Chair of Automatic Control

FAU

Numerical Optimization and Model Predictive Control

Prof. Dr.-Ing. Knut Graichen

Summer Semester 2023

Chair of Automatic Control

Faculty of Engineering

Friedrich-Alexander-Universität Erlangen-Nürnberg

"Cum enim mundi universi fabrica sit perfectissima atque a Creatore sapientissimo absoluta, nihil omnino in mundo contingit, in quo non maximi minimive ratio quaepiam eluceat; quamobrem dubium prorsus est nullum, quin omnes mundi effectus ex causis finalibus ope methodi maximorum et minimorum aeque feliciter determinari queant, atque ex ipsis causis efficientibus."

"As the construction of the entire world was most perfectly accomplished by the wisest Creator, nothing happens in this world in which the relation of a maximum or minimum does not shine forth; therefore there is absolutely no doubt that all effects in this world that are due to final causes can be equally determined from the method of maxima and minima as from the effected causes themselves." (freely translated)

Leonhard Euler, *Calculus of Variations*, 1744.

Contents

1 Introduction **1**

1.1 Parameter optimization problems 1

1.1.1 Mathematical formulation 1

1.1.2 Examples 2

1.2 Dynamic optimization problems 5

1.2.1 Mathematical formulation 5

1.2.2 Examples 6

1.3 References 10

2 Basics of optimization **11**

2.1 Local/global minima and admissible set 11

2.2 Gradient and Hessian matrix 13

2.3 Convexity 14

2.3.1 Convex sets 14

2.3.2 Convex functions 15

2.3.3 Convex optimization problems 18

2.4 References 19

3 Unconstrained optimization **21**

3.1 Necessary optimality conditions 21

3.2 Sufficient optimality conditions 22

3.3 Line search methods 23

3.3.1 Step size computation 25

3.3.2 Gradient method 28

3.3.3 Conjugate gradient method 30

3.3.4 Newton's method 30

3.3.5 Quasi-Newton method 32

3.4 Further numerical methods 33

3.4.1 Trust region method 33

3.4.2 Direct search methods 35

3.5 Example: Rosenbrock's "banana function" 37

3.6 References 40

4 Constrained optimization 43
4.1 Illustrative examples . . . 43
4.1.1 One equality constraint . . . 43
4.1.2 One inequality constraint . . . 45
4.1.3 Two inequality constraints . . . 47
4.2 Optimality conditions . . . 48
4.2.1 Constraint qualification . . . 48
4.2.2 First-order optimality conditions . . . 49
4.2.3 Second-order optimality conditions . . . 51
4.2.4 Interpretation of Lagrange multipliers . . . 52
4.3 Linear optimization . . . 53
4.3.1 Standard form . . . 54
4.3.2 Optimality conditions . . . 54
4.3.3 Simplex method . . . 55
4.3.4 Interior point method . . . 60
4.3.5 Example: Transport network . . . 63
4.4 Quadratic optimization . . . 64
4.4.1 Optimality conditions . . . 65
4.4.2 Equality-constrained quadratic problems . . . 65
4.4.3 Active set method . . . 66
4.4.4 Further numerical methods . . . 69
4.5 Nonlinear optimization . . . 69
4.5.1 Sequential quadratic programming (SQP) . . . 69
4.5.2 Further numerical methods . . . 74
4.6 Lagrangian duality . . . 76
4.6.1 Dual formulation . . . 76
4.6.2 Augmented Lagrangian method . . . 82
4.6.3 Consideration of inequality constraints . . . 84
4.6.4 Remarks . . . 85
4.7 Software overview . . . 86
4.8 References . . . 88

5 Dynamic optimization 89
5.1 Problem statement . . . 89
5.2 Calculus of variations . . . 91
5.2.1 Fundamentals . . . 91
5.2.2 Euler-Lagrange equations . . . 94
5.2.3 Example: Galileo's hanging chain . . . 96
5.3 Unconstrained optimal control problems . . . 97
5.3.1 Application of the calculus of variations . . . 98
5.3.2 Optimality conditions . . . 100

5.3.3 Example with free end time . . . 103
5.3.4 General procedure and singular case . . . 105
5.3.5 Interpretation of adjoint variables . . . 108
5.3.6 General terminal conditions . . . 109
5.4 Optimization of linear systems with quadratic cost functional . . . 110
5.4.1 Riccati differential equation . . . 110
5.4.2 Example: Swing-up of the pendulum on a cart . . . 111
5.4.3 Algebraic Riccati equation . . . 114
5.5 Optimal control problems with input constraints . . . 117
5.5.1 Pontryagin's maximum principle . . . 117
5.5.2 General procedure . . . 119
5.5.3 Minimization of the Hamiltonian . . . 120
5.5.4 Example: Double integrator . . . 123
5.5.5 State-dependent input constraints . . . 125
5.6 Numerical solution of optimality conditions (indirect methods) . . . 126
5.6.1 Discretization method . . . 126
5.6.2 Shooting method . . . 128
5.6.3 Handling of general terminal constraints and free end time . . . 129
5.6.4 Gradient method . . . 130
5.7 Numerical solution by discretization (direct methods) . . . 132
5.7.1 Partial discretization . . . 133
5.7.2 Full discretization . . . 134
5.8 Software overview . . . 135
5.9 References . . . 135

6 Model predictive control (MPC) 137

6.1 Basic concept behind MPC . . . 137
6.1.1 Dynamic optimization on moving horizon . . . 137
6.1.2 Continuous-time vs. discrete-time MPC . . . 139
6.2 MPC formulations . . . 140
6.2.1 Infinite horizon . . . 140
6.2.2 Finite horizon and terminal condition . . . 142
6.2.3 Finite horizon and terminal set . . . 143
6.2.4 Finite horizon and terminal cost . . . 145
6.2.5 Sufficiently long horizon . . . 147
6.3 Real-time implementation . . . 151
6.3.1 Suboptimal solution strategy . . . 152
6.3.2 Stability and incremental improvement . . . 153
6.3.3 Application example . . . 155
6.4 Software overview . . . 158
6.5 References . . . 159

1 Introduction

The word "optimization" refers to the search for an optimal solution to a given problem with the meaning of "optimal" being more or less well defined. Optimization problems are widely present in industry and economy, some examples are:

- Economy and financial sector: portfolio optimization or company development (e.g. maximization of profit through optimal distribution of investments, expenses etc.)
- Process and product optimization: process scheduling, minimization of production costs, energy efficient operation, etc.
- Control engineering: optimal feedback control, optimal trajectory planning, parameter estimation, etc.

The mathematical formulation of these problems leads to optimization problems that can be classified as follows:

- *Parameter optimization*: minimization of a function with the optimization variables being elements of the Euclidean space
- *Dynamic optimization*: minimization of a functional with the optimization variables being elements of a Hilbert space (e.g. time functions).

In this chapter, we present some introductory examples to highlight the difference between parameter and dynamic optimization.

1.1 Parameter optimization problems

Mathematically speaking, a *parameter optimization problem* minimizes a function $f(\boldsymbol{x})$ subject to additional constraints, where the optimization variables $\boldsymbol{x}$ are elements of the Euclidean space $\mathbb{R}^n$.

1.1.1 Mathematical formulation

A general nonlinear parameter optimization problem is typically expressed as

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \quad f(\boldsymbol{x}) \qquad \text{cost function} \tag{1.1}$$

$$\text{s.t.} \quad g_i(\boldsymbol{x}) = 0, \quad i = 1,\dots,p \qquad \text{equality constraints} \tag{1.2}$$

$$h_i(\boldsymbol{x}) \leq 0, \quad i = 1,\dots,q \qquad \text{inequality constraints} \tag{1.3}$$

We speak of *constrained* optimization, if the cost function $f(\boldsymbol{x})$ to be minimized is subject to (s.t.) constraints of the form (1.2) or (1.3), otherwise (1.1) is referred to as *unconstrained* optimization. The min-formulation in (1.1) became standard over the decades. A maximization problem can equally be formulated as minimization problem by negating the cost function

$$\max_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x}) = \min_{\boldsymbol{x}\in\mathbb{R}^n} -f(\boldsymbol{x}) .$$

Further well known terms for parameter optimization are *finite-dimensional optimization* or *mathematical programming*. The term "programming" is historically rooted, as the first applications of optimization were military action plans:

> "It is a word used by the military for a plan, a schedule, or in general for a program of actions. This is exactly what it means for computers: a program of actions for the computer to execute."
>
> George Danzig
> Founder of linear programming

Parameter optimization is typically differentiated in the following classes:

- *Linear programming*: The cost function and constraints are linear.
- *Quadratic programming*: The cost function is quadratic, the constraints are linear.
- *Nonlinear programming*: The cost function or at least one of the constraints is nonlinear.
- *Integer programming*: All optimization variables are discrete.
- *Mixed-integer programming*: The optimization variables are partly continuous and discrete.

1.1.2 Examples

Linear programming is often used in economic optimization tasks such as production planning and investment problems. The following example is a strongly simplified portfolio optimization problem.

Example 1.1 (Portfolio optimization) *An investor wants to invest 10000 euros in three equity funds with different profit expectations and risk ratings:*

Fund	*Expected profit/year*	*Risk classification*
A	10%	4
B	7%	2
C	4%	1

The investor wants to make a profit of at least 600 euros after one year. On the other hand, he would like to invest his money rather conservatively, i.e. at least 4000 euros shall be invested in fund C and the overall risk is to minimized. What is the optimal allocation of the investor's 10000 euros in order to fulfill these criteria?

The first step for solving the portfolio optimization problem is its mathematical formulation. To this end, we define the optimization variables $x_1, x_2, x_3 \geq 0$ as the ratios of the 10000 euros invested in fund A, B, and C, respectively. Hence, the third variable x_3 can be substituted by the relation

$$x_3 = 1 - x_1 - x_2 \,.$$

The requested minimum profit of 600 euros corresponds to the inequality constraint

$$10000\Big[0.1x_1 + 0.07x_2 + 0.04(1 - x_1 - x_2)\Big] \geq 600 \quad \Rightarrow \quad 6x_1 + 3x_2 \geq 2 \,. \tag{1.4}$$

Likewise, the minimum investment of 4000 euros in fund C implies

$$10000(1 - x_1 - x_2) \geq 4000 \quad \Rightarrow \quad x_1 + x_2 \leq 0.6 \,. \tag{1.5}$$

The optimization objective is to minimize the overall risk. This can be expressed by the function

$$f(\boldsymbol{x}) = 4x_1 + 2x_2 + (1 - x_1 - x_2) = 1 + 3x_1 + x_2 \,. \tag{1.6}$$

In summary, the linear optimization problem becomes

$$\begin{aligned}
\min_{\boldsymbol{x}\in\mathbb{R}^2} \quad & f(\boldsymbol{x}) = 1 + 3x_1 + x_2 && \text{(1.7a)}\\
\text{s.t.} \quad & 6x_1 + 3x_2 \geq 2 && \text{(1.7b)}\\
& x_1 + x_2 \leq 0.6 && \text{(1.7c)}\\
& x_1 + x_2 \leq 1 && \text{(1.7d)}\\
& x_1, x_2 \geq 0 \,. && \text{(1.7e)}
\end{aligned}$$

Figure 1.1 shows the single constraints and the admissible set. The shape of the contour lines $f(\boldsymbol{x}) = const.$ of the cost function (1.7a) illustrates that the point of the admissible set with the lowest value of $f(\boldsymbol{x})$ is located in the corner at $\boldsymbol{x}^*$. Hence, the optimal distribution of the investment of 10000 euros over the three funds is

$$x_1^* = \frac{1}{15}, \quad x_2^* = \frac{8}{15}, \quad x_3^* = \frac{6}{15} \,. \tag{1.8}$$

The following (academic) optimization problem is intended to illustrate the influence of constraints on an optimal solution.

Example 1.2 *We start with the unconstrained quadratic problem*

$$\min_{\boldsymbol{x}\in\mathbb{R}^2} \; f(\boldsymbol{x}) = (x_1 - 2)^2 + (x_2 - 1)^2 \,. \tag{1.9}$$

The contour lines $f(\boldsymbol{x}) = const.$ of the function $f(\boldsymbol{x})$ are shown in Figure 1.2 as a function of the two optimization variables $\boldsymbol{x} = [x_1, x_2]^\mathsf{T}$. It is easy to see that the minimum $f(\boldsymbol{x}^*) = 0$ is located at the position $\boldsymbol{x}^* = [2, 1]^\mathsf{T}$.

In order to investigate the influence of constraints on the optimal solution, we add an equality constraint of the form (1.2)

$$g(\boldsymbol{x}) = x_2 - 2x_1 = 0 \,. \tag{1.10}$$

The geometric interpretation is that a possible solution must lie on the straight line given by (1.10), see Figure 1.3. In particular, the optimal solution lies on the touchpoint of the straight line $g(\boldsymbol{x}) = 0$ with the contour line $f(\boldsymbol{x}) = 1.8$.

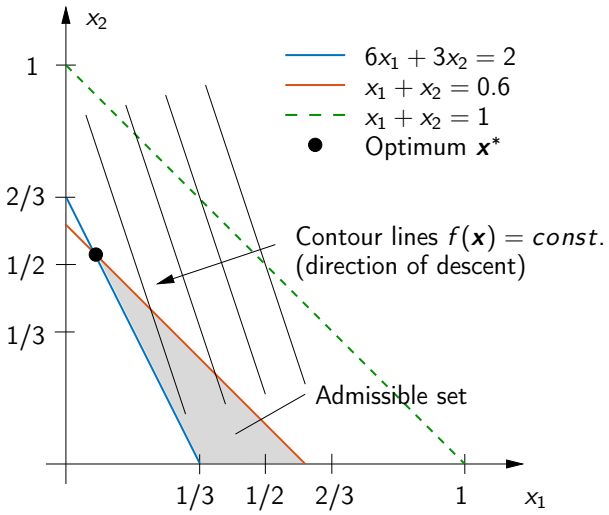


Figure 1.1: Illustration of the portfolio optimization problem in Example 1.1.

We now substitute the equality constraint (1.10) with the inequality constraint

$$h_1(\boldsymbol{x}) = x_1 + x_2 - 2 \leq 0\,, \tag{1.11}$$

which reduces the set of admissible points to the region below the straight line $h_1(\boldsymbol{x}) = 0$ (Figure 1.4). The optimum $f(\boldsymbol{x}^*) = 0.5$ at the point $\boldsymbol{x}^* = [1.5, 0.5]^\mathrm{T}$ lies at the boundary of the admissible set on a contour line which touches $h_1(\boldsymbol{x}) = 0$.

The admissible set is further reduced if we consider a second inequality constraint

$$h_2(\boldsymbol{x}) = x_1^2 - x_2 \leq 0\,. \tag{1.12}$$

The optimal point $\boldsymbol{x}^* = [1, 1]^\mathrm{T}$ with the minimum $f(\boldsymbol{x}^*) = 1$ now lies in the intersection of the curves $h_1(\boldsymbol{x}) = 0$ and $h_2(\boldsymbol{x}) = 0$, i.e. both constraints (1.11) and (1.12) are active, see Figure 1.5.

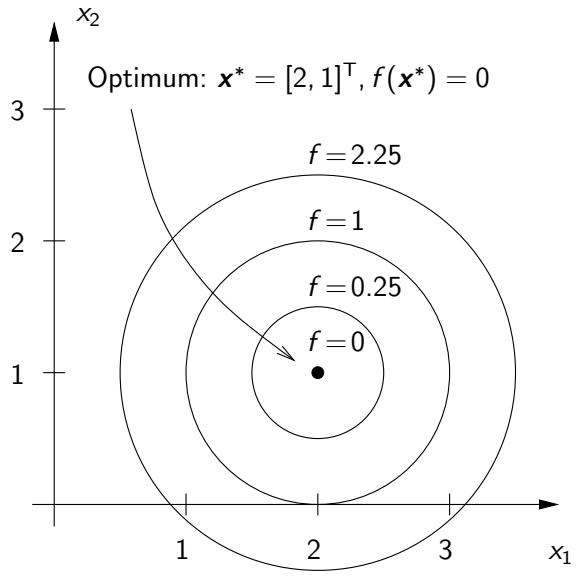


Figure 1.2: Geometric illustration of the unconstrained problem (1.9).

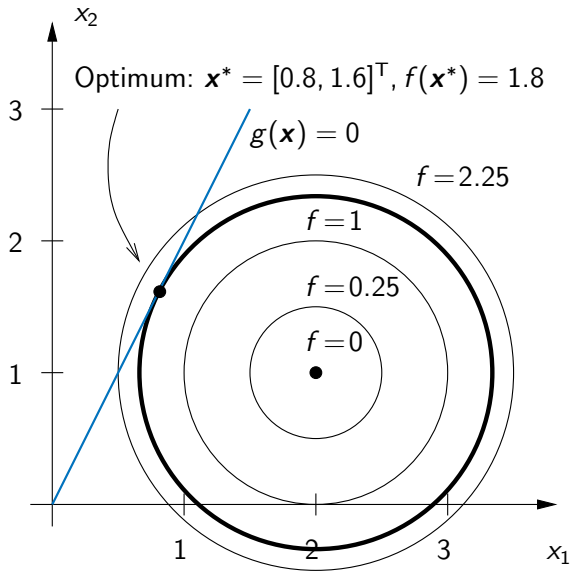


Figure 1.3: Geometric ilustration of the constrained problem (1.9), (1.10).

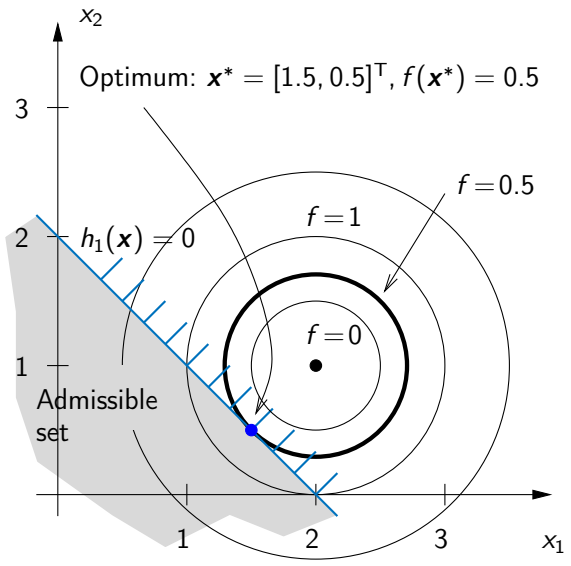


Figure 1.4: Geometric ilustration of the constrained problem (1.9), (1.11).

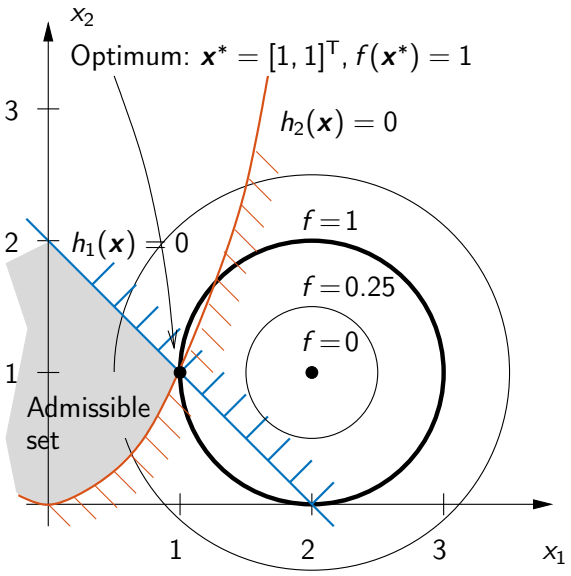


Figure 1.5: Geometric ilustration of the constrained problem (1.9), (1.11), (1.12).

Example 1.2 illustrates the influence of equality and inequality constraints on the solution (and solvability) of the optimization problem (1.1)-(1.3). The systematic analysis of parameter optimization problems as well as numerical methods for their solution will be covered in the chapters to come.

1.2 Dynamic optimization problems

As shown in the last section, the optimization variables in parameter optimization problems are elements of the Euclidean space $\mathbb{R}^n$. In dynamic optimization, we are concerned with minimizing a functional subject to constraints, where the optimization variables are elements of a Hilbert space. In particular, we speak of *dynamic optimization* if the optimization variables are functions of time t.

1.2.1 Mathematical formulation

The general structure of a dynamic optimization problem is

$$\min_{\boldsymbol{u}(\cdot)} \quad J(\boldsymbol{u}) = V(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} l(\boldsymbol{x}(t), \boldsymbol{u}(t), t)\,\mathrm{d}t \qquad \text{cost functional} \qquad (1.13a)$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t), \qquad \boldsymbol{x}(t_0) = \boldsymbol{x}_0 \qquad \text{system dynamics \& IC} \qquad (1.13b)$$

$$\boldsymbol{g}(\boldsymbol{x}(t_f), t_f) = \boldsymbol{0} \qquad \text{terminal constraints} \qquad (1.13c)$$

$$h_i(\boldsymbol{x}, \boldsymbol{u}) \leq 0, \qquad i = 1, \ldots, q \qquad \text{inequality constr.} \qquad (1.13d)$$

where $\boldsymbol{u} \in \mathbb{R}^m$ is the control input for the nonlinear system (1.13b) with the state $\boldsymbol{x} \in \mathbb{R}^n$ and the respective initial conditions (IC). The terminal conditions of the general form (1.13c) may e.g. be relevant to reach a desired state $\boldsymbol{x}_f$ at the end time t_f (i.e. $\boldsymbol{g}(\boldsymbol{x}(t_f), t_f) = \boldsymbol{x}(t_f) - \boldsymbol{x}_f$).

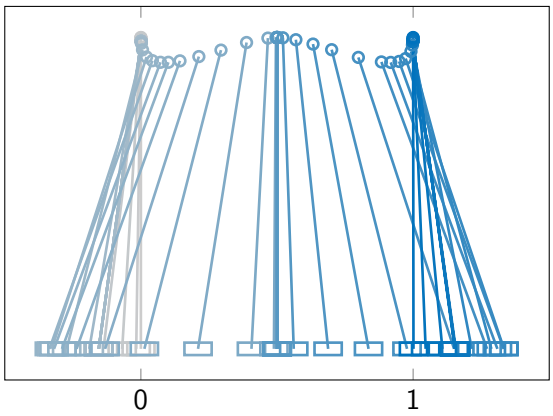


Figure 1.6: Snapshots of the side-stepping of the pendulum (Example 1.3).

The inequality constraints (1.13d) are also relevant in many practical applications, for instance, to account for constraints on the control $\boldsymbol{u}$ or on the state $\boldsymbol{x}$.

The problem of dynamic optimization is to find an optimal control trajectory $\boldsymbol{u}^*(t)$, $t \in [t_0, t_f]$ and associated state trajectory $\boldsymbol{x}^*(t)$ such that the cost functional (1.13a) is minimized while the dynamical system (1.13b), the terminal conditions (1.13c), and the constraints (1.13d) are satisfied. The end time t_f may be fixed (specified beforehand) or unknown. In the latter case, (1.13) is a *free end-time problem* and t_f is an additional optimization variable for solving (1.13).

Other well known terms for dynamic optimization are *infinite-dimensional optimization* or *dynamic programming*. The problem (1.13) is also called *optimal control problem*.

1.2.2 Examples

The following lines present some illustrative examples to familiarize ourselves with the concept of dynamic optimization.

Example 1.3 (Inverted pendulum) *A classical benchmark problem in control is the inverted pendulum on a cart. We consider a side-stepping scenario formulated as the following optimal control problem*

$$\min_{u(\cdot)} \quad J(u) = \int_0^{t_f} 1 + cu^2 \, \mathrm{d}t \tag{1.14a}$$

$$\text{s.t.} \quad \begin{bmatrix} 1 & \varepsilon\cos\theta \\ \cos\theta & 1 \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \varepsilon\dot{\theta}^2 \sin\theta + u \\ -\sin\theta \end{bmatrix}, \quad \varepsilon = m/(M+m) \tag{1.14b}$$

$$\boldsymbol{x}(0) = [0,0,0,0]^\mathsf{T}, \quad \boldsymbol{x}(t_f) = [1,0,0,0]^\mathsf{T} \tag{1.14c}$$

$$-1 \leq u \leq 1\,. \tag{1.14d}$$

The simplified and normalized equations of motion for the states $\boldsymbol{x} = [x, \dot{x}, \theta, \dot{\theta}]^\mathsf{T}$ are given by (1.14b) with the parameter ε that involves the mass of the pendulum and the cart, respectively. The input force u acting on the cart is box-constrained by (1.14d). Figure 1.6 shows an example of the lateral displacement of the pendulum to illustrate the movement.

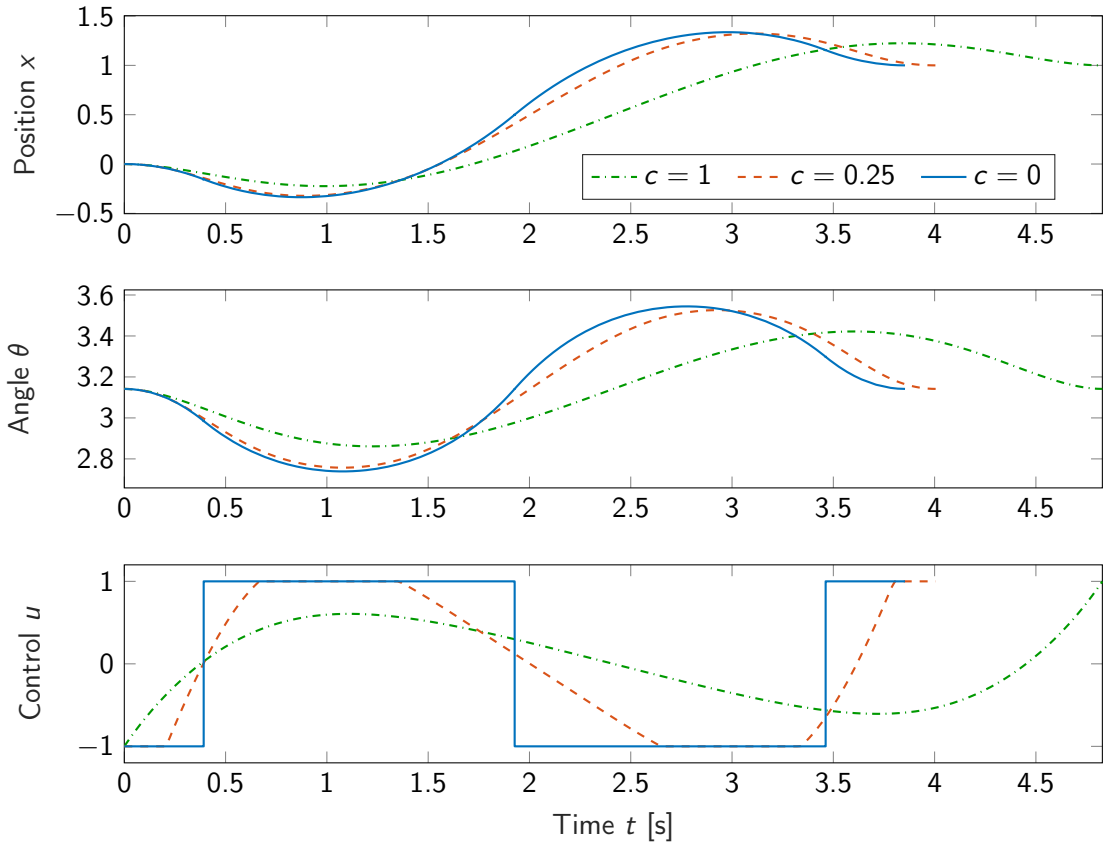


Figure 1.7: Optimal trajectories for the side-stepping of the pendulum (Example 1.3).

