}^i\right) - \mathbf{s}^{i+1}, \qquad i = 0,\dots,N-1, \tag{2.19b}$$

$$\mathbf{0} \le \mathbf{c}\left(\mathbf{s}^i,\mathbf{U}^i\left(\tau_i,\mathbf{q}^i\right)\right), \qquad i = 0,\dots,N-1, \tag{2.19c}$$

$$\mathbf{0} \le \mathbf{c}\left(\mathbf{s}^N,\mathbf{U}^{N-1}\left(\tau_N,\mathbf{q}^{N-1}\right)\right), \tag{2.19d}$$

$$\mathbf{0} \le \mathbf{r}\left(\mathbf{s}^0,\mathbf{s}^N,\mathbf{u},\mathbf{p}\right). \tag{2.19e}$$

This is an NLP of Form (2.2) which can be solved with tailored solution methods that exploit the specific problem structure which results from the Multiple Shooting discretization, cf., e. g., [25, 94, 93]. Since the control function is discretized while the differential states are determined by solving the resulting dynamics equation, we speak of a reduced discretization approach.

2.4.2 Direct Collocation

In this section, we present the Direct Collocation Approach for solving OCPs [13, 20, 70]. In contrast to the Multiple Shooting approach, Direct Collocation is a full discretization approach in which both the control function $\mathbf{u}(\cdot)$ and the differential states $\mathbf{x}(\cdot)$ are discretized using polynomial approximations. The discretization yields an NLP which is solved in a second step. The software package `grc` [102] provides an implementation of such an approach. In the following, we describe a general collocation discretization scheme for an OCP of Form (2.16).

State and Control Discretization

To increase readability, we focus on global discretization schemes in which each component of the differential states and the control function is approximated by one polynomial on the entire time horizon. Furthermore, we assume the polynomial degree for the differential states and the controls to be constant throughout all components, respectively. More precisely, we approximate $\mathbf{x}(\cdot)$ by a function

$$\mathbf{X}(t) = \sum_{k=0}^{N_X} \mathbf{a}^k \phi_k(t)$$

with parameters $\mathbf{a}^k \in \mathbb{R}^{n_x}$ and basis polynomials $\phi_k : [0,1] \to \mathbb{R}$, $k = 0,\dots,N_X$, in such a way that the components of $\mathbf{X}(\cdot)$ are polynomials of degree N_X. Similarly, $\mathbf{u}(\cdot)$ is approximated by

$$\mathbf{U}(t) = \sum_{k=0}^{N_U} \mathbf{b}^k \psi_k(t)$$

with parameters $\mathbf{b}^k \in \mathbb{R}^{n_u}$ and basis polynomials $\psi_k : [0,1] \to \mathbb{R}$, $k = 0,\dots,N_U$, such that the components $\mathbf{U}_j(t)$ are polynomials of degree N_U. In contrast to global discretization schemes, in local schemes we set up a grid $0 = \tau_0 < \tau_1 < \cdots < \tau_N = 1$ and approximate the components of the states and controls in the intervals $[\tau_i, \tau_{i+1}]$ by polynomials with possibly varying degree per interval.

Discretization of Objective Function and Constraints

We choose a set $\{t_l^c\}_{l=1}^{N_X} \subset [0,1]$ of cardinality N_X of so-called *collocation points* and demand the Differential Equation (2.16b) to hold at these points for the discretized states and controls,

$$\dot{\mathbf{X}}(t_l^c) = \mathbf{f}\big(\mathbf{X}(t_l^c), \mathbf{U}(t_l^c)\big) \quad \Longleftrightarrow \quad \sum_{k=0}^{N_X} \mathbf{a}^k \dot{\phi}_k(t_l^c) = \mathbf{f}\Big(\textstyle\sum_{k=0}^{N_X} \mathbf{a}^k \phi_k(t_l^c), \sum_{k=0}^{N_U} \mathbf{b}^k \psi_k(t_l^c)\Big)$$

for $l = 1,\dots,N_X$, such that the coefficients $\mathbf{a}^k$ are determined by given $\mathbf{b}^k$ and the value of $\mathbf{X}(\cdot)$ at the initial time $t = 0$.