The cost functional (1.14a) and therefore the characteristics of the dynamic optimization problem depends on the parameter c. The choice $c = 0$ implies the minimization of the end time

$$J(u) = \int_0^{t_f} 1 \, \mathrm{d}t = t_f \,, \tag{1.15}$$

which is also known as time optimality. For $c > 0$, the squared control u is weighted in the cost functional as a trade-off between time and energy optimality.

Figure 1.7 depicts the optimal trajectories for $c = 0$, $c = 0.25$, and $c = 1$. For $c = 0$, the input u shows a bang-bang behavior, whereas for $c > 0$ the control amplitudes decrease and the transition time t_f increases.

The inverted pendulum is a good example to illustrate that not every optimal control problem has a solution, especially when the end time t_f is not fixed. As shown in Figure 1.7, the end time t_f increases as the weighting of u^2 increases in comparison to the time-optimal part in the cost functional (1.14a). In particular, $c \to \infty$ corresponds to energy optimality, i.e.

$$J(u) = \int_0^{t_f} u^2 \, \mathrm{d}t \,. \tag{1.16}$$

In this case, the optimal control problem has no finite solution, because the side-stepping maneuver of the pendulum would take infinitely long with $t_f \to \infty$.

Example 1.4 (Goddard problem [2, 1]) *A classical aerospace optimal control problem is the maximization of the altitude of a rocket under the influence of drag and gravity. This problem was posed and solved by the American rocket pioneer Robert H. Goddard in 1919 and can be formulated as the normalized optimal control problem*

$$\min_{u(\cdot)} \quad -h(t_f) \tag{1.17}$$

$$\text{s.t.} \quad \dot{h} = v, \quad \dot{v} = \frac{u - D(h,v)}{m} - \frac{1}{h^2}, \quad \dot{m} = -\frac{u}{c}, \tag{1.18}$$

$$h(0) = 1, \quad v(0) = 0, \quad m(0) = 1, \quad m(t_f) = 0.6, \tag{1.19}$$

$$0 \leq u \leq 3.5. \tag{1.20}$$

The state variables are the altitude h, velocity v, and the mass m of the rocket. The drag is modeled with the function

$$D(h,v) = D_0 v^2 \exp\left[\beta(1-h)\right], \quad D_0 > 0 \tag{1.21}$$

that depends on the states h und v. The boundary conditions in (1.19) consist of the (normalized) initial conditions and the terminal condition for $m(t_f)$ that corresponds to the net mass of the rocket after the fuel is completely burnt. The control input of the system is the thrust u subject to the constraint (1.20).

Figure 1.8 shows the optimal trajectories for the Goddard rocket compared to a flight with maximum thrust.[1] *The optimal control trajectory $u^*(t)$ also starts with full thrust, but then breaks down and increases again with a parabolic shape until the fuel is consumed. This behavior is caused by the drag function $D(h,v)$. At the start of the flight, the velocity v is low and therefore the influence of drag is small which justifies to fly with full thrust. As the velocity increases, however, the drag becomes dominant and it is more economic to save fuel. As the altitude increases, the drag effect becomes smaller and the thrust increases again. This optimal strategy increases the final altitude by 2.7% compared to the full thrust flight, see Figure 1.8.*

Example 1.5 (Economic model [4, 3]) *The following example describes the behavior of a consumer who wants to maximize consumption, leisure time, and education over his lifetime. The level of education E and the capital C of an average consumer are described by the dynamics*

$$\dot{E} = \overbrace{E^{\varepsilon} u_2 u_3}^{\text{Education}} - \overbrace{\delta E}^{\text{Forgetting}}, \qquad E(0) = E_0 \tag{1.22}$$

$$\dot{C} = \underbrace{iC}_{\text{Interest}} + \underbrace{E u_2 g(u_3)}_{\text{Income}} - \underbrace{u_1}_{\text{Consumption}}, \quad C(0) = C_0. \tag{1.23}$$

The control variables are the consumption u_1, the ratio u_2 of working time w.r.t. the total time, and the ratio u_3 of education time w.r.t. the working time. The controls are subject to the constraints

$$u_1 > 0, \quad 0 \leq u_2 \leq 1, \quad 0 \leq u_3 < 1. \tag{1.24}$$

[1] The parameter values are $c = 0.5$, $D_0 = 310$, $\beta = 500$.

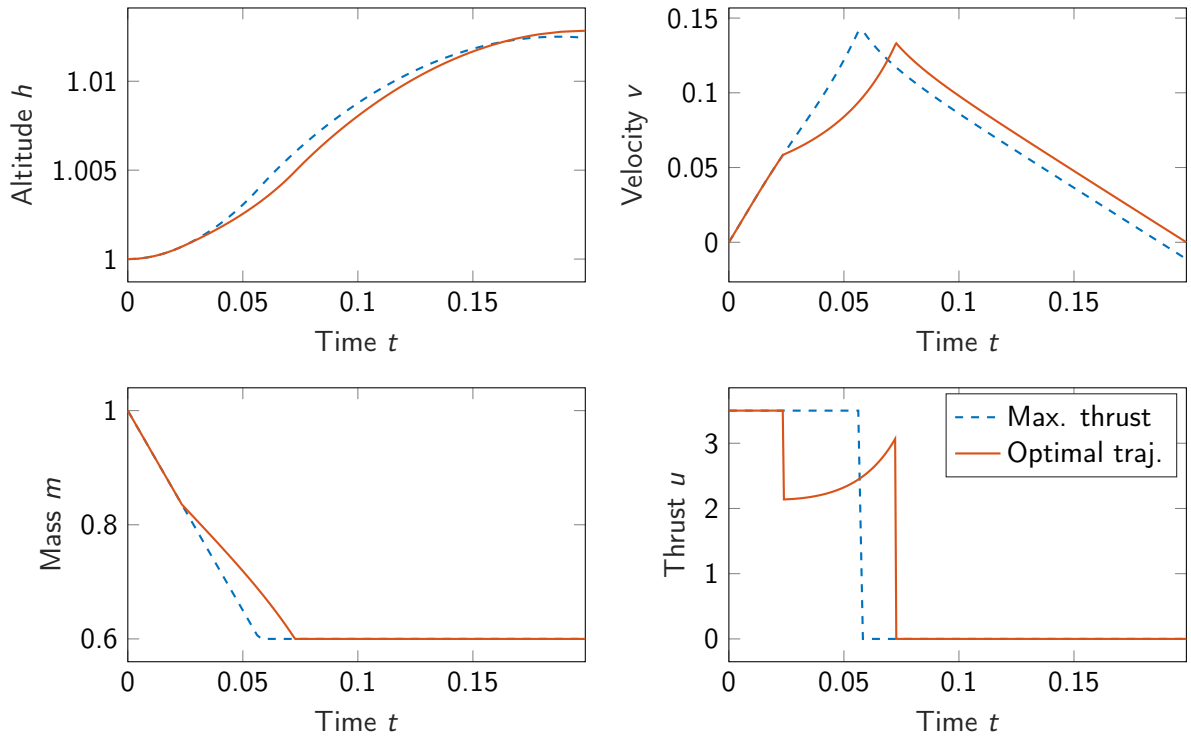


Figure 1.8: Optimal trajectories of the Goddard problem (Example 1.4).

As mentioned before, the consumer's optimization goal is to maximize consumption, leisure time, and education over the lifetime of $t_f = 75$ years, which is expressed in the following cost functional to be minimized

$$J(\boldsymbol{u}) = -C^{\kappa}(t_f) - \int_{t_0}^{t_f} U(t, u_1, u_2, E) e^{-\rho t} \, \mathrm{d}t \,. \tag{1.25}$$

The integral cost function

$$U(t, u_1, u_2, E) = \alpha_0 u_1^{\alpha} + \beta_0 (1 - u_2)^{\beta} + \gamma_0 t E^{\gamma} \tag{1.26}$$

weights the consumption u_1, the leisure time $(1 - u_2)$ and the level of education E. The term $-C^{\kappa}(t_f)$ in (1.25) additionally accounts for the capital of inheritance.

The optimal trajectories for the level of educational $E(t)$ and the capital $C(t)$ are shown in Figure 1.9.[2] *The first 17 years are the education phase of the life (i.e. $u_3 = 1$), followed by a long working phase of 34 years with ongoing education. During the last working phase of 10 years (age 52 to 61), the working time is continuously reduced (control u_2). From the 62nd year onward, retirement sets in. The level of education E is particularly high at age 30 to 60. The capital C is negative during the first half of life, which corresponds to taking out a loan. In the course of life, however, this is compensated by the rising income.*

[2] The following parameter values are used: $\alpha = -1$, $\alpha_0 = -1$, $\beta = -0.5$, $\beta_0 = -1$, $\gamma = 0.2$, $\gamma_0 = 5$, $\kappa = 0.2$, $\rho = 0.01$, $\varepsilon = 0.35$, $\delta = 0.01$, $i = 0.04$, $H_0 = 1$, $C_0 = 30$. The function $g(u_3)$ in (1.23) is defined by the parabola $g(u_3) = 1 - (1 - a)u_3 - au_3^2$.

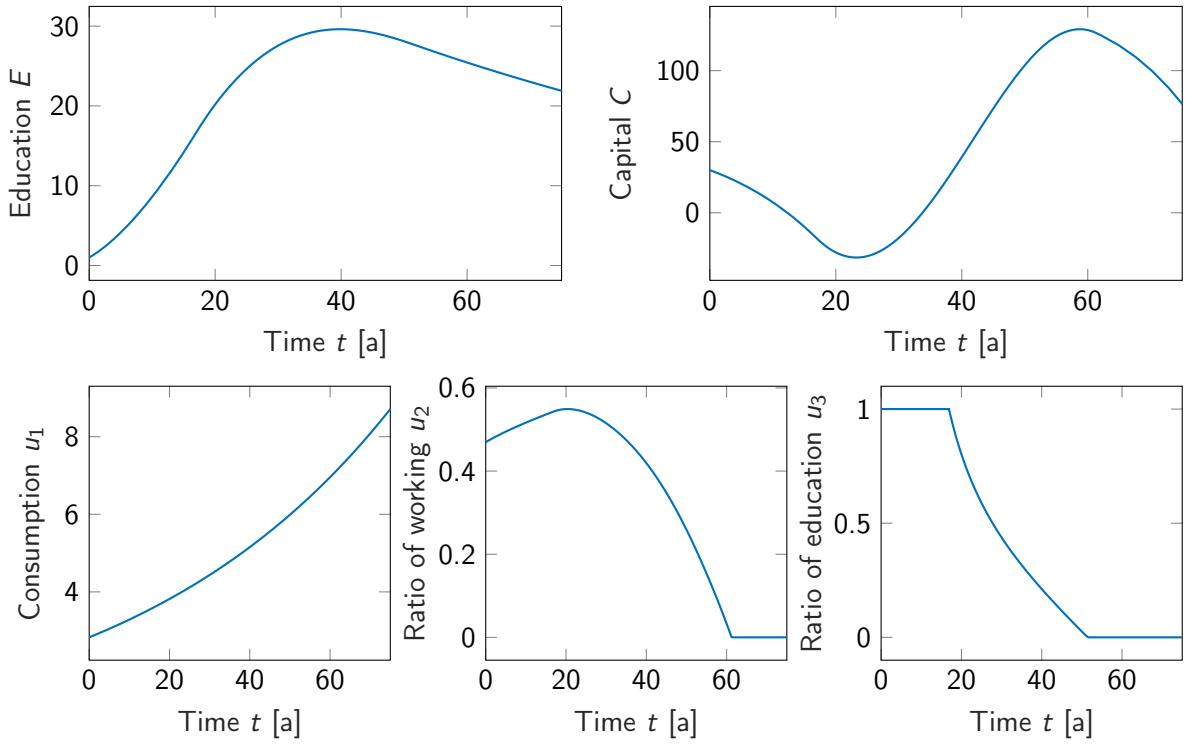


Figure 1.9: Optimal consumer behavior (Example 1.5).

1.3 References

[1] A.E. Bryson. *Dynamic Optimization*. Menlo Park, CA: Addison-Wesley, 1999.

[2] R.H. Goddard. "A method for reaching extreme altitudes". In: *Smithsonian Miscellaneous Collections 71* (1919).

[3] H.J. Oberle and R. Rosendahl. "Numerical computation of a singular-state subarc in an economic optimal control model". In: *Optimal Control Applications and Methods* 27 (2006), pp. 211–235.

[4] K. Pohmer. *Mikroökonomische Theorie der personellen Einkommens- und Vermögensverteilung*. Studies in Contemporary Economics. Vol. 16. Springer, 1985.

2 Basics of optimization

This chapter introduces basic concepts and properties of parameter optimization problems

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \quad f(\boldsymbol{x}) \tag{2.1}$$

$$\text{s.t.} \quad g_i(\boldsymbol{x}) = 0, \quad i = 1, \dots, p \tag{2.2}$$

$$h_i(\boldsymbol{x}) \leq 0, \quad i = 1, \dots, q \tag{2.3}$$

that are relevant for the following chapters.

2.1 Local/global minima and admissible set

Definition 2.1 (Local and global minima) *The function $f(\boldsymbol{x})$ with $f : \mathcal{X} \to \mathbb{R}$ and $\mathcal{X} \subseteq \mathbb{R}^n$ has*

a) a local minimum *at $\boldsymbol{x}^* \in \mathcal{X}$, if $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$ for all $\boldsymbol{x}$ in a neighborhood of $\boldsymbol{x}^*$,*

b) a strict local minimum, *if $f(\boldsymbol{x}^*) < f(\boldsymbol{x})$ for all $\boldsymbol{x}$ in a neighborhood of $\boldsymbol{x}^*$,*

c) a global minimum, *if $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{X}$,*

c) a unique global minimum, *if $f(\boldsymbol{x}^*) < f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{X}$.*

Definition (2.1) implies that a global minimum is always a local minimum. The different types of minima are illustrated in Figure 2.1.

Definition 2.2 (Admissible set) *The admissible set $\mathcal{X}_{ad}$ of the optimization problem (2.1)-(2.3) is defined as*

$$\mathcal{X}_{ad} = \Big\{ \boldsymbol{x} \in \mathbb{R}^n : \; g_i(\boldsymbol{x}) = 0, \quad i = 1, \dots, p \\ h_i(\boldsymbol{x}) \leq 0, \quad i = 1, \dots, q \Big\}. \tag{2.4}$$

All *admissible points* $\boldsymbol{x} \in \mathcal{X}_{ad}$ satisfy the constraints $g_i(\boldsymbol{x}) = 0$ and $h_i(\boldsymbol{x}) \leq 0$. This shows that the problem (2.1)-(2.3) can also be formulated as

$$\min_{\boldsymbol{x}\in\mathcal{X}_{ad}} \quad f(\boldsymbol{x}) \,. \tag{2.5}$$

For unconstrained problems, we have $\mathcal{X}_{ad} = \mathbb{R}^n$.

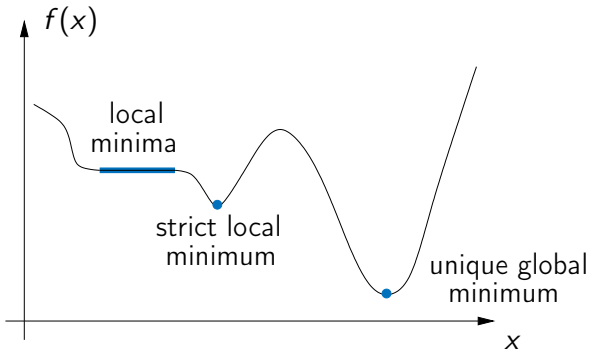


Figure 2.1: Different minima of a function $f(x)$ with $x \in \mathbb{R}$.

Example 2.1 *The admissible set for the problem in Example 1.2 with the two inequality constraints (1.11) and (1.12) is*

$$\mathcal{X}_{ad} = \left\{ \boldsymbol{x} \in \mathbb{R}^n \,|\, h_1(\boldsymbol{x}) = x_1 + x_2 - 2 \leq 0,\ h_2(\boldsymbol{x}) = x_1^2 - x_2 \leq 0 \right\}.$$

This corresponds to the region enclosed by the line $h_1(\boldsymbol{x}) = 0$ and the parabola $h_2(\boldsymbol{x}) = 0$, see Figure 1.5.

It is obvious that the admissible set $\mathcal{X}_{ad}$ must be non-empty, because otherwise the optimization problem (2.5) has no solution. Another necessary condition for $\mathcal{X}_{ad}$ can be derived from the equality constraint (2.2), since the algebraic restrictions $g_i(\boldsymbol{x}) = 0$ reduce the number of free optimization variables $\boldsymbol{x} \in \mathbb{R}^n$. This implies that the number p of (independent) equality constraints (2.2) must not be larger than the number of optimization variables $\boldsymbol{x} \in \mathbb{R}^n$ as a necessary condition for the admissible set $\mathcal{X}_{ad}$ to be non-empty.

Theorem 2.1 (Weierstrass) *Let $\mathcal{X}_{ad}$ be a compact non-empty set and $f : \mathcal{X}_{ad} \to \mathbb{R}$ a continuous function on $\mathcal{X}_{ad}$. Then, the problem*

$$\min_{\boldsymbol{x} \in \mathcal{X}_{ad}} f(\boldsymbol{x}) \tag{2.6}$$

attains its minimum at least at one point $\boldsymbol{x}^* \in \mathcal{X}_{ad}$.

Example 2.2 *Consider the constrained problem of Example 1.2 (also see Figure 1.5)*

$$\min_{\boldsymbol{x} \in \mathbb{R}^2} \quad (x_1 - 2)^2 + (x_2 - 1)^2 \tag{2.7}$$

$$\text{s.t.} \quad x_1 + x_2 - 2 \leq 0 \tag{2.8}$$

$$x_1^2 - x_2 \leq 0\,. \tag{2.9}$$

The cost function is continuous and the admissible set $\mathcal{X}_{ad}$ is non-empty and compact. Then, Theorem 2.1 implies that there exists at least one minimizing solution $\boldsymbol{x}^*$.

Note that Weierstrass' theorem does not hold for problems that are defined on an *open set* (e.g. $\mathcal{X}_{ad} = \mathbb{R}^n$). An example for this is

$$\min_{x \in (0,\infty)} \ \frac{1}{x}\,. \tag{2.10}$$

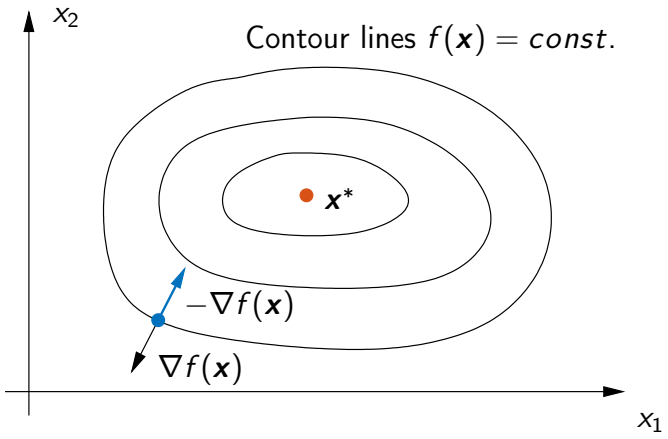


Figure 2.2: Direction of steepest descent $-\nabla f(\boldsymbol{x})$ for a function $f(\boldsymbol{x})$ with $\boldsymbol{x} \in \mathbb{R}^2$.

Although $f(x) = \frac{1}{x}$ is continuous on the open interval $\mathcal{X}_{ad} = (0, \infty)$, there exists no minimum. However, the existence of a solution to an unconstrained optimization problem can be guaranteed if the function $f(\boldsymbol{x})$ is *radially unbounded*, i.e. $\lim_{||\boldsymbol{x}|| \to \infty} f(\boldsymbol{x}) = \infty$ for all $\boldsymbol{x} \in \mathbb{R}^n$.

2.2 Gradient and Hessian matrix

The calculation of first and second order derivatives of the cost function and the constraints is of fundamental importance in optimization. The case of discontinuous functions or discontinuous derivatives is somewhat delicate as the "classical" optimality conditions that are derived in the following chapters do not hold for these problems and further theoretical and numerical problems arise. It is therefore often assumed that all functions of an optimization problem are continuous and sufficiently often differentiable.

Definition 2.3 (Gradient) *Suppose that $f : \mathcal{X} \to \mathbb{R}$ is continuously differentiable. Then, the* gradient *(i.e. the first partial derivative) of $f(\boldsymbol{x})$ at the point $\boldsymbol{x} = [x_1, \dots, x_n]^\mathsf{T}$ is defined by*

$$\nabla f(\boldsymbol{x}) = \left(\frac{\partial f}{\partial \boldsymbol{x}}\right)^\mathsf{T} = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}. \tag{2.11}$$

The negative gradient has a special meaning for the solution of an optimization problem as it represents the direction of steepest descent of a cost function $f(\boldsymbol{x})$ at the point $\boldsymbol{x}$, see Figure 2.2. This property will be examined in more detail in the next chapter.

Definition 2.4 (Hessian matrix) *Let $f : \mathcal{X} \to \mathbb{R}$ be twice continuously differentiable. Then,*

$$\nabla^2 f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 x_n} \\ \vdots & & \vdots \\ \frac{\partial^2 f}{\partial x_n x_1} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix} \tag{2.12}$$

is the Hessian matrix *or simply* Hessian *(i.e. the second partial derivative) of $f(\boldsymbol{x})$ at the point $\boldsymbol{x} = [x_1, \dots, x_n]^\mathsf{T}$.*

In case of scalar dependency, the ∇-notation is usually replaced by $f'(x)$ and $f''(x)$. According to the theorem of Schwarz, the continuity of the second partial derivative implies commutativity, i.e.

$$\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial^2 f}{\partial x_j \partial x_i} .$$

Hence, the Hessian matrix is symmetric, i.e. $\nabla^2 f(\boldsymbol{x}) = \nabla^2 f(\boldsymbol{x})^\mathrm{T}$, and always has real eigenvalues. An important property of the Hessian in the context of optimization is its definiteness that can be investigated as follows.

Theorem 2.2 (Definiteness) *The definiteness of a symmetric $(n \times n)$-matrix $\boldsymbol{A}$ is characterized by the following conditions:*

Matrix $\boldsymbol{A}$ is	*(1): for all $\boldsymbol{p} \in \mathbb{R}^n$ with $\boldsymbol{p} \neq \boldsymbol{0}$*	*(2): all eigenvalues λ_i are*	*(3): all principle minors D_i satisfy*
positive semidefinite ($\boldsymbol{A} \geq 0$)	$\boldsymbol{p}^\mathrm{T}\boldsymbol{A}\boldsymbol{p} \geq 0$	*real and* ≥ 0	
positive definite ($\boldsymbol{A} > 0$)	$\boldsymbol{p}^\mathrm{T}\boldsymbol{A}\boldsymbol{p} > 0$	*real and* > 0	$D_i > 0$
negative semidefinite ($\boldsymbol{A} \leq 0$)	$\boldsymbol{p}^\mathrm{T}\boldsymbol{A}\boldsymbol{p} \leq 0$	*real and* ≤ 0	
negative definite ($\boldsymbol{A} < 0$)	$\boldsymbol{p}^\mathrm{T}\boldsymbol{A}\boldsymbol{p} < 0$	*real and* < 0	$(-1)^{i+1} D_i < 0$

The eigenvalues λ_i, $i = 1, \ldots, n$ of the matrix $\boldsymbol{A}$ are the roots of the characteristic equation

$$\det(\lambda \boldsymbol{I} - \boldsymbol{A}) = 0$$

with the $(n \times n)$-unit matrix $\boldsymbol{I}$. Criterion (3) is also known as *Sylvester's criterion* for definite matrices. The principle minors D_i are the determinants of the upper left submatrices of $\boldsymbol{A} = [a_{ij}]$:

$$D_1 = a_{11} , \quad D_2 = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} , \quad \ldots \quad D_n = \begin{vmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \ldots & a_{nn} \end{vmatrix} .$$

Each of the three conditions (1)-(3) is necessary and sufficient. It is therefore only required to check one of them to determine the definiteness of the symmetric matrix $\boldsymbol{A}$, also see Exercise 2.1.

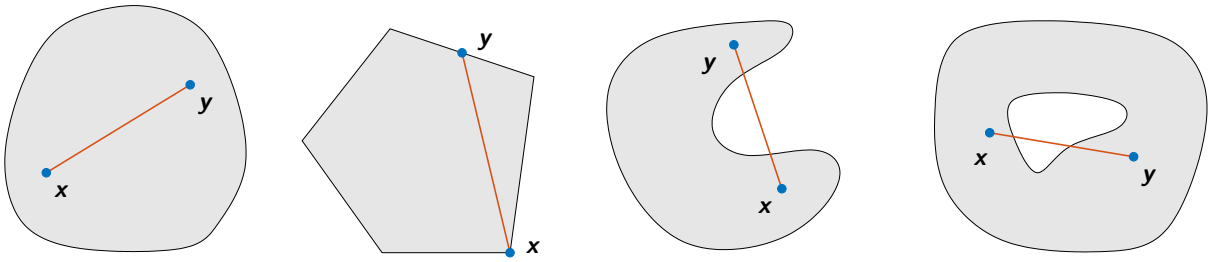
2.3 Convexity

The concept of convexity is of fundamental importance in optimization and in general leads to a simpler numerical solution of the optimization problem. Moreover, convexity is also related to the uniqueness of a solution. Convexity can be defined for sets and functions as well as for optimization problems.

2.3.1 Convex sets

Definition 2.5 (Convex set) *A set $\mathcal{X} \subseteq \mathbb{R}^n$ is* convex, *if the following condition is satisfied for all points $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$*

$$\boldsymbol{z} = k\,\boldsymbol{x} + (1-k)\,\boldsymbol{y} \in \mathcal{X} \quad \textit{for all} \quad k \in [0,1] . \tag{2.13}$$



(a) convex (b) convex (c) non-convex (d) non-convex

Figure 2.3: Examples of convex and non-convex sets.

A geometric interpretation of this definition is that a set $\mathcal{X} \subseteq \mathbb{R}^n$ is convex if and only if the connecting line between any two points $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$ is completely contained in $\mathcal{X}$. Figure 2.3 shows some examples of convex and non-convex sets.

A well known property of convex sets is that their intersection is also a convex set. This property is convenient for characterizing the convexity of the admissible set $\mathcal{X}_{ad}$ of optimization problems (cf. Theorem 2.4).

2.3.2 Convex functions

Definition 2.6 (Convex and concave function) *Let $\mathcal{X} \subseteq \mathbb{R}^n$ be a convex set. A function $f : \mathcal{X} \to \mathbb{R}$ is* convex *on $\mathcal{X}$, if the following condition holds for all points $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$, $\boldsymbol{x} \neq \boldsymbol{y}$ and $\boldsymbol{z} = k\,\boldsymbol{x} + (1-k)\,\boldsymbol{y}$*

$$f(\boldsymbol{z}) \leq k\,f(\boldsymbol{x}) + (1-k)\,f(\boldsymbol{y}) \quad \forall k \in [0,1]\,. \tag{2.14}$$

The function f is strictly convex, *if*

$$f(\boldsymbol{z}) < k\,f(\boldsymbol{x}) + (1-k)\,f(\boldsymbol{y}) \quad \forall k \in (0,1)\,. \tag{2.15}$$

The function f is (strictly) concave, *if $-f$ is (strictly) convex.*

If the function f is convex (concave), then all function values $f(\boldsymbol{z})$ with $\boldsymbol{z} = k\,\boldsymbol{x} + (1-k)\,\boldsymbol{y}$ lie below (above) the line connecting $f(\boldsymbol{x})$ and $f(\boldsymbol{y})$ for any $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$. Figure 2.4 shows some examples of convex and concave functions. Note that linear functions are both concave and convex.

Convex functions have some interesting properties:

a) The weighted sum of convex functions

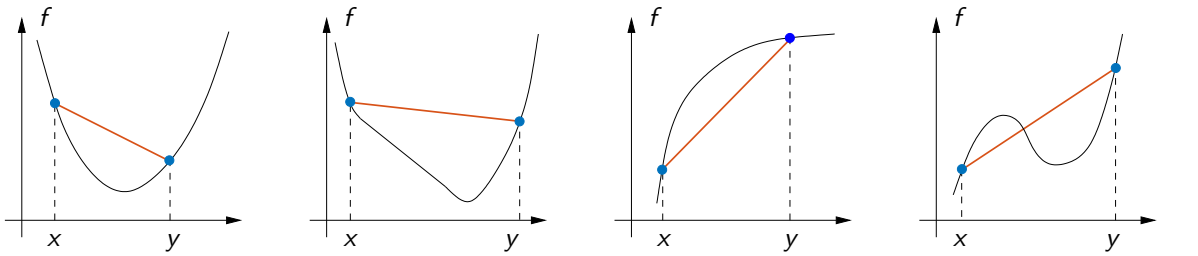
$$f(\boldsymbol{x}) = \sum_{i=1}^{k} a_i\, f_i(\boldsymbol{x}) \tag{2.16}$$

with the coefficients $a_i \geq 0, i = 1 \ldots, k$ is convex.

b) If the function $f(\boldsymbol{x})$ is convex, then the level set

$$\mathcal{S} = \{\boldsymbol{x} \in \mathbb{R}^n : f(\boldsymbol{x}) \leq k\} \tag{2.17}$$

with $k > 0$ is convex, see Figure 2.5.



(a) strictly convex (b) convex (but not strictly) (c) concave (d) neither convex nor concave

Figure 2.4: Examples of convex and concave functions.

c) A continuously differentiable function $f(\boldsymbol{x})$ is convex on $\mathcal{X}$ if the inequality

$$f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + (\boldsymbol{y} - \boldsymbol{x})^{\mathsf{T}} \nabla f(\boldsymbol{x}) \tag{2.18}$$

holds for all $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$. This can be seen as an alternative to Definition 2.6 and is illustrated in Figure 2.6. The geometric interpretation of this inequality is that at every point $\boldsymbol{x}$ of a convex function $f(\boldsymbol{x})$ there exists a *supporting hyperplane* (*supporting tangent* in the scalar case) as a lower bound on $f(\boldsymbol{x})$.

Example 2.3 *Figure 2.7 shows the non-convex function*

$$\min_{\boldsymbol{x} \in \mathbb{R}^2} \; f(\boldsymbol{x}) = -\frac{100}{(x_1 - 1)^2 + (x_2 - 1)^2 + 1} - \frac{200}{(x_1 + 1)^2 + (x_2 + 2)^2 + 1}\,. \tag{2.19}$$

The non-convexity of the function can also be seen from the contour lines in Figure 2.7 that define the non-convex level sets (2.17).

The following theorem characterizes the convexity of a function in terms of the definiteness of its Hessian matrix (also see Theorem 2.2).

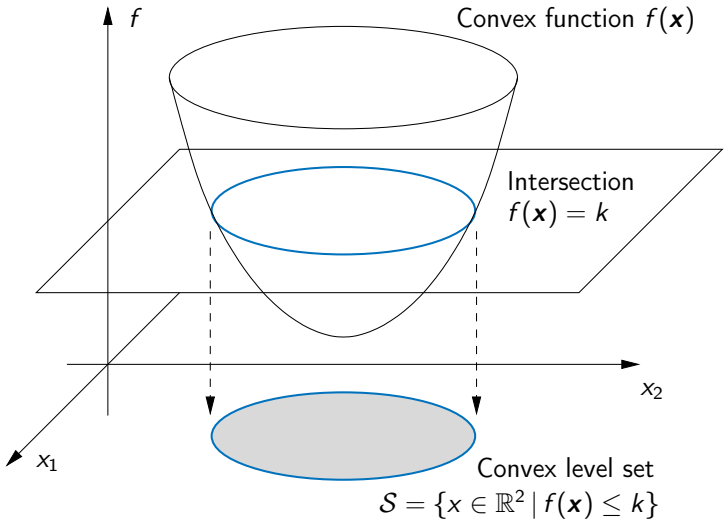


Figure 2.5: Convex level set $\mathcal{S}$ as the intersection of a convex function $f(\boldsymbol{x})$ with the plane $f(\boldsymbol{x}) = const$.

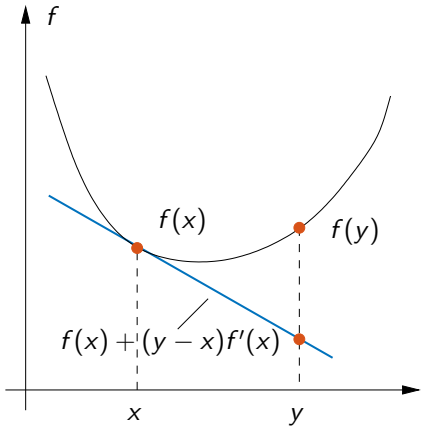


Figure 2.6: Supporting tangent of a convex function $f(x)$.

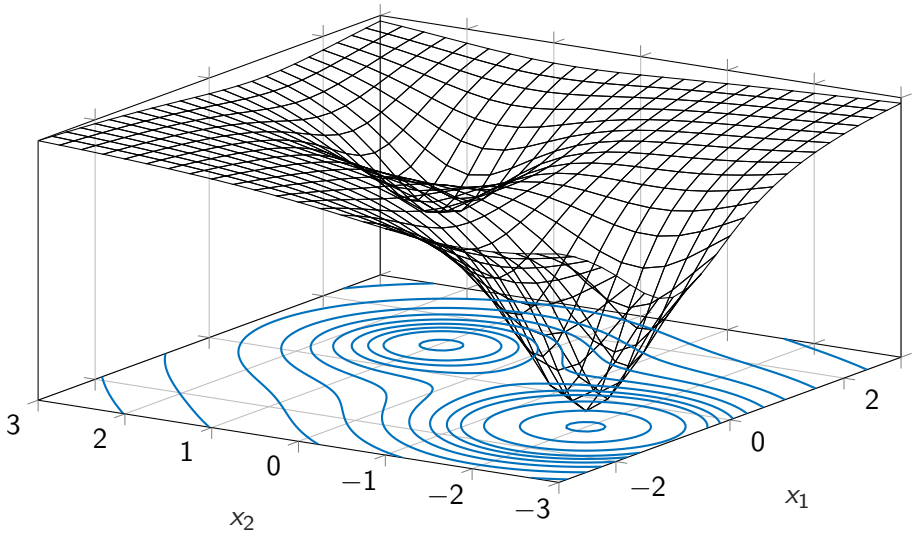


Figure 2.7: Shape and contour lines of the function (2.19).

Theorem 2.3 *Suppose that $\mathcal{X} \subseteq \mathbb{R}^n$ is a convex set and $f : \mathcal{X} \to \mathbb{R}$ is twice continuously differentiable. Then, the following statements are true:*

a) f is convex, *if $\nabla^2 f(\boldsymbol{x})$ is positive semidefinite for all $\boldsymbol{x} \in \mathcal{X}$,*

b) f is strictly convex, *if $\nabla^2 f(\boldsymbol{x})$ is positive definite for all $\boldsymbol{x} \in \mathcal{X}$,*

c) f is concave, *if $\nabla^2 f(\boldsymbol{x})$ is negative semidefinite for all $\boldsymbol{x} \in \mathcal{X}$,*

d) f is strictly concave, *if $\nabla^2 f(\boldsymbol{x})$ is negative definite for all $\boldsymbol{x} \in \mathcal{X}$.*

Note, however, that strict convexity of a function $f(\boldsymbol{x})$ does not necessarily imply the positive definiteness of the Hessian matrix. An example is the strictly convex function $f(x) = x^4$ with the second derivative $f''(0) = 0$ at point $x = 0$.

Exercise 2.1 *Determine the convexity of the following functions by checking the definiteness of the Hessian $\nabla^2 f(\boldsymbol{x})$ using the criteria (1)-(3) in Theorem 2.2:*

$$f(\boldsymbol{x}) = x_1^4 + x_1^2 - 2x_1x_2 + x_2^2\,, \quad \boldsymbol{x} \in \mathbb{R}^2 \tag{2.20}$$

$$f(\boldsymbol{x}) = x_2 + \frac{1}{x_1 x_2}\,, \qquad x_{1,2} > 0. \tag{2.21}$$

Exercise 2.2 *Investigate if the following functions are convex or concave on the given domain and plot the functions (e.g. with* MATLAB*):*

$$f(x) = e^x - 1\,, \quad x \in \mathbb{R} \tag{2.22}$$

$$f(\boldsymbol{x}) = x_1^{\alpha} x_2^{1-\alpha}\,, \quad x_1, x_2 > 0\,, \quad \alpha \in [0, 1]\,. \tag{2.23}$$

2.3.3 Convex optimization problems

The concept of convexity of sets and functions can be extended to optimization problems.

Definition 2.7 *The parameter optimization problem*

$$\min_{\boldsymbol{x}\in\mathcal{X}_{ad}} f(\boldsymbol{x})$$

is (strictly) convex *if the admissible set $\mathcal{X}_{ad}$ is convex and the cost function f is (strictly) convex on $\mathcal{X}_{ad}$.*

A sufficient condition for the convexity of the admissible set $\mathcal{X}_{ad}$ is given in the next theorem.

Theorem 2.4 *Suppose that the functions $g_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \dots, p$ are linear and that the functions $h_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \dots, q$ are convex. Then, the admissible set $\mathcal{X}_{ad}$ in (2.4) is convex.*

Linear and thus convex equality constraints $g_i(\boldsymbol{x}) = 0$ and convex inequality constraints $h_i(\boldsymbol{x}) \leq 0$ always have convex subsets, also see point b) on Page 15 and Figure 2.5. The theorem then directly follows from the already mentioned property that the intersection of convex sets is again a convex set. We can now state a sufficient condition for the convexity of optimization problems.

Theorem 2.5 *The optimization problem*

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & g_i(\boldsymbol{x}) = 0, \quad i = 1, \dots, p \\ & h_i(\boldsymbol{x}) \leq 0, \quad i = 1, \dots, q \end{aligned}$$

is (strictly) convex, *if*

a) the functions $g_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \dots, p$ are linear,

b) the functions $h_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \dots, q$ are convex,

c) the cost function f is (strictly) convex on $\mathcal{X}_{ad} \subseteq \mathbb{R}^n$.

Some important characteristics of convex optimization problems are:

a) Every local minimum is a global minimum.

b) If a strict local minimum exists, then it is unique and global.

c) A strictly convex optimization problem cannot possess more than one solution. In other words, if a local minimum exists, then it is a unique global minimum.

It is worth noting that the strict convexity of an optimization problem does not automatically imply the existence of a solution. To illustrate this point, we again consider the problem

$$\min_{x\in(0,\infty)} \ \frac{1}{x}\,. \tag{2.24}$$

The first and second derivatives of the function $f(x) = \frac{1}{x}$ are $f'(x) = -\frac{1}{x^2}$ and $f''(x) = \frac{2}{x^3}$. Obviously, we have $f''(x) > 0$ for all $x \in \mathcal{X} = (0, \infty)$, which shows that the optimization problem (2.24) is strictly convex. However, as discussed before, the problem has no solution. [1]

The above listed properties a)-c) of convex optimization problems show that convexity simplifies the analysis of optimization problems considerably. In fact, the majority of theoretical results for optimization problems and convergence properties of numerical algorithms are based on this assumption. Since most numerical methods converge to a local minimum, this automatically implies a (unique) global solution in case of (strict) convexity.

Example 2.4 *We again consider Example 2.2. The cost function (2.7) and the constraints (2.8), (2.9) are convex. Thus, the optimization problem (2.7)-(2.9) is convex and therefore the local solution* $\boldsymbol{x}^* = [1, 1]^\mathsf{T}$ *in Figure 1.5 is also the global solution.*

2.4 References

[1] S. Boyd and L. Vandenberghe. *Convex Optimization*. `http://www.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf`. Cambridge: Cambridge University Press, 2004.

[2] J. Nocedal and S.J. Wright. *Numerical Optimization*. New York: Springer, 2006.

[3] M. Papageorgiou, M. Leibold, and M. Buss. *Optimierung*. Berlin: Springer, 2012.

[1] Remember that Weierstrass' Theorem 2.1 only guarantees the existence of a solution if the admissible set $\mathcal{X}$ (or $\mathcal{X}_{ad}$, respectively) is compact.

3 Unconstrained optimization

This chapter addresses unconstrained parameter optimization problems

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x}) \tag{3.1}$$

and derives necessary and sufficient conditions for an optimal solution. Afterwards, the most common optimization algorithms are presented for solving (3.1) numerically.

3.1 Necessary optimality conditions

Optimality conditions are useful to find a (local) extremum point $\boldsymbol{x}^*$ and to verify if it is indeed a minimum for the unconstrained problem (3.1), i.e.

$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}^*) \tag{3.2}$$

for all $\boldsymbol{x}$ in a neighborhood of $\boldsymbol{x}^*$ (see Definition 2.1). Under the assumption that $f(\boldsymbol{x})$ is at least twice continuously differentiable, the function $f(\boldsymbol{x})$ can be expanded in a Taylor series at the point $\boldsymbol{x}^*$

$$f(\boldsymbol{x}^* + \delta\boldsymbol{x}) = f(\boldsymbol{x}^*) + \nabla f(\boldsymbol{x}^*)^\mathsf{T}\delta\boldsymbol{x} + \frac{1}{2}\delta\boldsymbol{x}^\mathsf{T}\nabla^2 f(\boldsymbol{x}^*)\,\delta\boldsymbol{x} + \mathcal{O}(||\delta\boldsymbol{x}||^3)\,. \tag{3.3}$$

If the point $\boldsymbol{x}^*$ is a local minimum, then (3.2) with $\boldsymbol{x} = \boldsymbol{x}^* + \delta\boldsymbol{x}$ must be satisfied for all sufficiently small $\delta\boldsymbol{x}$. Inserting (3.3) into (3.2) leads to the inequality

$$\nabla f(\boldsymbol{x}^*)^\mathsf{T}\delta\boldsymbol{x} + \frac{1}{2}\delta\boldsymbol{x}^\mathsf{T}\nabla^2 f(\boldsymbol{x}^*)\,\delta\boldsymbol{x} + \mathcal{O}(||\delta\boldsymbol{x}||^3) \geq 0\,. \tag{3.4}$$

The first term $\nabla f(\boldsymbol{x}^*)^\mathsf{T}\delta\boldsymbol{x}$ is dominant for sufficiently small $\delta\boldsymbol{x}$ and therefore determines the sign of the left side in (3.4). In other words, $\nabla f(\boldsymbol{x}^*)^\mathsf{T}\delta\boldsymbol{x} \geq 0$ must hold if $\boldsymbol{x}^*$ is a minimum point. Due to the fact that $\boldsymbol{x}^*$ is only a local minimum if the inequality (3.4) is satisfied for all (sufficiently small) $\delta\boldsymbol{x}$, we have

$$\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}\,. \tag{3.5}$$

This is a first-order optimality condition that holds for all extrema, because $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$ is also satisfied by maxima and saddle points, see Figure 3.1. A point $\boldsymbol{x}^*$ satisfying (3.5) is therefore often called *stationary point* and (3.5) is the *stationarity condition*.

A necessary second-order condition for a local minimum $\boldsymbol{x}^*$ follows from (3.4)

$$\delta\boldsymbol{x}^\mathsf{T}\nabla^2 f(\boldsymbol{x}^*)\,\delta\boldsymbol{x} \geq 0\,. \tag{3.6}$$

This condition must be satisfied for all sufficiently small directions $\delta\boldsymbol{x}$ in a neighborhood of $\boldsymbol{x}^*$, which implies that the Hessian $\nabla^2 f(\boldsymbol{x}^*)$ must be positive semidefinite as a consequence of Theorem 2.2. We summarize the necessary optimality conditions for a local minimum in the following lines.

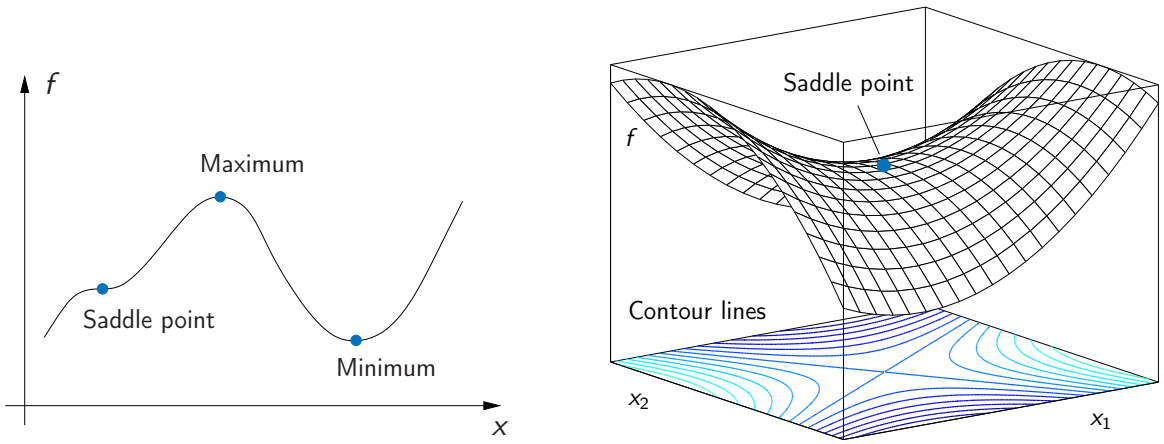


Figure 3.1: Examples of stationary points in $\mathbb{R}^1$ and $\mathbb{R}^2$.

Theorem 3.1 (Necessary optimality conditions) *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable and that $\boldsymbol{x}^*$ is a local minimum point of f. Then, the following first- and second-order optimality conditions hold*

$$\nabla f(\boldsymbol{x}^*) = \mathbf{0} \tag{3.7}$$

$$\nabla^2 f(\boldsymbol{x}^*) \geq \mathbf{0} \quad \textit{(positive semidefinite)}. \tag{3.8}$$

Note that the optimality conditions are only necessary but not sufficient. An example is the function $f(x) = x^3$ that has an extremum at $x^* = 0$, which is easily verified by the first derivative $f'(x) = 3x^2$. Although the second derivative $f''(x^*) = 6x^* = 0$ is positive semidefinite, $x^* = 0$ is not a minimum point but a saddle point, see Figure 3.1.

3.2 Sufficient optimality conditions

A sufficient condition that guarantees the existence of a local minimum $\boldsymbol{x}^*$ can be extracted from the inequality (3.4). If $\nabla f(\boldsymbol{x}^*) = \mathbf{0}$ holds and the quadratic term in (3.4) is strictly positive, i.e.

$$\delta \boldsymbol{x}^\mathsf{T} \nabla^2 f(\boldsymbol{x}^*)\, \delta \boldsymbol{x} > 0\,, \tag{3.9}$$

then the inequality (3.4) is strictly satisfied for all sufficiently small $\delta \boldsymbol{x}$ and therefore $\boldsymbol{x}^*$ is a strict local minimum.

Theorem 3.2 (Sufficient optimality conditions) *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a twice continuously differentiable function. If the following conditions are satisfied at some point $\boldsymbol{x}^*$*

$$\nabla f(\boldsymbol{x}^*) = \mathbf{0} \tag{3.10}$$

$$\nabla^2 f(\boldsymbol{x}^*) > \mathbf{0} \quad \textit{(positive definite)}, \tag{3.11}$$

then f has a strict local minimum at $\boldsymbol{x}^*$.

Note that a strict minimum point $\boldsymbol{x}^*$ does not automatically imply the positive definiteness of the Hessian $\nabla^2 f(\boldsymbol{x}^*)$. An example is the function $f(x) = x^4$, which obviously has a strict minimum at the point $x^* = 0$. However, the second derivative $f''(x) = 12x^2$ vanishes at the position $x^* = 0$ and is thus not positive definite.

The stationarity condition $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$ is a nonlinear system of equations of order n that can be solved for a stationary point $\boldsymbol{x}^* \in \mathbb{R}^n$. Afterwards, the positive definiteness of the Hessian matrix $\nabla^2 f(\boldsymbol{x}^*)$ can be analyzed using one of the criteria in Theorem 2.2 to investigate whether $f(\boldsymbol{x}^*)$ is a strict minimum.

Example 3.1 *Consider the optimization problem*

$$\min_{\boldsymbol{x}\in\mathbb{R}^2} \; x_1^2 + a x_2^2 - x_1 x_2 \tag{3.12}$$

with the parameter $a \in \mathbb{R}$. To characterize the stationary points $\boldsymbol{x}^*$, we compute the gradient and the Hessian

$$\nabla f(\boldsymbol{x}) = \begin{bmatrix} 2x_1 - x_2 \\ 2ax_2 - x_1 \end{bmatrix}, \quad \nabla^2 f(\boldsymbol{x}) = \begin{bmatrix} 2 & -1 \\ -1 & 2a \end{bmatrix}. \tag{3.13}$$

From $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$ we get $\boldsymbol{x}^* = [0,0]^\mathrm{T}$ as the unique stationary point for $a \neq \frac{1}{4}$. The definiteness of the Hessian $\nabla^2 f(\boldsymbol{x}^*)$ is checked via the principle minors, see Criterion (3) in Theorem 2.2,

$$D_1 = 2, \quad D_2 = 4a - 1\,. \tag{3.14}$$

This shows that $\nabla^2 f(\boldsymbol{x}^*)$ is positive definite for $a > \frac{1}{4}$, in which case $\boldsymbol{x}^* = [0,0]^\mathrm{T}$ is a strict minimum point. For $a < \frac{1}{4}$, we get $D_1 > 0$ and $D_2 < 0$ for which Sylvester's criterion does not apply. In fact, $\nabla^2 f(\boldsymbol{x})$ is indefinite for $a = -1$ and $\boldsymbol{x}^* = [0,0]^\mathrm{T}$ is a saddle point as shown in Figure 3.1b).

If the cost function $f(\boldsymbol{x})$ is convex or even strictly convex, we can obtain stronger results than in Theorem 3.1 and 3.2:

Theorem 3.3 (Sufficient optimality conditions for convex functions) *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable and (strictly) convex function. If $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$, then the function f has a (unique) global minimum at $\boldsymbol{x}^*$.*

3.3 Line search methods

The stationarity condition $\nabla f(\boldsymbol{x}^*) = \boldsymbol{0}$ cannot be solved analytically in most cases and numerical methods must be used instead to determine a stationary point $\boldsymbol{x}^*$ in an iterative manner. A widely used class of numerical algorithms are *line search methods*. Starting from a suboptimal initial solution $\boldsymbol{x}^0$ with $f(\boldsymbol{x}^0) > f(\boldsymbol{x}^*)$, line search methods approach the minimum point $\boldsymbol{x}^*$ iteratively by enforcing

$$f(\boldsymbol{x}^{k+1}) < f(\boldsymbol{x}^k) \tag{3.15}$$

in each optimization step k, such that the minimum point $\boldsymbol{x}^*$ is reached in the limit

$$\lim_{k\to\infty} \boldsymbol{x}^k = \boldsymbol{x}^*\,. \tag{3.16}$$

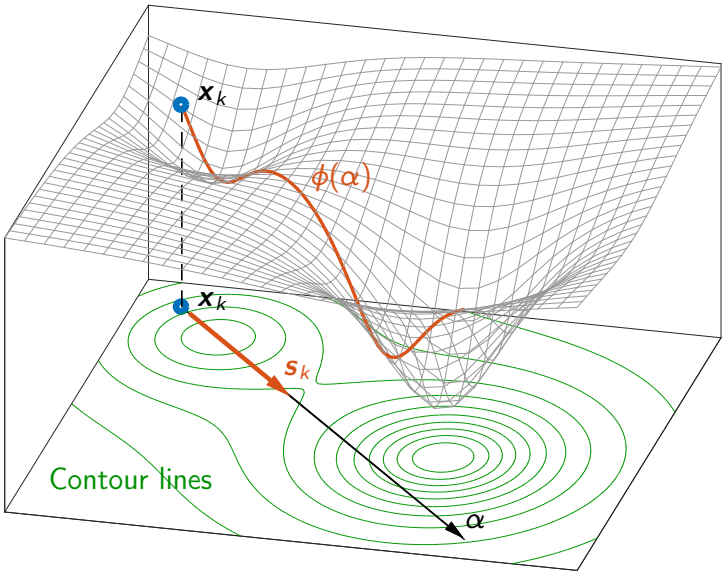


Figure 3.2: Illustration of line search problem (3.20).

All line search methods determine a *search direction* (or *descent direction*) $\boldsymbol{s}^k$, in which the inequality (3.15) is satisfied. To make this more precise, we consider the Taylor expansion of the cost function $f(\boldsymbol{x})$ at the point $\boldsymbol{x}^k$

$$f(\boldsymbol{x}^k + \boldsymbol{s}) = f(\boldsymbol{x}^k) + \nabla f(\boldsymbol{x}^k)^\mathsf{T} \boldsymbol{s} + \mathcal{O}(||\boldsymbol{s}||^2)\,. \tag{3.17}$$

The higher-order terms $\mathcal{O}(||\boldsymbol{s}||^2)$ can be neglected for sufficiently small $\boldsymbol{s}$ and we see that a suitable search direction $\boldsymbol{s}^k$ at the current iterate $\boldsymbol{x}^k$ must satisfy the *descent condition*

$$\nabla f(\boldsymbol{x}^k)^\mathsf{T} \boldsymbol{s}^k < 0 \tag{3.18}$$

in order for $f(\boldsymbol{x}^k + \boldsymbol{s}^k) < f(\boldsymbol{x}^k)$ to hold. There exist different methods for computing the search direction $\boldsymbol{s}^k$. The most common ones are introduced in Section 3.3.2 to 3.3.5.