For the Path Constraints (2.16c) we consider a set $\{t_l^e\}_{l=1}^{N_e} \subset [0,1]$ of evaluation points and demand

$$\mathbf{0} \le \mathbf{c}\big(\mathbf{X}(t_l^e), \mathbf{U}(t_l^e)\big) = \mathbf{c}\Big(\textstyle\sum_{k=0}^{N_X} \mathbf{a}^k \phi_k(t_l^e), \sum_{k=0}^{N_U} \mathbf{b}^k \psi_k(t_l^e)\Big), \quad l = 1,\dots,N_e.$$

The Boundary Constraints (2.16d) are transformed into

$$\mathbf{0} \le \mathbf{r}\big(\mathbf{X}(0), \mathbf{X}(1), \mathbf{u}, \mathbf{p}\big) = \mathbf{r}\Big(\textstyle\sum_{k=0}^{N_X} \mathbf{a}^k \phi_k(0), \sum_{k=0}^{N_X} \mathbf{a}^k \phi_k(1), \mathbf{u}, \mathbf{p}\Big),$$

and the objective function of the discretized problem is given by

$$\Phi^M\big(\mathbf{X}(1)\big) = \Phi^M\Big(\textstyle\sum_{k=0}^{N_X} \mathbf{a}^k \phi_k(1)\Big).$$

The Resulting Nonlinear Programming Problem

We set

$$\mathbf{a} = \begin{pmatrix} \mathbf{a}^0 \\ \vdots \\ \mathbf{a}^{N_X} \end{pmatrix} \in \mathbb{R}^{(N_X+1)n_x} \quad \text{and} \quad \mathbf{b} = \begin{pmatrix} \mathbf{b}^0 \\ \vdots \\ \mathbf{b}^{N_U} \end{pmatrix} \in \mathbb{R}^{(N_U+1)n_u}.$$

With the previously described discretization scheme, the original OCP transforms into

$$\min_{\mathbf{u},\mathbf{a},\mathbf{b}} \ \Phi^M\left(\textstyle\sum_{k=0}^{N_X} \mathbf{a}^k \phi_k(1)\right) \tag{2.20a}$$

$$\text{s.t. } \mathbf{0} = \textstyle\sum_{k=0}^{N_X} \mathbf{a}^k \dot{\phi}_k(t_l^c) - \mathbf{f}\left(\sum_{k=0}^{N_X} \mathbf{a}^k \phi_k(t_l^c), \sum_{k=0}^{N_U} \mathbf{b}^k \psi_k(t_l^c)\right), \quad l = 1,\dots,N_x, \tag{2.20b}$$

$$\mathbf{0} \le \mathbf{c}\left(\textstyle\sum_{k=0}^{N_X} \mathbf{a}^k \phi_k(t_l^e), \sum_{k=0}^{N_U} \mathbf{b}^k \psi_k(t_l^e)\right), \qquad l = 1,\dots,N_e, \tag{2.20c}$$

$$\mathbf{0} \le \mathbf{r}\left(\textstyle\sum_{k=0}^{N_X} \mathbf{a}^k \phi_k(0), \sum_{k=0}^{N_X} \mathbf{a}^k \phi_k(1), \mathbf{u}, \mathbf{p}\right). \tag{2.20d}$$

This is an NLP of Form (2.2) which can be solved with tailored solution methods, depending on the specific structure of the problem.

2.5 Derivative Generation

In the course of this thesis, we apply direct methods to OCPs and solve the arising discretized problems with SQP or Interior-Point methods, see Section 2.2.2. To this end, we need to efficiently provide (directional) derivatives of all functions which occur in the discretized problems. In the following, we state different strategies for the computation of partial derivatives of first order. This includes so-called sensitivities which arise in Direct Multiple Shooting. These considerations can be transferred to directional derivatives.

2.5.1 Three Approaches for Calculating Derivatives

For the numerical computation of derivatives of functions $\mathbf{f}: \mathbb{R}^n \to \mathbb{R}^m$ we consider three approaches which are of interest in this thesis. We follow the introductions in [3, ch. 2] and [110, ch. 8] where the interested reader can find details and further references, respectively.

Symbolic Differentiation

Symbolic Differentiation, also called analytical differentiation, can be used if an explicit formula of the considered function is available. An explicit expression for the

derivatives is derived. This process can be done by hand which is, however, cumbersome and prone to error for complex functions. An alternative is the use of software tools such as MATLAB® [100]. As it provides explicit formulas for derivatives, symbolic differentiation enables us to compute derivative values exact up to machine precision.