After the search direction $\boldsymbol{s}^k$ is computed, a suitable step size α^k must be chosen to determine how far to go in direction of $\boldsymbol{s}^k$, i.e.

$$\boxed{\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha^k \boldsymbol{s}^k\,.} \tag{3.19}$$

The optimal step size α^k is the solution of the (scalar) optimization problem

$$\boxed{\alpha^k = \arg\min_{\alpha > 0} \; \phi(\alpha) \quad \text{with} \quad \phi(\alpha) = f(\boldsymbol{x}^k + \alpha \boldsymbol{s}^k)\,,} \tag{3.20}$$

which corresponds to the maximum possible descent in the search direction $\boldsymbol{s}^k$. Figure 3.2 illustrates the idea behind the line search for a (non-convex) cost function $f(\boldsymbol{x})$ with $\boldsymbol{x} \in \mathbb{R}^2$ and a given descent direction $\boldsymbol{s}^k$.

The line search problem (3.20) is usually not solved exactly but in an approximate manner, see Section 3.3.1. The computed value of α^k then defines the step size in the k-th iteration and determines the next point (3.19). The basic line search algorithm is summarized in Table 3.1.

Initialization:	$\boldsymbol{x}^0$	(Initial guess)
	$k \leftarrow 0$	(Iteration index)
	$\varepsilon_x, \varepsilon_f$	(Convergence tolerance)
repeat		
	Compute search direction $\boldsymbol{s}^k$	
	Compute step size α^k	
	$\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x}^k + \alpha^k \boldsymbol{s}^k$	
	$k \leftarrow k+1$	
until	$\|\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\|\| \leq \varepsilon_x$ **or** $\|\|f(\boldsymbol{x}^k) - f(\boldsymbol{x}^{k-1})\|\| \leq \varepsilon_f$	

Table 3.1: General structure of line search methods.

3.3.1 Step size computation

The calculation of a suitable step size α^k represents a compromise between computation time and accuracy. The optimal solution would be to solve the scalar optimization problem (3.20) exactly or at least sufficiently accurate. In practice, however, this would require too many evaluations of the cost function f and possibly of the gradient ∇f. Most numerical methods therefore use a suboptimal line search to determine a step size α^k that leads to an adequate reduction of the cost function f in the next step $\boldsymbol{x}^{k+1}$.

Decrease and curvature conditions (Wolfe conditions)

A first criterion for a sufficient decrease of the cost function f is the inequality

$$\underbrace{f(\boldsymbol{x}^k + \alpha^k \boldsymbol{s}^k)}_{=\phi(\alpha^k)} \leq \underbrace{f(\boldsymbol{x}^k)}_{=\phi(0)} + c_1 \alpha^k \underbrace{\nabla f(\boldsymbol{x}^k)^\mathsf{T} \boldsymbol{s}^k}_{=\phi'(0)} \tag{3.21}$$

with the constant $c_1 \in (0,1)$. The sufficient decrease condition is sometimes called *Armijo condition*. The right side of the inequality (3.21) represents a straight line with negative slope $c_1 \phi'(0)$ that is guaranteed to lie above the function value $\phi(\alpha^k)$ for sufficiently small α^k and $c_1 \in (0,1)$. Thus, the sufficient decrease condition (3.21) declares a step size α^k valid only if the function value $\phi(\alpha^k)$ lies below this straight line. Figure 3.3 gives a graphical illustration of the sufficient decrease condition. In practice, c_1 is typically chosen to be very small (e.g. 10^{-4}) to make the decrease condition not too restrictive.

The Armijo condition (3.21) alone is often not sufficient to guarantee a noteworthy decrease of the cost function f as the condition is always satisfied for sufficiently small step sizes α^k, see Figure 3.3. Unnecessarily small values of α^k can be avoided by adding the so-called *curvature condition*[1]

$$\Big|\underbrace{\nabla f(\boldsymbol{x}^k + \alpha^k \boldsymbol{s}^k)^\mathsf{T} \boldsymbol{s}^k}_{=\phi'(\alpha^k)}\Big| \leq c_2 \Big|\underbrace{\nabla f(\boldsymbol{x}^k)^\mathsf{T} \boldsymbol{s}^k}_{=\phi'(0)}\Big| . \tag{3.22}$$

The constant $c_2 \in (c_1, 1)$ is chosen w.r.t. the first constant c_1 from (3.21). Since the left side of the curvature condition represents the gradient $\phi'(\alpha^k)$, the criterion implies that the

[1] Strictly speaking, (3.22) is a *strengthened* curvature condition. A weaker form of (3.22) can be stated without taking the absolute values on both sides.

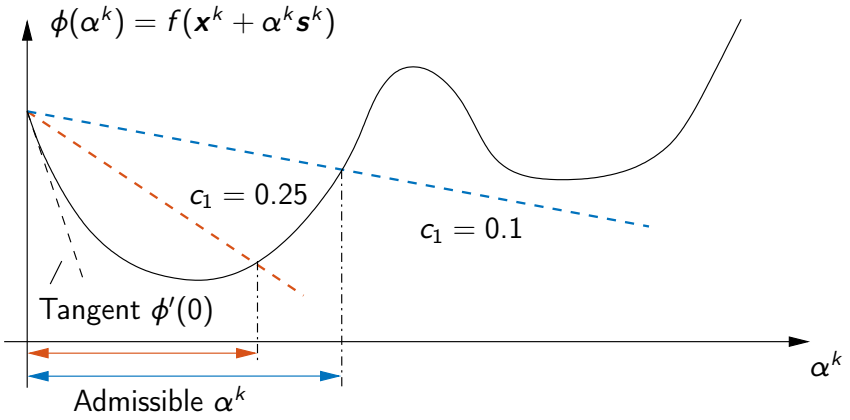


Figure 3.3: Illustration of the sufficient decrease condition.

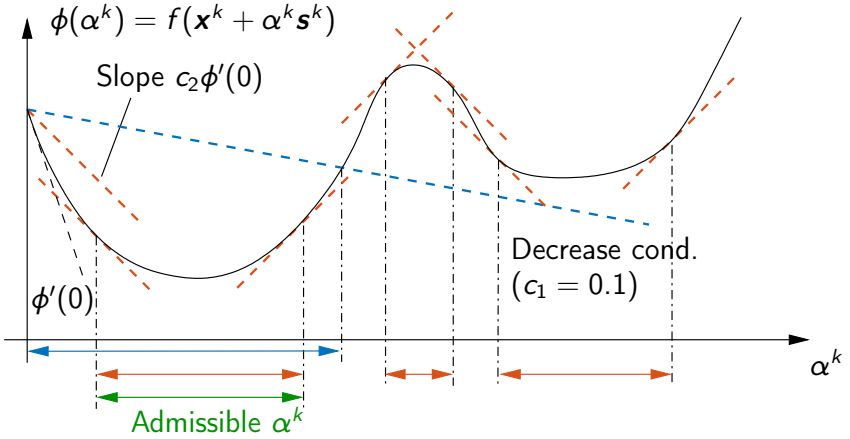


Figure 3.4: Illustration of the Wolfe conditions.

absolute value of the gradient $\phi'(\alpha^k)$ at point α^k is smaller or equal to c_2 times the absolute initial slope $\phi'(0)$, see Figure 3.4.

The formulation of the curvature condition indeed makes sense, since in the case of a strongly negative gradient $\phi'(\alpha^k)$ it can be assumed that ϕ and thus f can be further reduced by moving further along the α^k-axis. On the other hand, the curvature condition does not allow the slope $\phi'(\alpha^k)$ to become too large which indicates that one moves away from the local minimum of ϕ.

Both conditions together, (3.21) and (3.22), are also called *Wolfe conditions* and can be used together with most of the line search methods. Typically, c_2 is set to $c_2 = 0.9$ if Newton- or quasi-Newton method are used to compute the search direction $\boldsymbol{s}^k$. In case of the conjugate gradient method, $c_2 = 0.1$ is a standard choice. These methods are outlined in the following sections. It can be shown for continuously differentiable and lower bounded functions f that there always exists an interval where the Wolfe conditions are satisfied.

Backtracking

As already mentioned, the descent condition (3.21) alone is not sufficient to guarantee an adequate descent in the cost function f. However, if the step size candidates α^k are chosen appropriately, the curvature condition (3.22) can be omitted and the line search can be stopped based on the descent condition (3.21) alone. A suitable method in this regard is the *backtracking* algorithm, which in its simplest form is given in Table 3.2.

Initialization:	$\alpha \leftarrow \bar{\alpha} > 0$	(Initial step size)
	$\rho \in (0,1)$	(Backtracking parameter)
	c	(Armijo parameter)
repeat		
	$\alpha \leftarrow \rho\alpha$	
until	$f(\boldsymbol{x}^k + \alpha\boldsymbol{s}^k) \leq f(\boldsymbol{x}^k) + c\,\alpha\nabla f(\boldsymbol{x}^k)^\mathsf{T}\boldsymbol{s}^k$	
$\alpha^k \leftarrow \alpha$	(Step size)	

Table 3.2: Basic structure of the backtracking algorithm.

The procedure can be illustrated using Figure 3.3. If the initial value $\bar{\alpha}$ is far to the right, α is reduced by the factor ρ until $\phi(\alpha) = f(\boldsymbol{x}^k + \alpha\boldsymbol{s}^k)$ lies below the straight line (with $c = 0.1$ or 0.25). The successive reduction by factor ρ ensures that α^k becomes small enough to satisfy the descent condition (3.21) but *does not become too small*, which could lead to a stagnation of the process.

The backtracking procedure is often used together with the Newton method and the "natural" initial step size $\bar{\alpha} = 1$, see Section 3.3.4. However, backtracking is less appropriate for quasi-Newton or conjugate gradient methods, see Sections 3.3.3 and 3.3.5.

Golden section search

An alternative method for finding the step size α^k is the *golden section search* that generates a convergent series of interval reductions. The first step consists in finding an interval $[\alpha_l, \alpha_r]$ that contains a minimum of the function $\phi(\alpha)$, see Figure 3.5. This can be done, for instance, by starting with a sufficiently small α_l and successively increasing α_r from α_l onward until the function value $\phi(\alpha_r)$ starts to increase.

In the next step, two inner points $\alpha_l < \hat{\alpha}_l < \hat{\alpha}_r < \alpha_r$ are chosen as

$$\hat{\alpha}_l = \alpha_l + (1-a)(\alpha_r - \alpha_l), \quad \hat{\alpha}_r = \alpha_l + a(\alpha_r - \alpha_l) \tag{3.23}$$

with the parameter $a \in (\frac{1}{2}, 1)$. If $\phi(\hat{\alpha}_l) < \phi(\hat{\alpha}_r)$, the minimum lies in the interval $[\alpha_l, \hat{\alpha}_r]$ and the outer right point α_r is discarded, see "Step 1" in Figure 3.5. If $\phi(\hat{\alpha}_l) > \phi(\hat{\alpha}_r)$ instead, the minimum must lie in the interval $[\hat{\alpha}_l, \alpha_r]$ and the outer left point α_l is omitted. The next

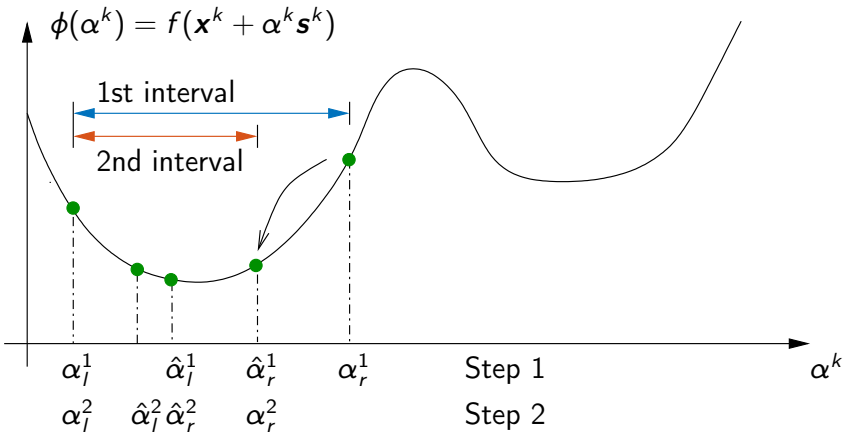


Figure 3.5: Illustration of the golden section search.

Initialization:	α_l, α_r	(Initial interval containing minimum)
	$a = 0.618$	(Golden ratio parameter)
	$\varepsilon_\alpha > 0$	(Convergence tolerance)
	$\hat{\alpha}_l \leftarrow \alpha_l + (1-a)(\alpha_r - \alpha_l)$	(Inner points)
	$\hat{\alpha}_r \leftarrow \alpha_l + a(\alpha_r - \alpha_l)$	
repeat		
	if $\phi(\hat{\alpha}_l) < \phi(\hat{\alpha}_r)$ **do**	
	$\alpha_r \leftarrow \hat{\alpha}_r$	
	$\hat{\alpha}_r \leftarrow \hat{\alpha}_l$	
	$\hat{\alpha}_l \leftarrow \alpha_l + (1-a)(\alpha_r - \alpha_l)$	
	else (i.e.. $\phi(\hat{\alpha}_l) \geq \phi(\hat{\alpha}_r)$)	
	$\alpha_l \leftarrow \hat{\alpha}_l$	
	$\hat{\alpha}_l \leftarrow \hat{\alpha}_r$	
	$\hat{\alpha}_r \leftarrow \alpha_l + a(\alpha_r - \alpha_l)$	
	end	
until	$\lvert\alpha_r - \alpha_l\rvert \leq \varepsilon_\alpha$	

Table 3.3: Golden section search.

iteration then considers the shortened interval $[\alpha_l, \hat{\alpha}_r]$ or $[\hat{\alpha}_l, \alpha_r]$ and the procedure starts again.

If the parameter a is chosen as

$$a = \frac{\sqrt{5}-1}{2} \approx 0.618\,, \tag{3.24}$$

then only one function evaluation is necessary to determine a new interior point, see "Step 2" with $\hat{\alpha}_r^2 = \hat{\alpha}_l^1$ in Figure 3.5. This choice of a is known as the *golden ratio* and also gives the *golden section search* its name. In each step of the algorithm, the interval is reduced by approx. 38%. Table 3.3 summarizes the algorithm.

When the golden section search is stopped, the step size α^k can either be computed from the mean $\alpha^k = (\alpha_l + \alpha_r)/2$ or from a quadratic interpolation between the smallest three function values ϕ at the positions $(\alpha_l, \hat{\alpha}_l, \hat{\alpha}_r, \alpha_r)$. The golden section search is a simple and robust method to compute a suitable step size α^k, but may require more iterations to converge than more sophisticated algorithms.

3.3.2 Gradient method

The computation of a suitable search direction $\boldsymbol{s}^k$ is of fundamental importance in line search methods and comes prior to the line search problem (3.20) that eventually leads to the next iterate

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha^k \boldsymbol{s}^k \tag{3.25}$$

with a meaningful descent in the cost f. Various methods exist to compute the search direction $\boldsymbol{s}^k$ that differ in terms of accuracy, convergence speed, and computational effort.

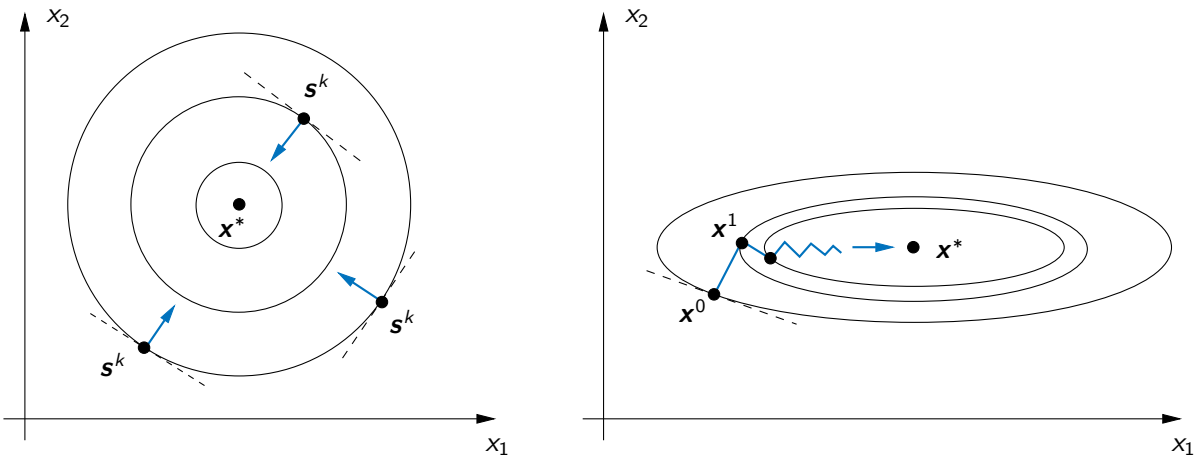


Figure 3.6: Well- and ill-conditioned problems (left/right) for the gradient method.

The simplest choice of $\boldsymbol{s}^k$ is to use the negative gradient at the point $\boldsymbol{x}^k$

$$\boxed{\boldsymbol{s}^k = -\nabla f(\boldsymbol{x}^k)\,.} \tag{3.26}$$

It is easy to see that the descent condition (3.18) is satisfied for non-stationary points (i.e. $\nabla f(\boldsymbol{x}^k) \neq \boldsymbol{0}$)

$$-\nabla f(\boldsymbol{x}^k)^\mathsf{T} \nabla f(\boldsymbol{x}^k) = -||\nabla f(\boldsymbol{x}^k)||^2 < 0\,. \tag{3.27}$$

In fact, the negative gradient minimizes the inequality (3.18) and is therefore also called direction of *steepest descent*. Geometrically interpreted, the gradient direction is orthogonal to the contour line $f(\boldsymbol{x}^k) = const.$ that passes through the point $\boldsymbol{x}^k$, see Figure 3.6.

A problem is well-conditioned if the negative gradient points towards the minimum $f(\boldsymbol{x}^*)$, see Figure 3.6 (left). In this case, the gradient method converges rapidly to the optimal solution. However, if the problem is ill-conditioned in the sense that the negative gradient does not, at least roughly, point in the direction of $\boldsymbol{x}^*$, then the gradient method typically converges very slowly as illustrated in Figure 3.6 (right). The next theorem makes this more precise for quadratic cost functions.

Theorem 3.4 (Convergence rate of gradient method) *For quadratic functions $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\mathsf{T}\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{b}^\mathsf{T}\boldsymbol{x} + c$ with positive definite matrix $\boldsymbol{Q}$, the gradient method with exact line search (3.20) converges according to*

$$f(\boldsymbol{x}^{k+1}) - f(\boldsymbol{x}^*) \leq \left(\frac{\kappa - 1}{\kappa + 1}\right)^2 \Big(f(\boldsymbol{x}^k) - f(\boldsymbol{x}^*)\Big)\,, \quad \kappa = \frac{\lambda_{\max}}{\lambda_{\min}} \tag{3.28}$$

with $\lambda_{\min}$ and $\lambda_{\max}$ being the smallest and largest eigenvalues of $\boldsymbol{Q}$.

The number κ is the spectral condition number of the matrix $\boldsymbol{Q}$ and directly affects the speed of convergence as demonstrated in the following exercise.

Exercise 3.1 *Determine the convergence rate (3.28) of the gradient method for the quadratic problem*

$$\min_{\boldsymbol{x}\in\mathbb{R}^2} \quad f(\boldsymbol{x}) = \frac{1}{2}\left(\frac{x_1 - x_1^*}{a}\right)^2 + \frac{1}{2}(x_2 - x_2^*)^2 \tag{3.29}$$

with the values $a = 1$ and $a = 5$, also see Figure 3.6.

The advantages and disadvantages of the gradient methods can be summarized as follows:

+ larger domain of convergence than in case of the Newton method (positive definiteness of the Hessian $\nabla^2 f(\boldsymbol{x}^k) > \boldsymbol{0}$ is not required)
+ simple and robust
- slow convergence for ill-conditioned or badly scaled problems
- limited accuracy (first-order method).

3.3.3 Conjugate gradient method

The *conjugate gradient method* is an important alternative to the conventional gradient method due to the fact that the computational effort is not much higher but the efficiency and convergence speed are significantly better. The conjugate gradient method uses information from the previous iteration to determine the search direction

$$\boxed{\begin{aligned} \boldsymbol{s}^0 &= -\nabla f(\boldsymbol{x}^0) && \text{(3.30a)} \\ \boldsymbol{s}^k &= -\nabla f(\boldsymbol{x}^k) + \beta^k \boldsymbol{s}^{k-1}\,. && \text{(3.30b)} \end{aligned}}$$

The conjugate gradient method is initialized in steepest descent direction $-\nabla f(\boldsymbol{x}^0)$, whereas all further steps contain a correction term based on the previous search direction $\boldsymbol{s}^{k-1}$. The factor β^k is chosen in dependence of the gradient norms of the current and last iteration

$$\boxed{\beta^k = \frac{||\nabla f(\boldsymbol{x}^k)||^2}{||\nabla f(\boldsymbol{x}^{k-1})||^2}\,.} \tag{3.31}$$

This formula is due to *Fletcher* and *Reeves*, others can be found in the textbooks at the end of the chapter.

Conjugate gradient methods do not require any complex matrix operations and can therefore be applied to large problems, where memory efficiency must be taken into account. Interestingly, it can be shown for quadratic problems

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\mathsf{T}\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{b}^\mathsf{T}\boldsymbol{x} + c\,, \quad \boldsymbol{x} \in \mathbb{R}^n$$

with positive definite Hessian $\boldsymbol{Q}$ that the conjugate gradient method with exact line search (3.20) converges in at most n iterations irrespective of the conditioning of $\boldsymbol{Q}$.

Exercise 3.2 *Verify that the conjugate gradient method with exact line search produces a search direction $\boldsymbol{s}^k$ that satisfies the descent condition (3.18).*

3.3.4 Newton's method

A special case of the line search method occurs if the search direction $\boldsymbol{s}^k$ is chosen such that the iteration rule $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha^k \boldsymbol{s}^k$ corresponds to the Newton method. In this case we speak of the *Newton direction* $\boldsymbol{s}^k$. To get to this point, we approximate the cost function $f(\boldsymbol{x}) = f(\boldsymbol{x}^k + \boldsymbol{s})$ at the point $\boldsymbol{x}^k$ by the quadratic model

$$m^k(\boldsymbol{s}) = f(\boldsymbol{x}^k) + \nabla f(\boldsymbol{x}^k)^\mathsf{T}\boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^\mathsf{T}\nabla^2 f(\boldsymbol{x}^k)\,\boldsymbol{s} \tag{3.32}$$

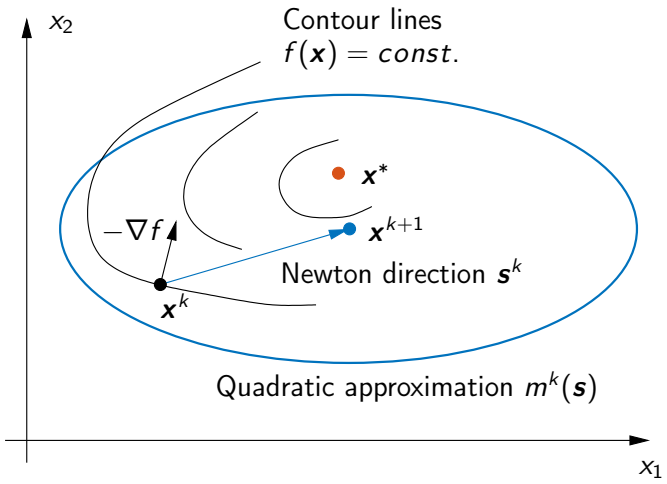


Figure 3.7: Illustration of Newton's method in comparison to the gradient method.

and compute the minimum point $\boldsymbol{s}$ from the stationarity condition

$$\nabla f(\boldsymbol{x}^k) + \nabla^2 f(\boldsymbol{x}^k)\, \boldsymbol{s} = \boldsymbol{0}\,. \tag{3.33}$$

If we assume that the Hessian matrix $\nabla^2 f(\boldsymbol{x}^k)$ is positive definite, the Newton direction $\boldsymbol{s}^k = \boldsymbol{s}$ at the point $\boldsymbol{x}^k$ becomes

$$\boxed{\boldsymbol{s}^k = -\left[\nabla^2 f(\boldsymbol{x}^k)\right]^{-1} \nabla f(\boldsymbol{x}^k)\,.} \tag{3.34}$$

It is easy to verify that the descent condition (3.18)

$$\nabla f(\boldsymbol{x}^k)^\mathsf{T} \boldsymbol{s}^k = -\nabla f(\boldsymbol{x}^k)^\mathsf{T} \left[\nabla^2 f(\boldsymbol{x}^k)\right]^{-1} \nabla f(\boldsymbol{x}^k) < 0 \tag{3.35}$$

is always satisfied since the inverse of a positive definite matrix $\nabla^2 f(\boldsymbol{x}^k)$ is also positive definite.

The Newton direction (3.34) defines the next point $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \boldsymbol{s}^k$, which corresponds to the "natural" step size $\alpha^k = 1$ for the iteration rule (3.25) of the line search method. Therefore, Newton methods typically use a default step size of $\alpha^k = 1$ that is only reduced by backtracking if the sufficient decrease condition for the cost function $f(\boldsymbol{x}^{k+1})$ is violated.

Newton's method is a second-order method, since the error of the approximation (3.32) is of order $\mathcal{O}(||\boldsymbol{s}||^3)$. It is reliable and also well suited for high-dimensional systems, if the quadratic approximation of the function f at the point $\boldsymbol{x}^k$ is sufficiently accurate. Figure 3.7 illustrates the advantage of the quadratic approximation of the Newton method compared to the steepest descent direction $-\nabla f$ of the gradient method.

Exercise 3.3 *Show that for quadratic problems*

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \; \frac{1}{2}\boldsymbol{x}^\mathsf{T}\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{b}^\mathsf{T}\boldsymbol{x} + c \qquad (\boldsymbol{Q} \textit{ positive definite}) \tag{3.36}$$

the Newton method converges within one iteration independent of the initial point $\boldsymbol{x}^0$.

The advantages and disadvantages of Newton's method are

+ quadratic convergence in the neighborhood of $\boldsymbol{x}^*$ (within few iterations)

+ high accuracy (second-order method)

- only local convergence (positive definiteness of $\nabla^2 f(\boldsymbol{x})$ usually not guaranteed for all $\boldsymbol{x}$)

- expensive computation of Hessian $\nabla^2 f$ in particular for large-scale problems.

The local convergence property of Newton methods can be globalized by keeping the Hessian matrix $\nabla^2 f(\boldsymbol{x})$ positive definite, e.g. by distorting its diagonal. Moreover, the complex computation of the Hessian matrix $\nabla^2 f$ can be mitigated by sparse numerics or finite-difference schemes or even completely avoided by the quasi-Newton method that is discussed in the following lines.

3.3.5 Quasi-Newton method

The *quasi-Newton method* is an attractive alternative to Newton's method, since it avoids the time-consuming calculation of the Hessian matrix $\nabla^2 f$ and its inverse. Instead, the Hessian is approximated and updated during the individual iterations by exploiting information about the gradients at the single iteration points.

To illustrate this idea, we approximate the gradient ∇f at the point $\boldsymbol{x}^k$ by the linear model

$$\nabla f(\boldsymbol{x}^k + \boldsymbol{s}) \approx \nabla f(\boldsymbol{x}^k) + \nabla^2 f(\boldsymbol{x}^k)\boldsymbol{s}\,. \tag{3.37}$$

For sufficiently small $\boldsymbol{s} = \boldsymbol{x}^{k-1} - \boldsymbol{x}^k$, we obtain an approximation for the gradient at the previous point $\boldsymbol{x}^{k-1}$

$$\nabla f(\boldsymbol{x}^{k-1}) \approx \nabla f(\boldsymbol{x}^k) + \nabla^2 f(\boldsymbol{x}^k)(\boldsymbol{x}^{k-1} - \boldsymbol{x}^k)\,. \tag{3.38}$$

This relation shows that the Hessian at point $\boldsymbol{x}^k$ can be approximated by a matrix $\boldsymbol{H}^k \approx \nabla^2 f(\boldsymbol{x}^k)$ that must satisfy the so-called *secant equation*

$$\boldsymbol{H}^k(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}) = \nabla f(\boldsymbol{x}^k) - \nabla f(\boldsymbol{x}^{k-1})\,. \tag{3.39}$$

Since the problem is underdetermined ($\boldsymbol{H}^k$ has n^2 elements, but (3.39) only n equations), there are typically further conditions that are imposed on $\boldsymbol{H}^k$ such as symmetry and minimal difference between $\boldsymbol{H}^k$ and $\boldsymbol{H}^{k-1}$. These additional conditions are combined with (3.39) in an optimization problem for $\boldsymbol{H}^k$, which can be solved analytically.

A popular formula for updating $\boldsymbol{H}^k$ is the *BFGS formula* due to *Broyden*, *Fletcher*, *Goldfarb*, and *Shanno*

$$\boldsymbol{H}^k = \boldsymbol{H}^{k-1} - \frac{\boldsymbol{H}^{k-1}\boldsymbol{d}^k(\boldsymbol{d}^k)^\mathsf{T}\boldsymbol{H}^{k-1}}{(\boldsymbol{d}^k)^\mathsf{T}\boldsymbol{H}^{k-1}\boldsymbol{d}^k} + \frac{\boldsymbol{y}^k(\boldsymbol{y}^k)^\mathsf{T}}{(\boldsymbol{d}^k)^\mathsf{T}\boldsymbol{y}^k} \tag{3.40}$$

with the vectors

$$\boldsymbol{d}^k = \boldsymbol{x}^k - \boldsymbol{x}^{k-1}, \quad \boldsymbol{y}^k = \nabla f(\boldsymbol{x}^k) - \nabla f(\boldsymbol{x}^{k-1})\,. \tag{3.41}$$

The initialization of the quasi-Newon method at point $\boldsymbol{x}^0$ is either done with a positive definite estimate $\boldsymbol{H}^0 \approx \nabla^2 f(\boldsymbol{x}^0)$ or in the most simple case by the unit matrix $\boldsymbol{H}^0 = \boldsymbol{I}$.

A drawback of the BFGS formula (3.40) is that Newton's method actually requires the inverse Hessian matrix to calculate the search direction $\boldsymbol{s}^k$, see (3.34). It is therefore more reasonable

to directly approximate the inverse Hessian matrix $\boldsymbol{B}^k = (\boldsymbol{H}^k)^{-1}$. The respective BFGS formula (3.40) for updating the inverse Hessian is

$$\boxed{\boldsymbol{B}^k = \left[\boldsymbol{I} - \rho^k \boldsymbol{d}^k (\boldsymbol{y}^k)^\mathsf{T}\right] \boldsymbol{B}^{k-1} \left[\boldsymbol{I} - \rho^k \boldsymbol{y}^k (\boldsymbol{d}^k)^\mathsf{T}\right] + \rho^k \boldsymbol{d}^k (\boldsymbol{d}^k)^\mathsf{T}, \quad \rho^k = \frac{1}{(\boldsymbol{y}^k)^\mathsf{T} \boldsymbol{d}^k}\,.} \tag{3.42}$$

Using $\boldsymbol{B}^k$ in the Newton direction (3.34) eventually gives the search direction

$$\boxed{\boldsymbol{s}^k = -\boldsymbol{B}^k \nabla f(\boldsymbol{x}^k)\,.} \tag{3.43}$$

The quasi-Newton method usually has good convergence properties, although it does not match the convergence speed of the pure Newton method. The big advantage over Newton's method, however, is that the matrix-vector multiplications in (3.42) are typically significantly faster to perform than the computation of the Hessian $\nabla^2 f$ and its inverse. A further advantage of the quasi-Newton method is that there are no singularity issues (at least in theory) because $\boldsymbol{H}^k$ as well as $\boldsymbol{B}^k$ are positive definite as long as the line search (3.20) is exact.

Similar to the conjugate gradient method, it can be shown that the quasi-Newton method solves a quadratic problem

$$f(\boldsymbol{x}) = \frac{1}{2} \boldsymbol{x}^\mathsf{T} \boldsymbol{Q}\, \boldsymbol{x} + \boldsymbol{b}^\mathsf{T} \boldsymbol{x} + c\,, \quad \boldsymbol{x} \in \mathbb{R}^n$$

with positive definite $\boldsymbol{Q}$ and exact line search (3.20) in at most n iterations.

3.4 Further numerical methods

This section presents two standard optimization algorithms that do not belong to the class of line search methods. The first one is the *trust region method* and the second one belongs to the class of *direct search methods* that does not rely on gradient information.

3.4.1 Trust region method

The *trust region method* constructs a local simplified model $m^k(\boldsymbol{s}) \approx f(\boldsymbol{x}^k)$ of the cost function $f(\boldsymbol{x}^k)$ at the point $\boldsymbol{x}^k$ that is solved instead of the original optimization problem (3.1):

$$\min_{||\boldsymbol{s}|| \leq \Delta^k} m^k(\boldsymbol{s}) \quad \text{with} \quad m^k(\boldsymbol{s}) \approx f(\boldsymbol{x}^k) \quad \text{for} \quad ||\boldsymbol{s}|| \leq \Delta^k\,. \tag{3.44}$$

The search radius for the solution is restricted to the *trust region* $||\boldsymbol{s}|| \leq \Delta^k$, which is motivated by the fact that the model $m^k(\boldsymbol{s})$ is in general only locally accurate i.e. for sufficiently small $\boldsymbol{s}$. In most cases, the model m^k is constructed as the quadratic form

$$m^k(\boldsymbol{s}) = \frac{1}{2} \boldsymbol{s}^\mathsf{T} \boldsymbol{H}^k\, \boldsymbol{s} + \nabla f(\boldsymbol{x}^k)^\mathsf{T} \boldsymbol{s} + f(\boldsymbol{x}^k)\,, \tag{3.45}$$

where the matrix $\boldsymbol{H}^k$ is either given by the Hessian $\nabla^2 f(\boldsymbol{x}^k)$ or an approximation of it. The solution of the new optimization problem (3.44) is a "candidate" $\boldsymbol{s}^k$ for the next iterate $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \boldsymbol{s}^k$. If the associated cost $f(\boldsymbol{x}^{k+1})$ does not lead to a sufficient cost reduction w.r.t. the previous one $f(\boldsymbol{x}^k)$, the trust region radius Δ^k is reduced and (3.45) is solved again.

During each of these iterations with shrinking trust region radius Δ^k, the distance between $\boldsymbol{x}^k$ and the new candidate $\boldsymbol{x}^{k+1}$ becomes smaller and $\boldsymbol{s}^k$ will in general change its direction. Thus, the trust region method differs clearly from line search methods in terms of how the search direction and step size are chosen:

- Line search methods fix the search direction $\boldsymbol{s}^k$ and determine the step size α^k afterwards.
- The trust region method chooses the trust region radius Δ^k and then simultaneously computes the direction and distance to the next point $\boldsymbol{x}^{k+1}$ in an iterative manner.

The trust region radius Δ^k is adapted in each iteration by evaluating the accuracy of the model m^k compared to the cost function f via the measure

$$\rho^k(\boldsymbol{s}^k) = \frac{f(\boldsymbol{x}^k) - f(\boldsymbol{x}^k + \boldsymbol{s}^k)}{m^k(\boldsymbol{0}) - m^k(\boldsymbol{s}^k)}\,. \tag{3.46}$$

The numerator describes the actual descent in the cost, the denominator the predicted one. For $\rho^k \approx 1$, the model m^k and the actual cost f show a good agreement in the current iteration, which motivates to enlarge the trust region radius Δ^k for the next step. If in contrast $\rho^k \ll 1$, then Δ^k should be reduced, since we do not have enough confidence in the model over the current trust region. The basic algorithm of the trust region method is sketched in Table 3.4.

Initialization:	$\bar{\Delta}, \Delta^0 \in (0, \bar{\Delta})$	(Trust region: initial & max. radius)
	$\eta \in [0, \frac{1}{4})$	(Parameter)
	$k \leftarrow 0$	(Iteration index)
	ε_x, $\varepsilon_{\nabla f}$	(Convergence tolerance)
repeat		
	$m^k(\boldsymbol{s}) \leftarrow$ (3.45)	(Model)
	$\boldsymbol{s}^k \leftarrow \arg\min \left\{ m^k(\boldsymbol{s}) : \|\|\boldsymbol{s}\|\| \leq \Delta^k \right\}$	(Often solved approximately)
	$\rho^k \leftarrow$ (3.46)	(Model quality)
	if $\rho^k < \frac{1}{4}$ **do**	
	$\Delta^{k+1} \leftarrow \frac{1}{4}\Delta^k$	(Shrink trust region)
	else if $\rho^k > \frac{3}{4}$ **and** $\|\|\boldsymbol{s}^k\|\| = \Delta^k$ **do**	
	$\Delta^{k+1} \leftarrow \min\{2\Delta^k, \bar{\Delta}\}$	(Enlarge trust region)
	else	
	$\Delta^{k+1} \leftarrow \Delta^k$	
	end if	
	if $\rho^k > \eta$ **do**	
	$\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x}^k + \boldsymbol{s}^k$	(Next step)
	$\boldsymbol{H}^{k+1} \leftarrow \boldsymbol{H}^k + \ldots$	(Update of Hessian matrix)
	else	
	$\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x}^k$	(Repeat step with $\Delta^{k+1} < \Delta^k$)
	end if	
	$k \leftarrow k + 1$	
until	$\|\|\nabla f(\boldsymbol{x}^k)\|\| \leq \varepsilon_{\nabla f}$	(Or alternative tolerance criterion)

Table 3.4: Trust region method.

3.4.2 Direct search methods

All of the algorithms introduced so far depend on information about the gradient ∇f (and the Hessian $\nabla^2 f$ if available) to compute the next iteration point $\boldsymbol{x}^{k+1}$ with a sufficient decrease in the cost. There are practical cases, however, where the gradient is not available or cannot be computed or approximated with reasonable effort because the problem at hand is either too complex or not continuously differentiable. An alternative for these kind of problems are *direct search methods* that use samples of function values to determine a new iteration point.

The probably best-known direct search algorithm in unconstrained nonlinear optimization is the *simplex method* [2] developed by *Nelder* and *Mead*, which is therefore also called *Nelder-Mead method*. The algorithm operates on a *simplex* with $n+1$ points $\boldsymbol{x}^i$, $i = 1, \ldots, n+1$ in the n-dimensional parameter space $\mathbb{R}^n$. [3] For instance, the simplex is a straight line in $\mathbb{R}^1$ and a triangle in $\mathbb{R}^2$.

The function values $f(\boldsymbol{x}^i)$ are computed for the $n+1$ points $\boldsymbol{x}^i$ of the simplex and sorted in ascending order, i.e.

$$f(\boldsymbol{x}^1) \leq f(\boldsymbol{x}^2) \leq \cdots \leq f(\boldsymbol{x}^{n+1}) \,. \tag{3.47}$$

The algorithm replaces the "worst" point $\boldsymbol{x}^{n+1}$ by a new one that lies on the parametrized straight line

$$g(a) = \bar{\boldsymbol{x}} + a(\bar{\boldsymbol{x}} - \boldsymbol{x}^{n+1}) \,, \quad \bar{\boldsymbol{x}} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}^i \tag{3.48}$$

running through the point $\boldsymbol{x}^{n+1}$ with the highest cost value $f(\boldsymbol{x}^{n+1})$ and the center $\bar{\boldsymbol{x}}$ of the remaining simplex. The simplex method uses different projections by varying a in order to generate a new simplex:

a) **Reflexion** (Figure 3.8a):
Compute the reflexion point $\boldsymbol{x}_r = g(1)$ and $f_r = f(\boldsymbol{x}_r)$.
If $f(\boldsymbol{x}^1) \leq f_r < f(\boldsymbol{x}^n)$, set $\boldsymbol{x}^{n+1} = \boldsymbol{x}_r$.

b) **Expansion** (Figure 3.8b):
If $f_r < f(\boldsymbol{x}^1)$, compute $\boldsymbol{x}_e = g(2)$ and $f_e = f(\boldsymbol{x}_e)$.
If $f_e < f_r$, set $\boldsymbol{x}^{n+1} = \boldsymbol{x}_e$. Otherwise, set $\boldsymbol{x}^{n+1} = \boldsymbol{x}_r$.

c) **Outer contraction** (Figure 3.8c):
If $f(\boldsymbol{x}^n) \leq f_r < f(\boldsymbol{x}^{n+1})$, compute $\boldsymbol{x}_c = g(\frac{1}{2})$ and $f_c = f(\boldsymbol{x}_c)$.
If $f_c \leq f_r$, set $\boldsymbol{x}^{n+1} = \boldsymbol{x}_c$. Otherwise, go to Step e).

d) **Inner contraction** (Figure 3.8d):
If $f_r \geq f(\boldsymbol{x}^{n+1})$, compute $\boldsymbol{x}_c = g(-\frac{1}{2})$ and $f_c = f(\boldsymbol{x}_c)$.
If $f_c < f(\boldsymbol{x}^{n+1})$, set $\boldsymbol{x}^{n+1} = \boldsymbol{x}_c$. Otherwise, go to Step e).

e) **Shrinking** (Figure 3.8e):
Shrink the simplex around the "best" point $\boldsymbol{x}^1$:
$\boldsymbol{x}^i = \frac{1}{2}(\boldsymbol{x}^1 + \boldsymbol{x}^i)$, $i = 2, \ldots, n+1$.

[2] The simplex method of Nelder and Mead is not to be confused with the simplex algorithm of *G. Danzig* in linear programming.

[3] We omit the iteration index k for the sake of readibility.

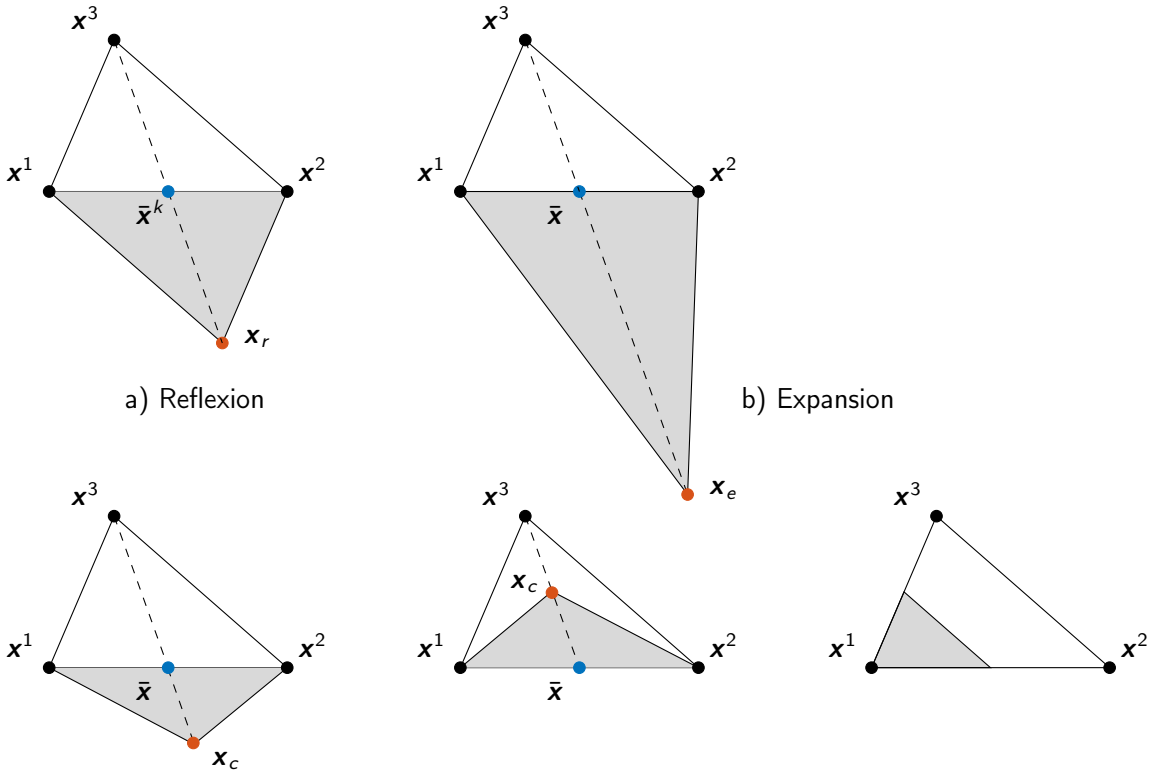


Figure 3.8: Operations of the simplex method of Nelder and Mead.

The algorithm and the individual simplex operations are summarized in Table 3.5. The simplex method is iteratively continued until a convergence criterion is satisfied. Over the iterations, the simplex moves towards the optimum and contracts.

Due to the fact that the simplex method is a heuristic algorithm, it is hard to guarantee convergence. In fact, the simplex may converge to a non-optimal point. In this case, a standard approach is to restart the algorithm with a new simplex at the last point. In practice, however, the simplex method usually performs well and robust and is one of the most popular optimization algorithms, albeit its moderate speed of convergence.

Initialization:	$\boldsymbol{x}^i, \ldots, \boldsymbol{x}^{n+1}$	(Initial simplex)
	$\varepsilon_x, \varepsilon_f > 0$	(Convergence tolerance)
repeat		
	Sorting (3.47)	
	$\bar{\boldsymbol{x}} \leftarrow$ (3.48)	(Center of remaining simplex)
	Exchange of $\boldsymbol{x}^{n+1}$	(Reflexion/Expansion/Contraction)
	or of all points $\boldsymbol{x}^i$	(Shrinking)
until	$\max_{2 \leq i \leq n+1} \lVert \boldsymbol{x}^i - \boldsymbol{x}^1 \rVert \leq \varepsilon_x$	**or** $f(\boldsymbol{x}^{n+1}) - f(\boldsymbol{x}^1) \leq \varepsilon_f$

Table 3.5: Simplex method of Nelder and Mead.

3.5 Example: Rosenbrock's "banana function"

A well-known benchmark problem in optimization is *Rosenbrock's* problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^2} f(\boldsymbol{x}) \quad \text{with} \quad f(\boldsymbol{x}) = 100(x_2 - x_1^2)^2 + (x_1 - 1)^2 \,. \tag{3.49}$$

Figure 3.9 shows the profile and the contour lines of the function, which for apparent reasons is also known as *banana function*. We will use Rosenbrock's problem to compare the algorithms of this chapter numerically under MATLAB.

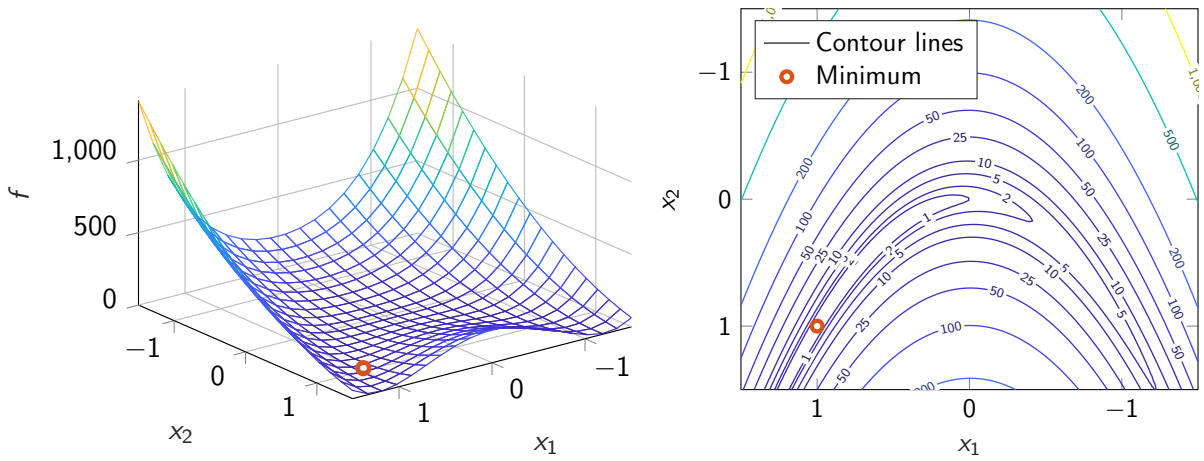


Figure 3.9: Profile and contour lines of Rosenbrock's banana function.

Exercise 3.4 *Verify that the point* $\boldsymbol{x}^* = [1, 1]^\mathsf{T}$ *is a unique minimum point. Are the function* $f(\boldsymbol{x})$ *and the optimization problem (3.49) convex?*

MATLAB offers the *Optimization Toolbox* for the solution of parameter optimization problems. The following functions are particularly relevant for unconstrained optimization:

- fminunc: line search method (gradient, quasi-Newton) & trust region method (Newton)
- fminsearch: simplex method of Nelder and Mead.

An alternative is the free MATLAB function minFunc [5] that offers a wide range of line search methods. Table 3.6 shows selected key figures of the different algorithms for the numerical solution of Rosenbrock's problem with fminunc, fminsearch, and minFunc. The initial point was $\boldsymbol{x}^0 = [-1, -1]^\mathsf{T}$ in all cases.

Figure 3.10 additionally shows the iteration profiles of the methods implemented in fminunc and fminsearch. For the sake of completeness, the MATLAB code for Rosenbrock's problem (3.49) is shown in Figure 3.12 at the end of the chapter to illustrate the option settings and function call for fminunc and fminsearch.

The gradient method shows a stagnating convergence, because even after reaching the maximum number of function evaluations (200), the minimum has not yet been reached. Figure 3.11 additionally shows the iterations of the gradient method plotted together with the contour lines of the Rosenbrock function (3.49). The single gradient steps in steepest descent

Method	Iter.	$f(\boldsymbol{x}^*)$	$\nabla f(\boldsymbol{x}^*)$	Function calls $f(\boldsymbol{x})$	$\nabla f(\boldsymbol{x})$	$\nabla^2 f(\boldsymbol{x})$
LS: Gradient method	57	0.1232	1.1978	200	200	-
LS: Conj. grad. method.	31	$4.1 \cdot 10^{-17}$	$6.9 \cdot 10^{-9}$	91	90	-
LS: Newton's method	21	$3.5 \cdot 10^{-17}$	$3.5 \cdot 10^{-8}$	29	28	24
LS: Quasi-Newton (BFGS)	23	$5.4 \cdot 10^{-12}$	$9.2 \cdot 10^{-6}$	29	29	-
TR: Newton's method	25	$2.2 \cdot 10^{-18}$	$2.1 \cdot 10^{-8}$	26	26	26
Simplex method (Nelder-Mead)	67	$5.3 \cdot 10^{-10}$	-	125	-	-

Table 3.6: Comparison of optimization methods for Rosenbrock's problem (LS: line search method, TR: trust region method).

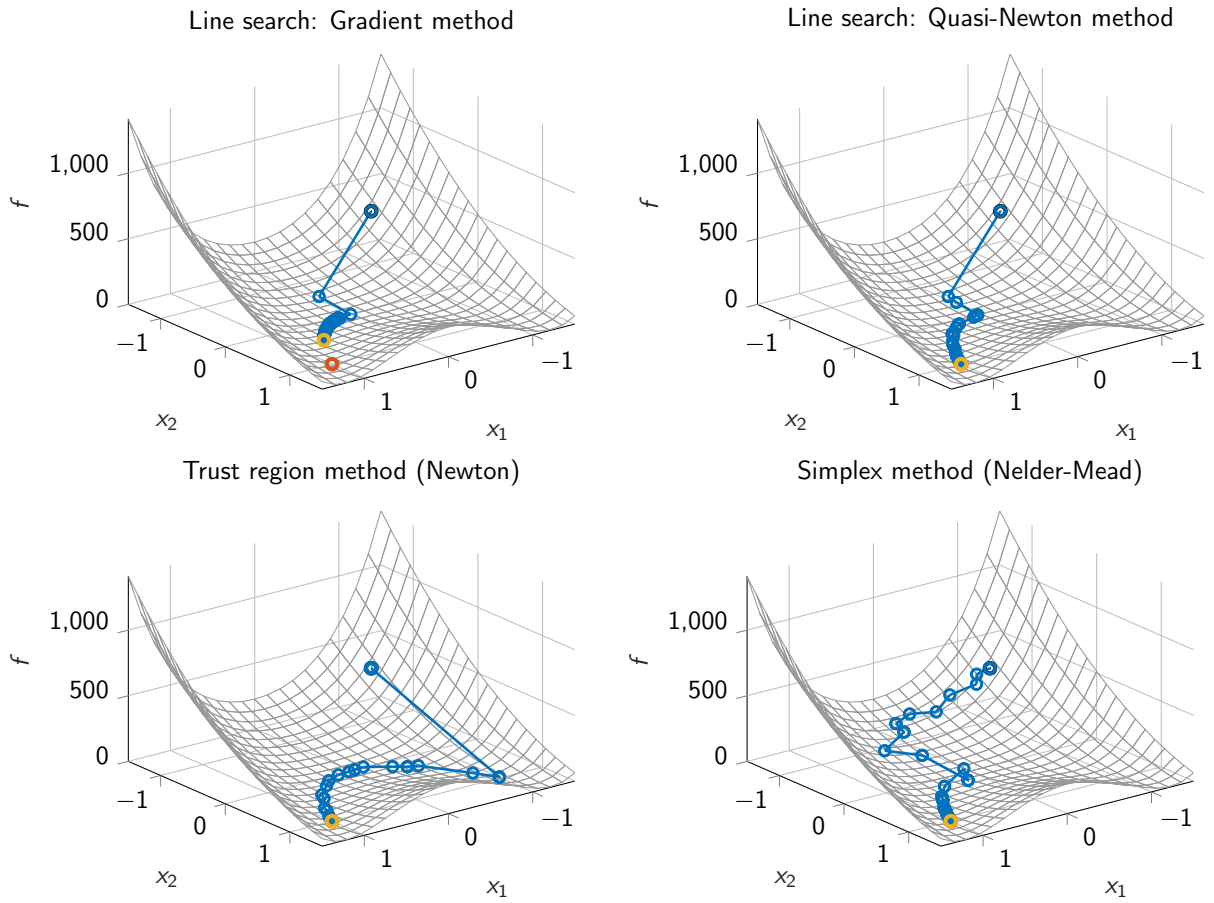


Figure 3.10: Iteration profiles for Rosenbrock's problem.

direction are perpendicular to the respective contour lines and become smaller over the iterations. The slow convergence has already been addressed in Theorem 3.4, also see Figure 3.6, and will be examined in more detail in the following exercise.