Finite Differences

The finite differences approach allows to treat the function $\mathbf{f}(\cdot)$ as a black box. The partial derivatives of the components of $\mathbf{f}(\cdot)$ are approximated, e. g., by one-sided difference quotients

$$\frac{\mathbf{f}_j(\mathbf{x}+\varepsilon\mathbf{e}_i)-\mathbf{f}_j(\mathbf{x})}{\varepsilon} \approx \frac{\partial \mathbf{f}_j}{\partial \mathbf{x}_i}(\mathbf{x}) \tag{2.21}$$

with $\varepsilon > 0$ and $\mathbf{e}_i$ being the i-th unit vector. Another variant is to use central difference approximations of the form

$$\frac{\mathbf{f}_j(\mathbf{x}+\varepsilon\mathbf{e}_i)-\mathbf{f}_j(\mathbf{x}-\varepsilon\mathbf{e}_i)}{2\varepsilon} \approx \frac{\partial \mathbf{f}_j}{\partial \mathbf{x}_i}(\mathbf{x}) \tag{2.22}$$

which leads to a higher accuracy of the gradient approximation at the cost of more function evaluations, cf. [110, sec. 8.1]. Independently from the choice of the difference quotient, in theory the approximation becomes arbitrary good if ε is chosen small enough. However, in practice this is not true due to cancellation errors which become relevant when ε gets too small. Therefore, when computing derivatives with finite differences one has to accept a significant loss of accuracy. Even for an optimal choice of ε (which depends on $\mathbf{f}_j(\cdot)$) one has to expect a loss of about one half of the significant digits of the function evaluation for the one-sided Approximation Scheme (2.21) and of about one third in case of the Central Scheme (2.22). For more details we refer to [3, sec. 2.2] and [110, sec. 8.1]. The above considerations transfer to directional derivatives, cf. [3, sec. 2.3].

Automatic Differentiation

The basic idea of Automatic Differentiation (AD) is to view a function as a concatenation of elementary functions with known derivative and to make excessive use of the chain rule in order to compute (directional) derivatives. In contrast to symbolic differentiation, "[...] the chain rule is not applied to manipulate symbolic expressions but works on numerical values" [3, p. 26]. Given a piece of computer code

for the function evaluation, AD software tools (such as `Adol-C` [148]) break down the function evaluation into elementary operations and generate a computational graph which is used for the derivative generation. This way one can efficiently compute derivative values exact up to machine precision. As entry points to the topic we refer to [3, sec. 2.3] and [110, sec. 8.2], respectively, where the latter reference also comments on limitations of the usage of AD tools.

2.5.2 Sensitivity Generation

In this section we follow [3], [69, sec. II.6], and [81, sec. 3.4], where further information can be found, respectively. We again consider the Multiple Shooting Discretization. To solve the resulting NLP of Form (2.19) with gradient-based methods (e. g., Interior-Point or SQP methods), we have to deal with so-called *sensitivities*

$$\frac{\partial}{\partial \mathbf{s}^i}\mathbf{x}\left(\tau_{i+1};\mathbf{s}^i,\mathbf{q}^i\right) \quad \text{and} \quad \frac{\partial}{\partial \mathbf{q}^i}\mathbf{x}\left(\tau_{i+1};\mathbf{s}^i,\mathbf{q}^i\right)$$

where $\mathbf{x}\left(\tau_{i+1};\mathbf{s}^i,\mathbf{q}^i\right)$ denotes the solution of the IVP (2.17) at $t=\tau_{i+1}$. In practice, the expression $\mathbf{x}\left(\tau_{i+1};\mathbf{s}^i,\mathbf{q}^i\right)$ is evaluated by numerical integration. A straightforward approach for the computation of the sensitivities is the so-called external numerical differentiation. Here, the numerical integrator is treated a black box function which maps $(\mathbf{s}^i,\mathbf{q}^i)$ to an approximation of $\mathbf{x}\left(\tau_{i+1};\mathbf{s}^i,\mathbf{q}^i\right)$, and the sensitivities are approximated by finite differences. This approach is easy to implement. However, due to the adaptivity of the integrator, the integrator output cannot be assumed to depend smoothly on $\mathbf{s}^i$ and $\mathbf{q}^i$. This impairs the accuracy of the sensitivity approximation and raises the need for very tight integration tolerances, cf. [69, sec. II.6], which renders the approach unfavorable in practice.

An approach to overcome this issue is Internal Numerical Differentiation (IND) which is described in [22, 23]. The idea is to fix the adaptive elements of the integrator after computation of the nominal (i. e., unperturbed) trajectory and to differentiate the generated discretization scheme which can be seen as a concatenation of differentiable functions that map the values of the solution trajectory from one grid point to another. Thus, for instance one can apply AD to achieve the exact derivatives (up to machine precision) of the approximation of the nominal trajectory obtained by the integrator. Compared to external numerical differentiation, IND significantly reduces the accuracy requirements on the numerical integration which has a substantial impact on the computational effort.