Exercise 3.5 *Determine the convergence rate of the gradient method in the neighborhood of the minimum* $\boldsymbol{x}^* = [1, 1]^\mathsf{T}$ *of Rosenbrock's problem (3.49).*

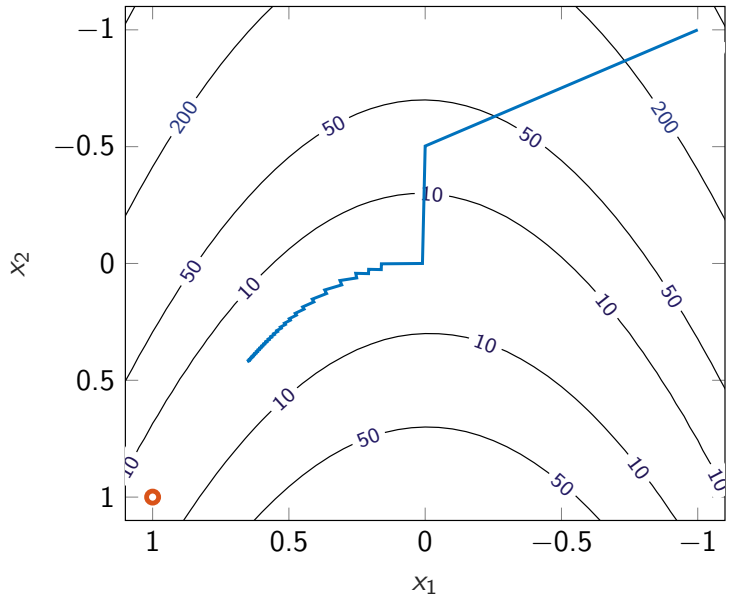


Figure 3.11: Iterations of the gradient method.

The quasi-Newton method in Figure 3.10 converges significantly better than the gradient method. The Newton/trust region method in Figure 3.10 initially starts in the "wrong" direction, because the quadratic approximation (3.45) at the initial point $\boldsymbol{x}^0 = [-1, 1]^\mathrm{T}$ has its minimum close to $\boldsymbol{x} \approx [-1, 1]^\mathrm{T}$. However, the individual steps along the valley of the Rosenbrock function are much larger since Newton's method uses the Hessian matrix explicitly and does not rely on an approximation as in the case of the quasi-Newton method.

Table 3.6 and Figure 3.10 additionally show the results for the simplex method that are obtained with the MATLAB function fminsearch. The graphical output under fminsearch does not provide the possibility to display the individual simplices (or simplexes). The next exercise, therefore, examines the simplex method in more detail to get an impression of the simplex operations and its robustness.

Exercise 3.6 *Write a* MATLAB *function that solves Rosenbrock's problem (3.49) with the simplex method of Nelder and Mead. Construct the first simplex with the edges*

$$\boldsymbol{x}^{0,1} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \quad \boldsymbol{x}^{0,2} = \boldsymbol{x}^{0,1} + s \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \boldsymbol{x}^{0,3} = \boldsymbol{x}^{0,1} + s \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.50}$$

and the side length $s = 1$. Use the tolerance $\varepsilon_x = \varepsilon_f = 10^{-9}$ for the convergence criterion in Table 3.5 and compare your results with the fminsearch results in Table 3.6. Plot the simplices graphically for the single iterations. Investigate the robustness and convergence behavior for different values of the side length s of the inital simplex and different initial points $\boldsymbol{x}^{0,1}$.

Exercise 3.7 (optional) *Write a* MATLAB *function that solves Rosenbrock's problem (3.49) with the Newton line search method. Use backtracking to determine the step size α^k with the default $\bar{\alpha} = 1$ and the Armijo condition (3.21) as stopping criterion. Compare the convergence results with Table 3.6. Plot the single iterates graphically and vary the initial point $\boldsymbol{x}^0$.*

3.6 References

[1] S. Boyd and L. Vandenberghe. *Convex Optimization*. `http://www.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf`. Cambridge: Cambridge University Press, 2004.

[2] C.T. Kelley. *Iterative Methods for Optimization*. Society for Industrial and Applied Mathematics (SIAM), 1999.

[3] J. Nocedal and S.J. Wright. *Numerical Optimization*. New York: Springer, 2006.

[4] M. Papageorgiou, M. Leibold, and M. Buss. *Optimierung*. Berlin: Springer, 2012.

[5] Mark Schmidt. *minFunc (*MATLAB*-Funktion zur Lösung unbeschränkter Optimierungsprobleme)*. University of British Columbia, Vancouver, Kanada. Available at `https://www.cs.ubc.ca/~schmidtm/Software/minFunc.html`.

```
function Xopt = rosenbrock_problem(Xinit,methodQ)
% ------------------------------------------------------
% Xinit:   initial point
% methodQ: 1 - line search: gradient method
%          2 - line search: quasi-Newton
%          3 - trust region method (Newton)
%          4 - Nelder-Mead simplex method
global old
old = [Xinit; rosenbrock(Xinit)];

% options for optimization
opt1 = optimoptions('fminunc','Display','iter','PlotFcns',@plot_iterates);
opt2 = optimset('Display','iter','PlotFcns',@plot_iterates);

switch methodQ
 case 1,      % line search: gradient method
  opt1 = optimoptions(opt1,'HessUpdate','steepdesc','GradObj','on');
 case 2,      % line search: quasi-Newton
  opt1 = optimoptions(opt1,'Algorithm','quasi-newton','GradObj','on');
case 3,      % trust region method (Newton)
  opt1 = optimoptions(opt1,'Algorithm','trust-region','Hessian','on','GradObj','on');
end

if methodQ<4,   Xopt = fminunc(@rosenbrock,Xinit,opt1);        % numeric solution with
else            Xopt = fminsearch(@rosenbrock,Xinit,opt2);     % fminunc or finsearch
end

function [f, grad, H] = rosenbrock(x)
% ------------------------------------------------------
grad = {}; H = {};
f = 100*(x(2)-x(1)^2)^2 + (x(1)-1)^2;                          % Rosenbrock function
if nargout>1,                                                  % if gradient is requested
  grad = [ -400*(x(2)-x(1)^2)*x(1)+2*(x(1)-1);
            200*(x(2)-x(1)^2) ];
end
if nargout>2,                                                  % if Hessian is requested
  H = [ -400*(x(2)-3*x(1)^2)+2,   -400*x(1);
        -400*x(1),                 200 ];
end

function stop = plot_iterates(x,info,state)
% ------------------------------------------------------
global old
f = rosenbrock(x);
switch state                                                   % graphical output:
 case 'init',                                                  % initialization
  plot_surface(x,f);
 case 'iter',                                                  % iterations
  plot3([old(1),x(1)],[old(2),x(2)],[old(3),f],'b-o','LineWidth',1);
 case 'done',                                                  % after last iterations
  plot3(x(1),x(2),f,'go','LineWidth',5);
end
stop = false;                                                  % no stopping criterion
old = [x;f];

function plot_surface(x,f)
% ------------------------------------------------------
[X1,X2] = meshgrid(-1.5:0.15:1.5);                             % 3D profile of
F = 100*(X2-X1.^2).^2 + (X1-1).^2;                             % Rosenbrock function
h = surf(X1,X2,F,'EdgeColor',0.6*[1,1,1],'FaceColor','none');
hold on; axis tight;
plot3(x(1),x(2),f,'ko','LineWidth',5);                         % initial point
plot3(1,1,0,'ro','LineWidth',5);                               % optimal solution
xlabel('x_1'); ylabel('x_2'); zlabel('f')
set(gcf,'ToolBar','figure');                                   % figure settings
set(gca,'Xdir','reverse','Ydir','reverse');
```

Figure 3.12: MATLAB code for Rosenbrock's problem (fminunc, fminsearch).

4 Constrained optimization

This chapter extends the consideration of unconstrained (parameter) optimization problems to constrained ones of the form

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \quad f(\boldsymbol{x}) \tag{4.1a}$$

$$\text{s.t.} \quad g_i(\boldsymbol{x}) = 0, \quad i = 1, \dots, p \tag{4.1b}$$

$$h_i(\boldsymbol{x}) \leq 0, \quad i = 1, \dots, q \tag{4.1c}$$

with the equality and inequality constraints (4.1b) and (4.1c). As it will turn out, the necessary and sufficient optimality conditions as well as the numerical algorithms to solve constrained optimization problems differ significantly from the unconstrained case (Chapter 3).

4.1 Illustrative examples

We recycle Example 1.2 from the introduction and successively augment the problem by the constraints (1.10)-(1.12) to highlight the extension of the unconstrained case as well as the difference between the handling of equality and inequality constraints.

4.1.1 One equality constraint

Example 4.1 *We consider the optimization problem (1.9) with the equality constraint (1.10)*

$$\min_{\boldsymbol{x}\in\mathbb{R}^2} \quad f(\boldsymbol{x}) = (x_1 - 2)^2 + (x_2 - 1)^2 \tag{4.2a}$$

$$\text{s.t.} \quad g(\boldsymbol{x}) = x_2 - 2x_1 = 0\,. \tag{4.2b}$$

Figure 4.1 shows the constraint and the contour lines of $f(\boldsymbol{x})$. Obviously, an optimal point $\boldsymbol{x}^*$ must lie on the straight line $g(\boldsymbol{x}) = 0$. If we consider, for instance, the contour line $f(\boldsymbol{x}) = 4$, then two intersection points with $g(\boldsymbol{x}) = 0$ exist that move closer together for decreasing contour lines $f(\boldsymbol{x}) < 4$ and eventually lead to the minimum point

$$\boldsymbol{x}^* = \begin{bmatrix} 0.8 \\ 1.6 \end{bmatrix} \quad \textit{with} \quad f(\boldsymbol{x}^*) = 1.8\,. \tag{4.3}$$

An important role in this regard play the gradients of the functions $f(\boldsymbol{x})$ and $g(\boldsymbol{x})$

$$\nabla f(\boldsymbol{x}) = \begin{bmatrix} 2(x_1 - 2) \\ 2(x_2 - 1) \end{bmatrix}, \quad \nabla g(\boldsymbol{x}) = \begin{bmatrix} -2 \\ 1 \end{bmatrix}. \tag{4.4}$$

For all non-optimal points on the constraint $g(\boldsymbol{x}) = 0$, the gradients ∇f and ∇g are not collinear, i.e. there exists a component of $-\nabla f$ (steepest descent) in direction of $g(\boldsymbol{x}) = 0$

that leads to a decrease of the cost function f while staying on the constraint $g(\boldsymbol{x}) = 0$. Vice versa, the gradients at the optimal point $\boldsymbol{x}^*$

$$\nabla f(\boldsymbol{x}^*) = \begin{bmatrix} -2.4 \\ 1.2 \end{bmatrix}, \quad \nabla g(\boldsymbol{x}^*) = \begin{bmatrix} -2 \\ 1 \end{bmatrix} \tag{4.5}$$

must be collinear, i.e. $\nabla f(\boldsymbol{x}^*) || \nabla g(\boldsymbol{x}^*)$, such that the cost cannot be further reduced without leaving the constraint. Mathematically speaking, the optimality condition is

$$0 = \nabla f(\boldsymbol{x}^*) + \lambda^* \nabla g(\boldsymbol{x}^*) \tag{4.6}$$

for some $\lambda^* \in \mathbb{R}$, in our case $\lambda^* = -1.2$.

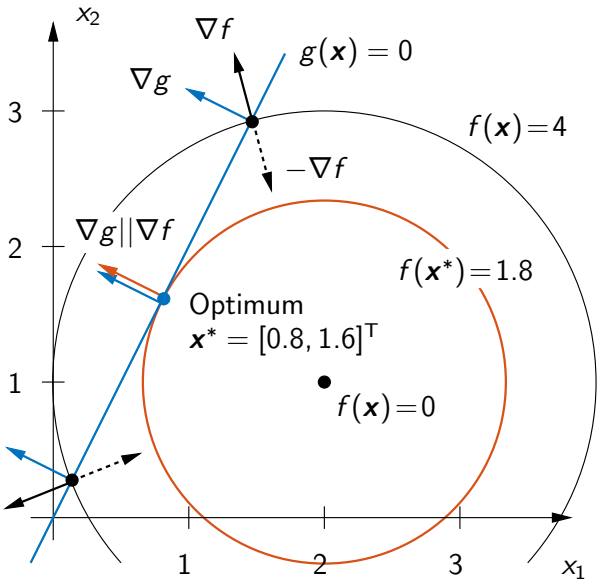


Figure 4.1: Illustration of the equality-constrained optimization problem (4.2).

The result of the last example can be generalized with the *Lagrangian function*

$$L(\boldsymbol{x}, \lambda) = f(\boldsymbol{x}) + \lambda\, g(\boldsymbol{x})\,, \tag{4.7}$$

where the equality constraint is added to the cost via the *Lagrange multiplier* λ. The collinearity of the gradients ∇f and ∇g at an optimal point $\boldsymbol{x}^*$ corresponds to the *stationarity condition*

$$\nabla_{\boldsymbol{x}} L(\boldsymbol{x}^*, \lambda^*) = \nabla f(\boldsymbol{x}^*) + \lambda^* \nabla g(\boldsymbol{x}^*) = \boldsymbol{0}\,. \tag{4.8}$$

Together with the equality constraint $g(\boldsymbol{x}^*) = 0$, we have a set of equations of order $n+1$ for the $n+1$ unknown variables $\boldsymbol{x}^* \in \mathbb{R}^n$ and $\lambda^* \in \mathbb{R}$. It is, however, important to note that (4.8) is only a *necessary* and not a *sufficient* condition for optimality.

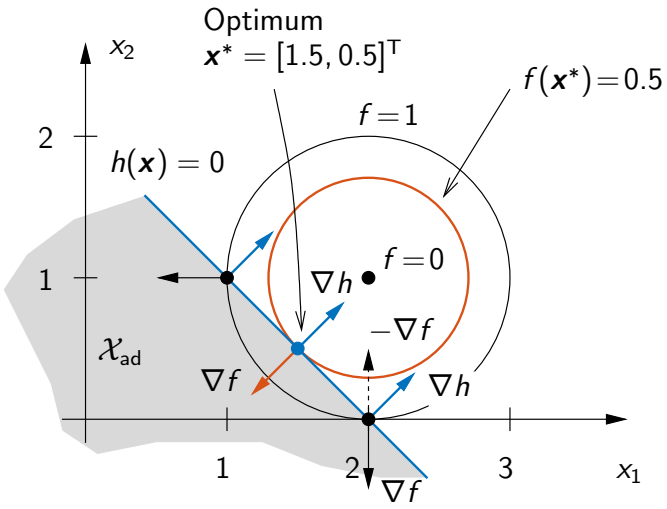


Figure 4.2: Illustration of inequality-constrained optimization problem (4.9).

4.1.2 One inequality constraint

Example 4.2 *The equality constraint (1.10) of the previous example is now replaced by the inequality constraint (1.11), i.e.*

$$\min_{x\in\mathbb{R}^2} \quad f(x) = (x_1 - 2)^2 + (x_2 - 1)^2 \tag{4.9a}$$

$$\text{s.t.} \quad h(x) = x_1 + x_2 - 2 \leq 0\,. \tag{4.9b}$$

Figure 4.2 shows the optimal point $x^* = [1.5, 0.5]^T$ *that lies on the boundary of the admissible set* [1]

$$\mathcal{X}_{ad} = \{x \in \mathbb{R}^2 : h(x) \leq 0\}\,.$$

The constraint (4.9b) is active, *i.e.* $h(x^*) = 0$, *and the gradients* $\nabla f(x^*) = [-1, -1]^T$, $\nabla h(x^*) = [1, 1]^T$ *are collinear at the optimal point* x^*. *For all other points where the constraint (4.9b) is active, there exists a component of the gradient* $-\nabla f$ *in direction of* $h(x) = 0$ *that leads to a reduction of the cost* f *while keeping the inequality constraint (4.9b) active, see Figure 4.2.*

Similar to the equality-constrained case, an inequality constraint $h(x) \leq 0$ can be added to the Lagrangian function

$$L(x, \mu) = f(x) + \mu\, h(x) \tag{4.10}$$

with a Lagrange multiplier μ. The difference now is that we have to distinguish whether the constraint $h(x) \leq 0$ is active or not at the optimal point x^*. If $h(x^*) < 0$ strictly holds, then x^* corresponds to the unconstrained case and we have

$$\nabla_x L(x^*, \mu^*) = \nabla f(x^*) = 0 \quad \text{with} \quad \mu^* = \mathbf{0}\,. \tag{4.11}$$

In case of an active constraint, $h(x) = 0$ can be considered as equality constraints, i.e. the gradients ∇f and ∇h must be collinear at the optimal point x^*. This can be expressed with the Lagrangian function (4.10) as

$$\nabla_x L(x^*, \mu^*) = \nabla f(x^*) + \mu^* \nabla h(x^*) = 0\,. \tag{4.12}$$

[1] A useful geometric interpretation of an active inequality constraint $h(x) \leq 0$ is that the negative gradient $-\nabla h(x)$ always points in the direction of the admissible set $\mathcal{X}_{ad}$.

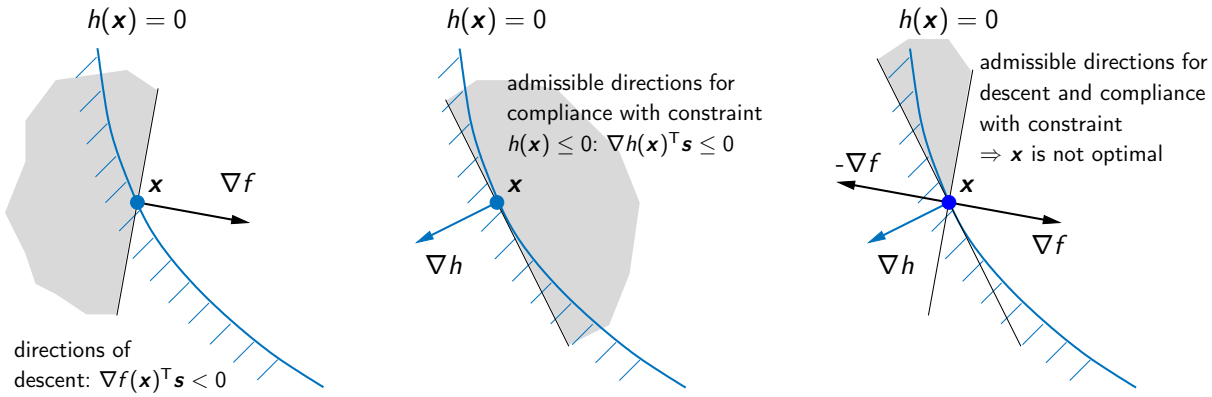


Figure 4.3: Geometric interpretation of the conditions (4.13) and (4.14) for an active inequality constraint.

However, in contrast to the equality-constrained case, the sign of the multiplier μ^* now plays a crucial role. To illustrate this further, the function $h(\boldsymbol{x})$ is expanded in a first-order Taylor series

$$h(\boldsymbol{x} + \boldsymbol{s}) = h(\boldsymbol{x}) + \nabla h(\boldsymbol{x})^T \boldsymbol{s} + \mathcal{O}(||\boldsymbol{s}||^2)\,.$$

In case of an active constraint $h(\boldsymbol{x}) = 0$, all (sufficiently small) admissible directions $\boldsymbol{s}$ in the sense of $h(\boldsymbol{x} + \boldsymbol{s}) \leq 0$ must satisfy

$$\nabla h(\boldsymbol{x})^T \boldsymbol{s} \leq 0\,. \tag{4.13}$$

In addition, the cost function $f(\boldsymbol{x})$ is expanded in a Taylor series

$$f(\boldsymbol{x} + \boldsymbol{s}) = f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^T \boldsymbol{s} + \mathcal{O}(||\boldsymbol{s}||^2)\,.$$

If the point $\boldsymbol{x}$ is *no* minimum, there exists an admissible descent direction $\boldsymbol{s}$ with $f(\boldsymbol{x} + \boldsymbol{s}) < f(\boldsymbol{x})$ if

$$\nabla f(\boldsymbol{x})^T \boldsymbol{s} < 0 \tag{4.14}$$

and (4.13) are satisfied at the same time. Vice versa, if $\boldsymbol{x}^*$ is a minimum point at the boundary $h(\boldsymbol{x}^*) = 0$, then no direction $\boldsymbol{s}$ is allowed to exist such that *both* conditions (4.13), (4.14) are satisfied.

This reasoning is illustrated in Figure 4.3 for a non-optimal point $\boldsymbol{x}$. Obviously, no admissible direction $\boldsymbol{s}$ exists that satisfies both inequalities (4.13) and (4.14) if $-\nabla f$ and ∇h point in the same direction, i.e.

$$-\nabla f(\boldsymbol{x}^*) = \mu^* \nabla h(\boldsymbol{x}^*) \quad \text{with} \quad \mu^* \geq 0\,. \tag{4.15}$$

In other words, if (4.15) holds then there exists no admissible direction $\boldsymbol{s}$ that leads to a reduction of the cost function $f(\boldsymbol{x}^* + \boldsymbol{s})$ without violating the active constraint $h(\boldsymbol{x}^*) = 0$. The discussion shows that the sign of μ^* is important. If the multiplier was negative, i.e. $\mu^* < 0$, then ∇f and ∇h would point into the same direction and a half plane of admissible directions would exist for a decrease in the cost f while satisfying the constraint $h(\boldsymbol{x}) \leq 0$.

The conditions for a minimum at the point $\boldsymbol{x}^*$ can be summarized as follows:

$$\exists \mu^* \geq 0: \quad \nabla_x L(\boldsymbol{x}^*, \mu^*) = 0\,, \quad \mu^* h(\boldsymbol{x}^*) = 0\,. \tag{4.16}$$

The equation $\mu^* h(\boldsymbol{x}^*) = 0$ is called *complementary condition*. It is trivially satisfied in case of an active constraint $h(\boldsymbol{x}^*) = 0$ and implies $\mu^* = 0$ in case of $h(\boldsymbol{x}^*) < 0$.

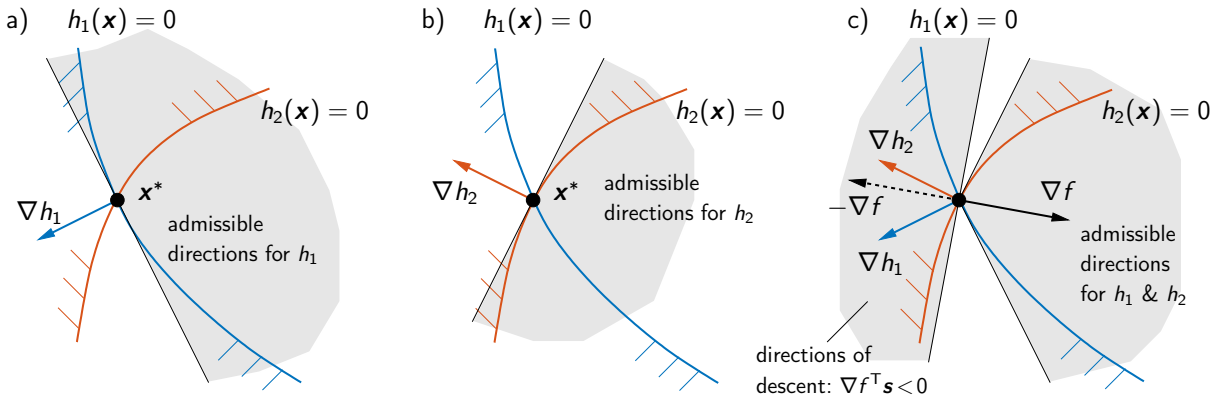


Figure 4.4: Geometric interpretation of the conditions (4.18) and (4.19) for two active inequality constraints.

4.1.3 Two inequality constraints

To further generalizes the previous considerations, we consider an optimization problem with two inequality constraints

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{4.17a}$$

$$\text{s.t.} \quad h_1(x) \leq 0 \tag{4.17b}$$

$$h_2(x) \leq 0 \tag{4.17c}$$

and follow the same reasoning as in the previous example of one inequality constraint.

If we consider a point x where both constraints are active, i.e. $h_1(x) = 0$ and $h_2(x) = 0$, then an admissible (suffiently small) direction s must satisfy the condition (4.13) for both constraints

$$\nabla h_1(x)^T s \leq 0\,, \quad \nabla h_2(x)^T s \leq 0\,. \tag{4.18}$$

These conditions can be interpreted in the $\mathbb{R}^2$-plane: all admissible directions s must lie in the cone defined by the tangents at the curves $h_1(x) = 0$ and $h_2(x) = 0$, see Figure 4.4.

If the point x is no minimum point, then there exist directions s that satisfy the descent condition (4.14)

$$\nabla f(x)^T s < 0 \tag{4.19}$$

while complying with the constraints according to (4.18). Vice versa, at a minimum point x^* no direction s exists that satisfies both (4.18) and (4.19). This is the case if the negative gradient $-\nabla f(x^*)$ points in the cone that is spanned by the gradients $\nabla h_1(x^*)$ and $\nabla h_2(x^*)$, see Figure 4.4c.

In other words, the negative gradient $-\nabla f(x^*)$ at a minimum point x^* with active inequality constraints must be a *positive* linear combination of the gradients of the constraint functions, i.e.

$$\exists\, \mu_1^*, \mu_2^* \geq 0: \quad -\nabla f(x^*) = \mu_1^* \nabla h_1(x^*) + \mu_2^* \nabla h_2(x^*)\,. \tag{4.20}$$

Similar to before, the Lagrangian function

$$L(x, \mu) = f(x) + \mu_1 h_1(x) + \mu_2 h_2(x)\,, \quad \mu = [\mu_1, \mu_2]^T \tag{4.21}$$

can be used to formulate the necessary optimality conditions for a local minimum

$$\exists\, \mu_1^*, \mu_2^* \geq 0: \quad \nabla_x L(\boldsymbol{x}^*, \boldsymbol{\mu}^*) = 0\,, \quad \mu_1^* h_1(\boldsymbol{x}^*) = 0\,, \quad \mu_2^* h_2(\boldsymbol{x}^*) = 0\,. \tag{4.22}$$

Both complementary conditions in (4.22) are satisfied, if the respective constraint $h_i(\boldsymbol{x}^*) \leq 0$ is active or if the multiplier is $\mu_i^* = 0$.

Exercise 4.1 *The optimization problem of Example 4.2 is extended by the second inequality constraint (1.12)*

$$\min_{\boldsymbol{x}\in\mathbb{R}^2} \quad f(\boldsymbol{x}) = (x_1 - 2)^2 + (x_2 - 1)^2 \tag{4.23a}$$

$$\text{s.t.} \quad h_1(\boldsymbol{x}) = x_1 + x_2 - 2 \leq 0 \tag{4.23b}$$

$$h_2(\boldsymbol{x}) = x_1^2 - x_2 \leq 0\,. \tag{4.23c}$$

Compute the Lagrange multipliers for the optimal solution $\boldsymbol{x}^* = [1, 1]^\mathrm{T}$ and verify the sign condition (4.20) graphically.

4.2 Optimality conditions

The derivations from the previous section can be generalized to the constrained optimization problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \quad f(\boldsymbol{x}) \tag{4.24a}$$

$$\text{s.t.} \quad g_i(\boldsymbol{x}) = 0, \quad i = 1, \ldots, p \tag{4.24b}$$

$$h_i(\boldsymbol{x}) \leq 0, \quad i = 1, \ldots, q \tag{4.24c}$$

with p equality constraints and q inequality constraints.

4.2.1 Constraint qualification

The equality-constrained example in Section 4.1.1 already showed that the $n + 1$ equations

$$g(\boldsymbol{x}) = 0, \quad \nabla_x L(\boldsymbol{x}, \lambda) = \nabla f(\boldsymbol{x}) + \lambda\, \nabla g(\boldsymbol{x}) = \boldsymbol{0}$$

must be solved together for the unknowns $\boldsymbol{x} \in \mathbb{R}^n$ and $\lambda \in \mathbb{R}$. The same is true in the case of an active inequality constraint $h(\boldsymbol{x}) = 0$ in order to determine $\boldsymbol{x} \in \mathbb{R}^n$ and $\mu \in \mathbb{R}$ from the set of equations

$$h(\boldsymbol{x}) = 0, \quad \nabla_x L(\boldsymbol{x}, \mu) = \nabla f(\boldsymbol{x}) + \mu\, \nabla h(\boldsymbol{x}) = \boldsymbol{0}\,.$$

For the general optimization problem (4.24) with p equality constraints and q inequality constraints, an additional assumption for active constraints is necessary to ensure the solvability of the optimality conditions. To this end, we require the following definition.

Definition 4.1 (Active constraint set) *The active constraint set of the optimization problem (4.24) at a point $\boldsymbol{x}$ is defined by*

$$\mathcal{A}(\boldsymbol{x}) = \{i \in \{1, \ldots, q\} \,\big|\, h_i(\boldsymbol{x}) = 0\}\,. \tag{4.25}$$

The so-called *constraint qualification* requires the gradients of the equality constraints and active inequality constraints to be linearly independent, i.e.

$$\operatorname{rank} \begin{bmatrix} \nabla \boldsymbol{g}(\boldsymbol{x}) \\ \nabla \boldsymbol{h}_{\mathcal{A}(\boldsymbol{x})}(\boldsymbol{x}) \end{bmatrix} = p + |\mathcal{A}(\boldsymbol{x})| \ . \tag{4.26}$$

The vector gradient notation $\nabla \boldsymbol{g}(\boldsymbol{x})$ is defined as $\nabla \boldsymbol{g}(\boldsymbol{x}) = [\nabla g_1(\boldsymbol{x}), \dots, \nabla g_p(\boldsymbol{x})]^\mathsf{T}$. The second term $\nabla \boldsymbol{h}_{\mathcal{A}(\boldsymbol{x})}(\boldsymbol{x})$ is similarly defined with the vector function $\boldsymbol{h}_{\mathcal{A}(\boldsymbol{x})}$ consisting of the active inequality constraints at point $\boldsymbol{x}$

$$\boldsymbol{h}_{\mathcal{A}(\boldsymbol{x})}(\boldsymbol{x}) = \left[h_i(\boldsymbol{x}) \right]_{i \in \mathcal{A}(\boldsymbol{x})} . \tag{4.27}$$

The rank condition (4.26) also means that at most $n - p$ inequality constraints can be active at point $\boldsymbol{x}$. Otherwise, the set of equations consisting of $\boldsymbol{g}(\boldsymbol{x}) = 0$ and $\boldsymbol{h}_{\mathcal{A}(\boldsymbol{x})}(\boldsymbol{x}) = 0$ would be overdetermined.

The constraint qualification (4.26) is the most common one in optimization. However, note that (4.26) is only a *sufficient* condition and might be too conservative in some cases. Further constraint qualifications can be found in the literature.

4.2.2 First-order optimality conditions

The Lagrangian function was already introduced for the examples in Section 4.1. We now extend the concept to the general optimization problem (4.24).

Definition 4.2 (Lagrangian function) *The Lagrangian function of the optimization problem (4.24) is defined as*

$$\boxed{L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\boldsymbol{x}) + \sum_{i=1}^{p} \lambda_i g_i(\boldsymbol{x}) + \sum_{i=1}^{q} \mu_i h_i(\boldsymbol{x})} \tag{4.28}$$

with the Lagrange multipliers $\boldsymbol{\lambda} = [\lambda_1, \dots, \lambda_p]^\mathsf{T}$ *and* $\boldsymbol{\mu} = [\mu_1, \dots, \mu_q]^\mathsf{T}$.

We can now state the necessary first-order optimality conditions for the optimization problem (4.24) as follows:

Theorem 4.1 (Necessary first-order optimality conditions) *Let* $\boldsymbol{x}^*$ *be a local solution of the optimization problem (4.24) and let the constraint qualification (4.26) be satisfied at point* $\boldsymbol{x}^*$. *If the functions* f, g_i, *and* h_i *are continuously differentiable, then there exist Lagrange multipliers* $\boldsymbol{\lambda}^* = [\lambda_1^*, \dots, \lambda_p^*]^\mathsf{T}$ *and* $\boldsymbol{\mu}^* = [\mu_1^*, \dots, \mu_q^*]^\mathsf{T}$ *such that the following conditions are satisfied*

$$\boxed{\begin{aligned} \nabla_x L(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) &= \boldsymbol{0} && && \text{(4.29a)} \\ g_i(\boldsymbol{x}^*) &= 0 \,, && i = 1, \dots, p && \text{(4.29b)} \\ h_i(\boldsymbol{x}^*) &\leq 0 \,, && i = 1, \dots, q && \text{(4.29c)} \\ \mu_i^* &\geq 0 \,, && i = 1, \dots, q && \text{(4.29d)} \\ \mu_i^* \, h_i(\boldsymbol{x}^*) &= 0, && i = 1, \dots, q \,. && \text{(4.29e)} \end{aligned}}$$

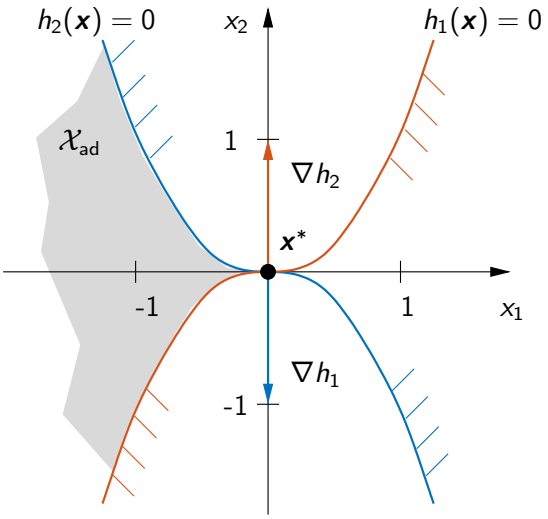


Figure 4.5: Illustration of problem (4.30) in Example 4.3.

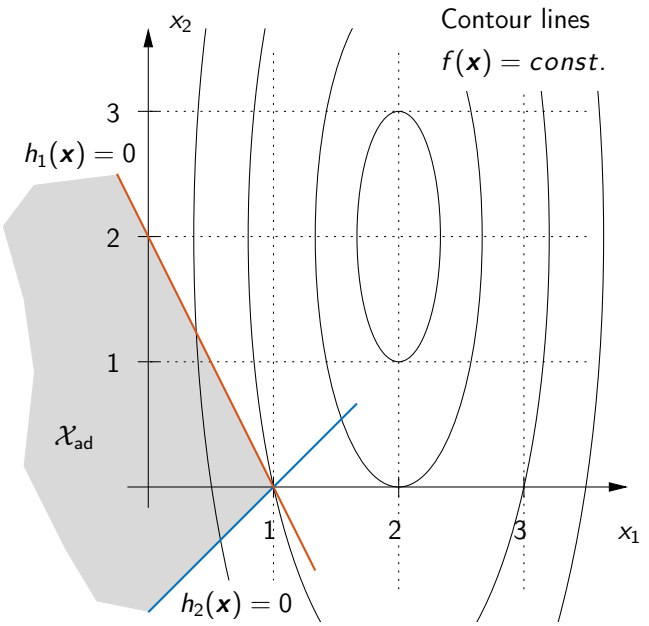


Figure 4.6: Contour lines and admissible set in Exercise 4.2.

The conditions (4.29) are known as *Karush-Kuhn-Tucker conditions* or simply *KKT conditions*. The *complementary condition* implies that the i-th inequality constraint is either active, i.e. $h_i(\boldsymbol{x}^*) = 0$, or $\mu_i^* = 0$ holds. The optimality conditions in Theorem 4.1 are only valid if the constraint qualification (4.26) is satisfied at $\boldsymbol{x}^*$. Otherwise, the KKT conditions (4.29) do not apply as the following example shows.

Example 4.3 *Consider the problem*

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^2} \quad & -x_1 && \text{(4.30a)} \\ \text{s.t.} \quad & h_1(\boldsymbol{x}) = x_1^3 - x_2 \leq 0 && \text{(4.30b)} \\ & h_2(\boldsymbol{x}) = x_1^3 + x_2 \leq 0\,. && \text{(4.30c)} \end{aligned}$$

Figure 4.5 shows the admissible set $\mathcal{X}_{ad}$ at the optimal point $\boldsymbol{x}^* = [0,0]^\mathsf{T}$ at which both constraints are active. The gradients of the cost function f and the constraints at point $\boldsymbol{x}^*$ are

$$\nabla f(\boldsymbol{x}^*) = \begin{bmatrix} -1 \\ 0 \end{bmatrix}, \quad \nabla h_1(\boldsymbol{x}^*) = \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \quad \nabla h_2(\boldsymbol{x}^*) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{4.31}$$

The KKT conditions (4.29a) for this problem read

$$\nabla_x L(\boldsymbol{x}^*, \boldsymbol{\mu}^*) = \begin{bmatrix} -1 \\ 0 \end{bmatrix} + \mu_1^* \begin{bmatrix} 0 \\ -1 \end{bmatrix} + \mu_2^* \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \mathbf{0}\,. \tag{4.32}$$

This set of equations has no solution because the constraint qualification (4.26) is violated at the optimal point $\boldsymbol{x}^*$.

Exercise 4.2 *Compute the optimal solution $\boldsymbol{x}^*$ for the optimization problem*

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & f(\boldsymbol{x}) = (x_1-2)^2 + \left(\frac{x_2-2}{3}\right)^2 && \text{(4.33a)}\\ \text{s.t.} \quad & h_1(\boldsymbol{x}) = 2x_1 + x_2 - 2 \leq 0 && \text{(4.33b)}\\ & h_2(\boldsymbol{x}) = x_1 - x_2 - 1 \ \leq 0 && \text{(4.33c)} \end{aligned}$$

by investigating the KKT conditions (4.29) for the different cases of active/inactive constraints. Mark the different candidate points in Figure 4.6.

Exercise 4.3 *The following optimization problem is given in dependence of the parameter s*

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \left(x_1 - \frac{3}{2}\right)^2 + \left(x_2 - s\right)^4 \quad \text{s.t.} \quad \begin{bmatrix} 1-x_1-x_2 \\ 1-x_1+x_2 \\ 1+x_1-x_2 \\ 1+x_1+x_2 \end{bmatrix} \geq \boldsymbol{0}\,. \tag{4.34a}$$

Determine the values of s for which the point $\boldsymbol{x}^* = [1, 0]^\mathsf{T}$ satisfies the KKT conditions (4.29).

4.2.3 Second-order optimality conditions

Theorem 4.2 (Necessary second-order optimality conditions) *Let $\boldsymbol{x}^*$ be a local solution of the optimization problem (4.24) and let the constraint qualification (4.26) be satisfied at point $\boldsymbol{x}^*$ such that the KKT conditions (4.29) hold with associated Lagrange multipliers $\boldsymbol{\lambda}^*$ and $\boldsymbol{\mu}^*$. If f, g_i, and h_i are twice continuously differentiable, then*

$$\boldsymbol{s}^\mathsf{T} \nabla^2_{\boldsymbol{x}\boldsymbol{x}} L(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)\, \boldsymbol{s} \geq 0 \quad \forall \boldsymbol{s} \in \mathcal{C}(\boldsymbol{x}^*, \boldsymbol{\mu}^*) \tag{4.35}$$

with

$$\begin{aligned} \mathcal{C}(\boldsymbol{x}^*, \boldsymbol{\mu}^*) = \Big\{ \boldsymbol{s} \in \mathbb{R}^n : \ & \nabla g_i(\boldsymbol{x}^*)^\mathsf{T} \boldsymbol{s} = 0, \quad i = 1, \dots, p \\ & \nabla h_i(\boldsymbol{x}^*)^\mathsf{T} \boldsymbol{s} = 0, \quad i \in \mathcal{A}(\boldsymbol{x}^*),\ \mu_i^* > 0 \\ & \nabla h_i(\boldsymbol{x}^*)^\mathsf{T} \boldsymbol{s} \leq 0, \quad i \in \mathcal{A}(\boldsymbol{x}^*),\ \mu_i^* = 0 \Big\}\,. \end{aligned} \tag{4.36}$$

The condition (4.35) has similarity with the unconstrained case (Theorem 3.1), where the Hessian $\nabla^2 f(\boldsymbol{x}^*)$ at point $\boldsymbol{x}^*$ has to be positive semidefinite, i.e. $\boldsymbol{s}^\mathsf{T} \nabla^2 f(\boldsymbol{x}^*) \boldsymbol{s} \geq 0$ for all $\boldsymbol{s} \in \mathbb{R}^n$, $\boldsymbol{s} \neq \boldsymbol{0}$. In the constrained case, the set $\mathcal{C}(\boldsymbol{x}^*, \boldsymbol{\mu}^*)$ only contains admissible directions $\boldsymbol{s}$ that comply with the constraints (4.24b) and (4.24c). In particular, only directions $\boldsymbol{s}$ are considered for which the gradients do not provide enough information whether the cost decreases or increases in direction of $\boldsymbol{s}$.[2]

[2] For active inequality constraints with $\mu_i^* > 0$, all admissible and sufficiently small directions $\boldsymbol{s}$ with $\nabla h_i(\boldsymbol{x}^*)^\mathsf{T} \boldsymbol{s} < 0$ lead to an increase of the cost $f(\boldsymbol{x}^* + \boldsymbol{s})$, see Section 4.2.4.

Theorem 4.3 (Sufficient optimality conditions) *Let $\boldsymbol{x}^*$ be a feasible point of the optimization problem (4.24) with associated Lagrange multipliers $\boldsymbol{\lambda}^*$ and $\boldsymbol{\mu}^*$ such that the KKT conditions (4.29) are satisfied. If f, g_i, and h_i are twice continuously differentiable and*

$$\boldsymbol{s}^\mathsf{T} \nabla^2_{\boldsymbol{xx}} L(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)\, \boldsymbol{s} > 0 \quad \forall \boldsymbol{s} \in \mathcal{C}(\boldsymbol{x}^*, \boldsymbol{\mu}^*)\,,\ \boldsymbol{s} \neq \boldsymbol{0} \tag{4.37}$$

with $\mathcal{C}(\boldsymbol{x}^*, \boldsymbol{\mu}^*)$ in (4.36), then $\boldsymbol{x}^*$ is a strict local minimum point.

If the Hessian $\nabla^2_{\boldsymbol{xx}} L$ is positive definite, i.e.

$$\boldsymbol{s}^\mathsf{T} \nabla^2_{\boldsymbol{xx}} L(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)\, \boldsymbol{s} > 0 \quad \forall \boldsymbol{s} \in \mathbb{R}^n\,,\ \boldsymbol{s} \neq \boldsymbol{0} \tag{4.38}$$

then the sufficient condition (4.37) is always satisfied without the necessity to compute the set $\mathcal{C}(\boldsymbol{x}^*, \boldsymbol{\mu}^*)$. This is in particular the case for quadratic problems with positive definite Hessian $\nabla^2 f(\boldsymbol{x})$ of the cost function.

For convex problems, the evaluation of the second-order optimality conditions can be avoided, because the KKT conditions (4.29) are necessary and sufficient in this case.

Theorem 4.4 (Sufficient optimality conditions for convex problems) *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ and $h_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \dots, q$ are convex and continuously differentiable functions and that $g_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \dots, p$ is continuously differentiable and linear. If a point $\boldsymbol{x}^*$ with Lagrange multipliers $\boldsymbol{\lambda}^*$ and $\boldsymbol{\mu}^*$ satisfies the KKT conditions (4.29), then $\boldsymbol{x}^*$ is a global minimum point of the optimization problem (4.24).*

Exercise 4.4 *Verify the sufficient optimality conditions of Theorem 4.3 and Theorem 4.4 for the problem (4.33).*

4.2.4 Interpretation of Lagrange multipliers

The meaning of the Lagrange multipliers was motivated by the introductory examples in Section 4.1. There exists another important interpretation that is often used in post-optimization analysis. To this end, we consider an optimization problem with a single equality constraint

$$\begin{aligned} &\min_{\boldsymbol{x} \in \mathbb{R}^n} \ f(\boldsymbol{x}) \\ &\text{s.t.} \ \ g(\boldsymbol{x}) = 0\,. \end{aligned}$$

The stationarity condition for an optimal solution $\boldsymbol{x}^*$ reads

$$\nabla_{\boldsymbol{x}} L(\boldsymbol{x}^*, \lambda^*) = \nabla f(\boldsymbol{x}^*) + \lambda^* \nabla g(\boldsymbol{x}^*) = \boldsymbol{0}\,. \tag{4.39}$$

To investigate the relation between the Lagrange multiplier and the optimal solution $\boldsymbol{x}^*$, we consider a small perturbation of the constraint

$$g(\boldsymbol{x}) = \varepsilon \quad \Rightarrow \quad \bar{g}(\boldsymbol{x}(\varepsilon), \varepsilon) = g(\boldsymbol{x}(\varepsilon)) - \varepsilon = 0\,. \tag{4.40}$$

The variables $\boldsymbol{x}(\varepsilon)$ now depend on ε with the original solution $\boldsymbol{x}^* = \boldsymbol{x}(0)$ for $\varepsilon = 0$. Since $\bar{g}(\boldsymbol{x}(\varepsilon), \varepsilon) = 0$ must hold for all ε, we can take the total derivative

$$\frac{\partial \bar{g}}{\partial \varepsilon} + \nabla_x \bar{g}(\boldsymbol{x}(\varepsilon), \varepsilon)^\mathsf{T} \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}\varepsilon} = 0 \quad \Rightarrow \quad \nabla_x \bar{g}(\boldsymbol{x}(\varepsilon), \varepsilon)^\mathsf{T} \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}\varepsilon} = 1 \tag{4.41}$$

with $\partial \bar{g}/\partial \varepsilon = -1$. Of particular interest is the effect of ε on the minimal cost $f(\boldsymbol{x}^*)$, which can be expressed with $\bar{f}(\varepsilon) = f(\boldsymbol{x}(\varepsilon))$ as follows

$$\left.\frac{\mathrm{d}\bar{f}}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} = \nabla f(\boldsymbol{x}^*)^\mathsf{T} \left.\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} \overset{(4.39)}{=} -\lambda^* \nabla g(\boldsymbol{x}^*)^\mathsf{T} \left.\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} \overset{(4.41)}{\Rightarrow} \boxed{\left.\frac{\mathrm{d}\bar{f}}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} = -\lambda^* .} \tag{4.42}$$

Thus, the multiplier λ^* is a *direct measure for the rate of change (sensitivity)* of the optimal cost w.r.t. a variation of the equality constraint $g(\boldsymbol{x}^*) = 0$. Similar reasoning can be applied to an active inequality constraint

$$h(\boldsymbol{x}) \leq \varepsilon \quad \Rightarrow \quad \boxed{\left.\frac{\mathrm{d}\bar{f}}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} = -\mu^*} \tag{4.43}$$

with the optimal multiplier μ^* for $h(\boldsymbol{x}^*) = 0$. The interpretation is the same as in case of the equality constraint: the Lagrange multiplier μ^* of an active inequality constraint $h(\boldsymbol{x}) \leq 0$ at the optimal point $\boldsymbol{x}^*$ is the sensitivity of the optimal cost $f(\boldsymbol{x}^*)$ w.r.t. this constraint.

A large value of λ^* or μ^* thus indicates that a relaxation of this constraint can lead to a significant decrease of the cost function. This interpretation of the multipliers can be directly extended to the general case (4.24) with p equality constraints and q inequality constraints.

The equation (4.43) also helps to explain the sign condition (4.29d) of the KKT conditions. If a multiplier at an optimal point were negative, i.e. $\mu^* < 0$, this would imply

$$\left.\frac{\mathrm{d}\bar{f}}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} > 0$$

and thus a reduction of the cost function $f(\boldsymbol{x}^*)$ could be achieved by leaving the active constraint into the interior of the admissible set (i.e. for $\varepsilon < 0$). In this case, however, $\boldsymbol{x}^*$ would not be the optimal solution, which implies that the multiplier must be non-negative ($\mu^* \geq 0$).

Exercise 4.5 *Interprete the multiplier values graphically for the problem (4.33).*

4.3 Linear optimization

Linear optimization (also known as *linear programming*) is a special case of the general non-linear problem (4.1) with linear cost function and constraints. Linear programming is not only popular because of its wide range of applications such as production planning or transportation problems but also due to its simplicity and the existence of reliable algorithms. In particular, linear problems are always convex, see Theorem 2.5 and 4.4. Hence, every local solution is also a global one.

4.3.1 Standard form

Most numerical algorithms of linear programming are based on the standard form of a linear optimization problem

$$\boxed{\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & \boldsymbol{c}^\mathsf{T}\boldsymbol{x} && \text{(4.44a)}\\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\,, \quad \boldsymbol{A}\in\mathbb{R}^{p\times n}, \quad \boldsymbol{b}\in\mathbb{R}^p && \text{(4.44b)}\\ & \boldsymbol{x} \geq \boldsymbol{0}\,. && \text{(4.44c)} \end{aligned}}$$

Every linear optimization problem can be transformed into the standard form. For instance, consider the linear problem

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & \boldsymbol{c}^\mathsf{T}\boldsymbol{x} && \text{(4.45a)}\\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\,, \quad \boldsymbol{A}\in\mathbb{R}^{q\times n}, \quad \boldsymbol{b}\in\mathbb{R}^q\,. && \text{(4.45b)} \end{aligned}$$

The inequality constraint is transformed into equality constraints and additional non-negativity constraints of the form (4.44c) by means of *slack variables* $\boldsymbol{z}$

$$\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \quad \Rightarrow \quad \boldsymbol{A}\boldsymbol{x} + \boldsymbol{z} = \boldsymbol{b}\,, \quad \boldsymbol{z} \geq \boldsymbol{0}\,. \tag{4.46}$$

Since the variables $\boldsymbol{x}$ so far are not constrained to be non-negative, we split $\boldsymbol{x}$ in non-negative and non-positive parts

$$\boldsymbol{x} = \boldsymbol{x}^+ - \boldsymbol{x}^-\,, \quad \boldsymbol{x}^-, \boldsymbol{x}^+ \geq \boldsymbol{0}\,. \tag{4.47}$$

In summary, the problem (4.45) is transformed in the standard form (4.44)

$$\begin{aligned} \min_{\bar{\boldsymbol{x}}\in\mathbb{R}^{2n+q}} \quad & \begin{bmatrix}\boldsymbol{c}^\mathsf{T}, -\boldsymbol{c}^\mathsf{T}, \boldsymbol{0}^\mathsf{T}\end{bmatrix}\bar{\boldsymbol{x}} && \text{(4.48a)}\\ \text{s.t.} \quad & \begin{bmatrix}\boldsymbol{A}, -\boldsymbol{A}, \boldsymbol{I}\end{bmatrix}\bar{\boldsymbol{x}} = \boldsymbol{b}\,, \quad \bar{\boldsymbol{x}} = \begin{bmatrix}\boldsymbol{x}^+\\ \boldsymbol{x}^-\\ \boldsymbol{z}\end{bmatrix} && \text{(4.48b)}\\ & \bar{\boldsymbol{x}} \geq 0 && \text{(4.48c)} \end{aligned}$$

with the extended optimization vector $\bar{\boldsymbol{x}}$.

Exercise 4.6 *Transform the following linear problem into the standard form (4.44)*

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^3} \quad & -x_1 + 2x_2 + x_3 && \text{(4.49a)}\\ \text{s.t.} \quad & x_1 + x_3 = 1 && \text{(4.49b)}\\ & -x_1 + x_2 + x_3 \leq 1 && \text{(4.49c)}\\ & x_2, x_3 \geq 0\,. && \text{(4.49d)} \end{aligned}$$

4.3.2 Optimality conditions

In what follows, we assume that the optimization problem is already given in the standard form (4.44). With the Lagrangian function

$$L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \boldsymbol{c}^\mathsf{T}\boldsymbol{x} + \boldsymbol{\lambda}^\mathsf{T}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) + \boldsymbol{\mu}^\mathsf{T}(-\boldsymbol{x})$$

with $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_p]^\mathsf{T}$, $\boldsymbol{\mu} = [\mu_1, \ldots, \mu_n]^\mathsf{T}$ and its gradient

$$\nabla_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \boldsymbol{c} + \boldsymbol{A}^\mathsf{T}\boldsymbol{\lambda} - \boldsymbol{\mu}\,,$$

the optimality conditions for the linear problem (4.44) are a special case of the KKT conditions (4.29)

$$\boldsymbol{c} + \boldsymbol{A}^\mathsf{T}\boldsymbol{\lambda}^* - \boldsymbol{\mu}^* = \boldsymbol{0} \tag{4.50a}$$
$$\boldsymbol{A}\boldsymbol{x}^* - \boldsymbol{b} = \boldsymbol{0} \tag{4.50b}$$
$$\boldsymbol{x}^* \geq \boldsymbol{0} \tag{4.50c}$$
$$\mu_i^* \geq 0\,, \quad i = 1, \ldots, n \tag{4.50d}$$
$$\mu_i^*(-x_i^*) = \mu_i^* x_i^* = 0\,, \quad i = 1, \ldots, n\,. \tag{4.50e}$$

Due to the convexity of the linear problem (4.44), the optimality conditions (4.50) are necessary and sufficient for a global minimum. This also means that the second-order optimality conditions do not need to be evaluated, which do not provide further insight anyway, since the Hessian matrix $\nabla_{\boldsymbol{x}\boldsymbol{x}} L = \boldsymbol{0}$ is the zero matrix.

4.3.3 Simplex method

The *simplex method* was developed by *George B. Danzig* in 1948 and marked the start of modern numerical optimization together with the advent of digital computers. The simplex algorithm exploits the geometric property that the admissible set

$$\mathcal{X}_{ad} = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\,,\ \boldsymbol{x} \geq \boldsymbol{0}\} \tag{4.51}$$

of linear optimization problems forms a convex *polyhedron* with a finite number of vertices, see Figure 4.7. The optimal solution $\boldsymbol{x}^*$ – if it exists – always lies on the boundary of $\mathcal{X}_{ad}$ and, in particular, on a vertex of the polyhedron if the solution $\boldsymbol{x}^*$ is unique. The simplex method starts from an initial vertex of the polyhedron and then proceeds along the edges towards the optimal solution $\boldsymbol{x}^*$.

Definition 4.3 (Simplex method – Notations)

Basis and basis matrix:	*Let $\mathcal{B}$ be a subset of $\{1, \ldots, n\}$ with p elements. If the $(p \times p)$-matrix $\boldsymbol{B} = [\boldsymbol{A}_i]_{i\in\mathcal{B}}$ is regular, then $\mathcal{B}$ is a* basis *and $\boldsymbol{B}$ is the* basis matrix. [3]
Basic variables:	$\boldsymbol{x}_B = [x_i]_{i\in\mathcal{B}} = \boldsymbol{B}^{-1}\boldsymbol{b}$
Non-basic variables:	$\boldsymbol{x}_N = [x_i]_{i\in\mathcal{N}} = \boldsymbol{0}$ *with* $\mathcal{N} = \{1, \ldots, n\}\backslash\mathcal{B}$
Basic feasible solution:	$\boldsymbol{x}$ *with* $\boldsymbol{x}_B = \boldsymbol{B}^{-1}\boldsymbol{b} \geq \boldsymbol{0}$ *and* $\boldsymbol{x}_N = \boldsymbol{0}$.
Non-degenerate basic feasible solution:	$\boldsymbol{x}$ *with* $\boldsymbol{x}_B = \boldsymbol{B}^{-1}\boldsymbol{b} > \boldsymbol{0}$ *and* $\boldsymbol{x}_N = \boldsymbol{0}$.

The equality constraints (4.44b) with $p < n$ form an underdetermined set of equations from which p elements, the basic variables $\boldsymbol{x}_B$, can be computed. The remaining $n - p$ elements are the non-basic variables that are set to zero, i.e. $\boldsymbol{x}_N = \boldsymbol{0}$.

[3] The vector $\boldsymbol{A}_i$ denotes the i-th column of the matrix $\boldsymbol{A}$.

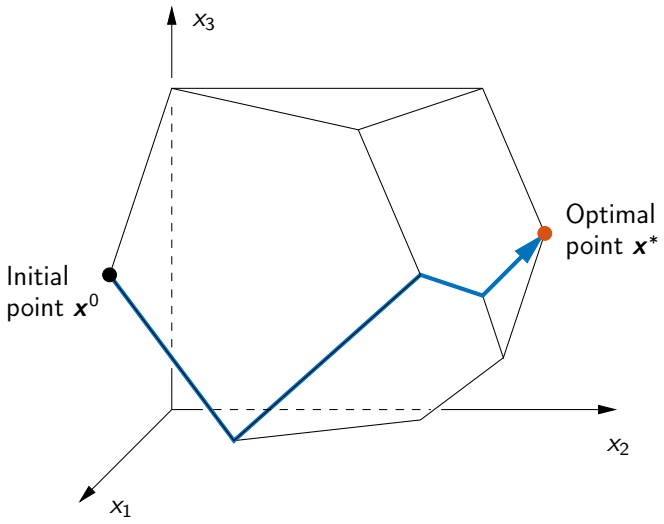


Figure 4.7: Illustration of the simplex method.

Under the assumption that the $(p \times n)$-matrix $\boldsymbol{A}$ has row rank p, it is always possible to determine a basis $\mathcal{B}$ such that the associated basis matrix $\boldsymbol{B} = [\boldsymbol{A}_i]_{i \in \mathcal{B}}$ is regular. The equality constraints (4.44b) can then be split in $\boldsymbol{x}_B$ and $\boldsymbol{x}_N$

$$\boldsymbol{b} = \boldsymbol{B}\boldsymbol{x}_B + \boldsymbol{N}\boldsymbol{x}_N\,, \quad \boldsymbol{N} = [\boldsymbol{A}_i]_{i \in \mathcal{N}}\,, \tag{4.52}$$

in order to compute $\boldsymbol{x}_B$ by means of the regular basis matrix $\boldsymbol{B}$

$$\boldsymbol{x}_B = \boldsymbol{B}^{-1}(\boldsymbol{b} - \boldsymbol{N}\boldsymbol{x}_N) = \boldsymbol{B}^{-1}\boldsymbol{b} \quad \text{with} \quad \boldsymbol{x}_N = \boldsymbol{0}\,. \tag{4.53}$$

If $\boldsymbol{x}_B \geq \boldsymbol{0}$, then the basic variables $\boldsymbol{x}_B$ and $\boldsymbol{x}_N = \boldsymbol{0}$ form a basic feasible solution. The initialization phase of the simplex method consists in finding a basic solution with $\boldsymbol{x}_B \geq \boldsymbol{0}$, which is addressed at the end of the section. Once this initial point $\boldsymbol{x}$ is determined, the subsequent iterations of the simplex method always produce basic feasible solutions.

Example 4.4 *We consider the example*

$$\min_{\boldsymbol{x} \in \mathbb{R}^4} \boldsymbol{c}^\mathsf{T}\boldsymbol{x}\,, \qquad \boldsymbol{c}^\mathsf{T} = [1, 2, 3, 4] \tag{4.54a}$$

$$\text{s.t.} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\,, \quad \boldsymbol{A} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & -3 \end{bmatrix}\,, \quad \boldsymbol{b} = \begin{bmatrix} 1 \\ \frac{1}{2} \end{bmatrix} \tag{4.54b}$$

$$\boldsymbol{x} \geq \boldsymbol{0}\,. \tag{4.54c}$$

The following table shows various bases $\mathcal{B}$ with no basic feasible solution existing in the last case. The choice $\{1, 3\}$ does not form a basis because $\boldsymbol{B}$ would be singular.

Basis $\mathcal{B}$	*Basis matrix* $\boldsymbol{B}$	$\boldsymbol{x}_B = \boldsymbol{B}^{-1}\boldsymbol{b}$	$\boldsymbol{x}_N = \boldsymbol{0}$	*Basic feasible solution?*
$\mathcal{B} = \{1, 2\}$	$\boldsymbol{B} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$	$x_1 = 0.5$ $x_2 = 0.5$	$x_3 = 0$ $x_4 = 0$	*yes*
$\mathcal{B} = \{1, 4\}$	$\boldsymbol{B} = \begin{bmatrix} 1 & 1 \\ 1 & -3 \end{bmatrix}$	$x_1 = 0.875$ $x_4 = 0.125$	$x_2 = 0$ $x_3 = 0$	*yes*
$\mathcal{B} = \{2, 4\}$	$\boldsymbol{B} = \begin{bmatrix} 1 & 1 \\ 0 & -3 \end{bmatrix}$	$x_2 = 7/6$ $x_4 = -1/6$	$x_1 = 0$ $x_3 = 0$	*no*

The principle approach of the simplex method consists in iteratively verifying the optimality conditions (4.50) and the subsequent determination of a new basic feasible solution $\boldsymbol{x}^+$ with $f(\boldsymbol{x}^+) < f(\boldsymbol{x})$.

Step 1: Verification of optimality conditions (4.50)

A basic feasible solution (4.53), i.e. $\boldsymbol{x}_B \geq \boldsymbol{0}$ and $\boldsymbol{x}_N = \boldsymbol{0}$, already satisfies the KKT condition (4.50b) and (4.50c). The stationarity condition (4.50a) is separated in terms of the basis $\mathcal{B}$ and non-basis $\mathcal{N}$

$$\begin{bmatrix} \boldsymbol{c}_B \\ \boldsymbol{c}_N \end{bmatrix} + \begin{bmatrix} \boldsymbol{B}^\mathsf{T} \\ \boldsymbol{N}^\mathsf{T} \end{bmatrix} \boldsymbol{\lambda} = \begin{bmatrix} \boldsymbol{\mu}_B \\ \boldsymbol{\mu}_N \end{bmatrix} . \tag{4.55}$$

The complementarity condition is already satisfied for $\boldsymbol{\mu}_N$ (due to $\boldsymbol{x}_N = \boldsymbol{0}$) and for the basic variables we set the multipliers to $\boldsymbol{\mu}_B = \boldsymbol{0}$. The remaining vectors $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}_N$ in (4.55) become

$$\boldsymbol{\lambda} = -(\boldsymbol{B}^\mathsf{T})^{-1} \boldsymbol{c}_B \,, \quad \boldsymbol{\mu}_N = \boldsymbol{c}_N + \boldsymbol{N}^\mathsf{T} \boldsymbol{\lambda} \,. \tag{4.56}$$

The last remaining KKT condition is $\boldsymbol{\mu}_N \geq \boldsymbol{0}$. If this condition is satisfied by (4.56), the basic solution $(\boldsymbol{x}_B, \boldsymbol{x}_N)$ is the optimal solution $\boldsymbol{x}^*$ and the algorithm terminates.

Step 2: Determination of new basic variable

If one or more multipliers $\mu_i \,, i \in \mathcal{N}$ are negative, then $\boldsymbol{x}$ is not the optimal solution. In this case, the simplex method constructs a new basic feasible solution $(\boldsymbol{x}_B^+, \boldsymbol{x}_N^+)$ by swapping two elements of $\boldsymbol{x}_B$ and $\boldsymbol{x}_N$ with the indices $r \in \mathcal{B}$ and $s \in \mathcal{N}$. We set $x_r^+ = 0$ and allow x_s^+ to raise in value such that

$$\boldsymbol{x}_B^+ = [\ldots, x_r^+ = 0, \ldots]^\mathsf{T} \,, \quad \boldsymbol{x}_N^+ = [\ldots, x_s^+ \geq 0, \ldots]^\mathsf{T} \,. \tag{4.57}$$

At first, an appropriate element x_s with $s \in \mathcal{N}$ and $\mu_s < 0$ must be determined. A reasonable choice is to pick the element with the most negative multiplier μ_i corresponding to the strongest violation of the optimality conditions $\boldsymbol{\mu}_N \geq \boldsymbol{0}$, i.e.

$$\mu_s = \min_{i \in \mathcal{N} | \mu_i < 0} \; \mu_i \quad \text{with associated index} \;\; s \in \mathcal{N} \,. \tag{4.58}$$

Step 3: Determination of new non-basic variable

In order to choose the index r of the new non-basic variable x_r^+ from the previous basic variables $\boldsymbol{x}_B$, we have to determine the value of x_s^+. A starting point in this regard is the equality constraint (4.52) that must be satisfied for $\boldsymbol{x}_B^+$ and $\boldsymbol{x}_N^+$. With (4.53) we get for the new basic variable

$$\begin{aligned} \boldsymbol{x}_B^+ &= \boldsymbol{B}^{-1}(\boldsymbol{b} - \boldsymbol{N}\boldsymbol{x}_N^+) \\ &= \boldsymbol{x}_B - \boldsymbol{B}^{-1}\boldsymbol{A}_s \, x_s^+ \quad \text{with} \quad \boldsymbol{x}_B = \boldsymbol{B}^{-1}\boldsymbol{b} \\ &= \boldsymbol{x}_B - \boldsymbol{d} \, x_s^+ \qquad\qquad\quad \boldsymbol{d} = \boldsymbol{B}^{-1}\boldsymbol{A}_s \,, \end{aligned} \tag{4.59}$$

where $\boldsymbol{A}_s$ denotes the s-th column of the matrix $\boldsymbol{A}$. The vector $\boldsymbol{d}$ corresponds to the direction of change of the previous basic variable $\boldsymbol{x}_B$ if x_s^+ is increased.

In particular, if $d_i > 0$ for an element of $\boldsymbol{d}$, then an increase of x_s^+ leads to a reduction of the respective element $x_{B,i}^+$ of $\boldsymbol{x}_B^+$. This means that x_s^+ can only be increased until an element of

Given (by initialization or previous iteration step):

$\mathcal{B},\ \mathcal{N},\ \boldsymbol{x}_B = \boldsymbol{B}^{-1}\boldsymbol{b} \geq \boldsymbol{0},\ \boldsymbol{x}_N = \boldsymbol{0}$

KKT conditions:

$\boldsymbol{\lambda} \leftarrow -(\boldsymbol{B}^\mathsf{T})^{-1}\boldsymbol{c}_B$ Lagrange multipliers
$\boldsymbol{\mu}_N \leftarrow \boldsymbol{c}_N + \boldsymbol{N}^\mathsf{T}\boldsymbol{\lambda}$
if $\boldsymbol{\mu}_N \geq \boldsymbol{0}$ **then**
  return $\boldsymbol{x}_B^* = \boldsymbol{x}_B\,,\ \boldsymbol{x}_N^* = \boldsymbol{x}_N$ Optimal solution
end

Next basic feasible solution:

Index $s \in \mathcal{N}$ with $\mu_s < 0$ from (4.58)
Direction $\boldsymbol{d} = \boldsymbol{B}^{-1}\boldsymbol{A}_s$
if $\boldsymbol{d} \leq \boldsymbol{0}$ **then**
  stop Unconstrained problem
end
x_s^+ and index r from (4.60)
$\boldsymbol{x}_B^+ = \boldsymbol{x}_B - \boldsymbol{d}\,x_s^+$ New basic variables
$\boldsymbol{x}_N^+ = [0,\ldots,0,x_s^+,0,\ldots,0]^\mathsf{T}$ New non-basic variables

New basis:

$\mathcal{B} \leftarrow \mathcal{B}\backslash\{r\} \cup \{s\}\,, \quad \mathcal{N} \leftarrow \{1,\ldots,n\}\backslash\mathcal{B}$

Table 4.1: One iteration of the simpex methtod.

$\boldsymbol{x}_B^+$ reaches the boundary of the constraint $\boldsymbol{x}_B^+ \geq \boldsymbol{0}$. According to (4.59), a basic variable $x_{B,i}^+$ becomes zero for $x_s^+ = x_{B,i}^+/d_i$. Thus, x_s^+ can be determined from

$$x_s^+ = \min_{i|d_i>0} \frac{x_{B,i}}{d_i} \quad \text{with associated index} \quad r \in \mathcal{B}\,, \tag{4.60}$$

such that the index r denotes the basic variable x_r^+ that first reaches the boundary of the constraint $\boldsymbol{x}_B^+ \geq \boldsymbol{0}$. With x_s^+, we get the new basic variable $\boldsymbol{x}_B^+$ from (4.59), where the position of $x_r^+ = 0$ in vector $\boldsymbol{x}_B^+$ is exactly the index i that solves (4.60).

After determining the new basic feasible point (4.57), the basis $\mathcal{B}$ is updated accordingly and the indices $r \in \mathcal{B}$ and $s \in \mathcal{N}$ are swapped before the next simplex iteration starts. Table 4.1 summarizes the operations of a simplex iteration.

The increase of the non-basic variable x_s^+ corresponds to the movement along an edge of the polyhedron until the next vertex is reached, see Figure 4.7. In the non-degenerate case (Definition 4.3), each step of the simplex method leads to a reduction of the cost and the algorithm converges to the optimal point $\boldsymbol{x}^*$ in a finite number of steps. An exception is the case that the vector $\boldsymbol{d}$ contains no positive elements. In this case, x_s^+ can be chosen arbitrarily large without violating the constraint $\boldsymbol{x}_B^+ \geq \boldsymbol{0}$. This implies that the admissible region $\mathcal{X}_{ad}$ of the problem is unbounded and no finite solution exists.

A computational drawback of the simplex method is that the number of iterations in the worst case corresponds to the number of vertices of the polyhedron that grows exponentially with the problem dimension $\boldsymbol{x} \in \mathbb{R}^n$. However, in most practical applications, the simplex method converges significantly faster then the worst case scenario.

Example 4.5 *We consider again problem (4.54). The simplex method is initialized with*

$$\mathcal{B} = \{1,2\}, \quad \boldsymbol{B} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}, \quad \boldsymbol{x}_B = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}, \quad \boldsymbol{c}_B = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$
$$\mathcal{N} = \{3,4\}, \quad \boldsymbol{N} = \begin{bmatrix} 1 & 1 \\ 1 & -3 \end{bmatrix}, \quad \boldsymbol{x}_N = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{c}_N = \begin{bmatrix} 3 \\ 4 \end{bmatrix}.$$

The evaluation of the KKT conditions acccording to (4.56) yields

$$\boldsymbol{\lambda} = -(\boldsymbol{B}^\mathsf{T})^{-1}\boldsymbol{c}_B = \begin{bmatrix} -2 \\ 1 \end{bmatrix}, \quad \boldsymbol{\mu}_N = \boldsymbol{c}_N + \boldsymbol{N}^\mathsf{T}\boldsymbol{\lambda} = \begin{bmatrix} 2 \\ -1 \end{bmatrix}.$$

Obviously, the point is not optimal due to $\mu_{N,2} = \mu_4 < 0$. One step of the simplex method leads to

$$(4.58): \quad s = 4, \quad \mu_s = -1, \quad \boldsymbol{d} = \boldsymbol{B}^{-1}A_4 = \begin{bmatrix} 0 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} 1 \\ -3 \end{bmatrix} = \begin{bmatrix} -3 \\ 4 \end{bmatrix}$$
$$(4.60): \quad r = 2, \quad x_s^+ = \frac{x_{B,2}}{d_2} = 0.125$$
$$(4.59): \quad \boldsymbol{x}_B^+ = \boldsymbol{x}_B - \boldsymbol{d}x_s^+ = \begin{bmatrix} 0.875 \\ 0 \end{bmatrix} = \begin{bmatrix} x_1^+ \\ x_2^+ \end{bmatrix}, \quad \boldsymbol{x}_N^+ = \begin{bmatrix} 0 \\ x_s^+ \end{bmatrix} = \begin{bmatrix} 0 \\ 0.125 \end{bmatrix} = \begin{bmatrix} x_3^+ \\ x_4^+ \end{bmatrix}.$$

For the next simplex iteration, the elements of $\mathcal{B}$ and $\mathcal{N}$ with the indices $r \in \mathcal{B}$ and $s \in \mathcal{N}$ are swapped, leading to the new basic feasible solution

$$\mathcal{B} = \{1,4\}, \quad \boldsymbol{B} = \begin{bmatrix} 1 & 1 \\ 1 & -3 \end{bmatrix}, \quad \boldsymbol{x}_B = \begin{bmatrix} 0.875 \\ 0.125 \end{bmatrix}, \quad \boldsymbol{c}_B = \begin{bmatrix} 1 \\ 4 \end{bmatrix}$$
$$\mathcal{N} = \{3,2\}, \quad \boldsymbol{N} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}, \quad \boldsymbol{x}_N = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{c}_N = \begin{bmatrix} 3 \\ 2 \end{bmatrix}.$$

The entries in $\mathcal{N}$, $\boldsymbol{N}$, $\boldsymbol{x}_B$, and $\boldsymbol{c}_N$ can be reordered in ascending order if desired. The multipliers are again computed with (4.56)

$$\boldsymbol{\lambda} = -(\boldsymbol{B}^\mathsf{T})^{-1}\boldsymbol{c}_B = \begin{bmatrix} -1.75 \\ 0.75 \end{bmatrix}, \quad \boldsymbol{\mu}_N = \boldsymbol{c}_N + \boldsymbol{N}^\mathsf{T}\boldsymbol{\lambda} = \begin{bmatrix} 2 \\ 0.25 \end{bmatrix} > \boldsymbol{0}.$$

Both entries of $\boldsymbol{\mu}_N = [\mu_{N,1}, \mu_{N,2}]^\mathsf{T} = [\mu_3, \mu_2]^\mathsf{T}$ are positive, which shows that the point $\boldsymbol{x}^* = [0.875, 0, 0, 0.125]^\mathsf{T}$ is the optimal solution.

Initialization phase

If a basic feasible initial point cannot easily be found, the original problem (4.44) can be modified by introducing additional variables $\boldsymbol{z}$ along with the auxiliary optimization problem

$$\begin{aligned} \min_{(\boldsymbol{x},\boldsymbol{z})\in\mathbb{R}^{n+p}} \quad & \sum_{i=1}^{p} z_i \\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{x} + \boldsymbol{E}\boldsymbol{z} = \boldsymbol{b} \qquad \text{with} \qquad E_{ii} = \begin{cases} 1 & \text{if} \quad b_i \geq 0 \\ -1 & \text{if} \quad b_i < 0 \end{cases} \\ & \boldsymbol{x} \geq \boldsymbol{0}, \quad \boldsymbol{z} \geq \boldsymbol{0}. \end{aligned} \tag{4.61}$$

A basic feasible solution for this problem is

$$\boldsymbol{x} = \boldsymbol{0}\,, \quad z_i = |b_i|\,, \quad i = 1, \ldots, p\,, \tag{4.62}$$

which shows that the simplex method can always be applied to the modified problem (4.61) with the initialization (4.62).

If the simplex methods terminates with an optimal solution $\boldsymbol{x}^*$ and $\boldsymbol{z}^* = \boldsymbol{0}$, then $\boldsymbol{x}^*$ is a basic feasible solution for the original problem (4.44). The basis $\mathcal{B}^*$ and the basis matrix $\boldsymbol{B}^*$ of the last iteration can be used to initialize the first simplex iteration for the original problem (4.44). If not all elements z_i^* of $\boldsymbol{z}^*$ are identical zero, then the original problem (4.44) has no solution.

4.3.4 Interior point method

A drawback of the simplex method is that the maximum number of iterations grows exponentially with the problem size. Although the performance is usually better in practice, the number of iterations nevertheless can grow significantly for complex or high-dimensional problems. An alternative are *interior point methods* that basically go back to *Karmarkar's projection algorithm* from 1984. Interior point methods have a polynomial complexity w.r.t. the problem size n in contrast to the exponential complexity of the simplex method, which makes them particularly favorable for large-scale problems.

General approach

In principle, the KKT conditions (4.50) could be solved directly with a Newton method to determine an optimal point $\boldsymbol{x}^*$. This idea, however, would result in a very slow convergence as the step size of the Newton method would tend to very small values in order to satisfy the constraints $x_i \geq 0$ and $\mu_i \geq 0$ in each iteration.

The basic idea of interior point methods is to relax the complementary condition (4.50e) by a *barrier parameter* τ, i.e. $x_i \mu_i = \tau$. The reason for this designation will become clear later on. In vector notation, we get the nonlinear set of equations

$$\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \begin{bmatrix} \boldsymbol{c} + \boldsymbol{A}^\mathsf{T}\boldsymbol{\lambda} - \boldsymbol{\mu} \\ \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \\ \boldsymbol{X}\boldsymbol{M}\boldsymbol{e} \end{bmatrix} - \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \\ \tau \boldsymbol{e} \end{bmatrix} = \boldsymbol{0}\,, \quad (\boldsymbol{x}, \boldsymbol{\mu}) \geq \boldsymbol{0} \tag{4.63}$$

with the matrices $\boldsymbol{X} = \operatorname{diag}(\boldsymbol{x})$, $\boldsymbol{M} = \operatorname{diag}(\boldsymbol{\mu})$, and $\boldsymbol{e} = [1, \ldots, 1]^\mathsf{T}$. It can be shown that the optimal solution $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ is obtained for $\tau \to 0$

$$\lim_{\tau \to 0} (\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = (\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)\,. \tag{4.64}$$

In many practical implementations of the interior point method that solve (4.63) in an iterative manner, the so-called *duality measure*

$$\bar{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i \mu_i = \frac{1}{n} \boldsymbol{x}^\mathsf{T} \boldsymbol{\mu} \tag{4.65}$$

is used to adapt the barrier parameter in each iteration k in dependence of a reduction factor σ^k

$$\tau^k = \sigma^k \bar{\mu}^k\,, \quad \sigma^k \in [0, 1]\,. \tag{4.66}$$

The iteration scheme for solving the nonlinear set of equations (4.63), (4.66) reads

$$\begin{bmatrix} \boldsymbol{x}^{k+1} \\ \boldsymbol{\lambda}^{k+1} \\ \boldsymbol{\mu}^{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}^{k} \\ \boldsymbol{\lambda}^{k} \\ \boldsymbol{\mu}^{k} \end{bmatrix} + \alpha^k \boldsymbol{s}^k \quad \text{with} \quad \boldsymbol{s}^k = \begin{bmatrix} \Delta\boldsymbol{x}^{k} \\ \Delta\boldsymbol{\lambda}^{k} \\ \Delta\boldsymbol{\mu}^{k} \end{bmatrix}. \tag{4.67}$$

The search direction $\boldsymbol{s}^k$ is computed with Newton's method as the solution of the vector equation [4]

$$\underbrace{\begin{bmatrix} \boldsymbol{0} & \boldsymbol{A}^\mathsf{T} & -\boldsymbol{I} \\ \boldsymbol{A} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{M}^k & \boldsymbol{0} & \boldsymbol{X}^k \end{bmatrix}}_{\nabla \boldsymbol{F}(\boldsymbol{x}^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)} \begin{bmatrix} \Delta\boldsymbol{x}^{k} \\ \Delta\boldsymbol{\lambda}^{k} \\ \Delta\boldsymbol{\mu}^{k} \end{bmatrix} = \underbrace{\begin{bmatrix} -\boldsymbol{c} - \boldsymbol{A}^\mathsf{T}\boldsymbol{\lambda}^k + \boldsymbol{\mu}^k \\ -\boldsymbol{A}\boldsymbol{x}^k + \boldsymbol{b} \\ -\boldsymbol{X}^k\boldsymbol{M}^k\boldsymbol{e} + \sigma^k \bar{\mu}^k \boldsymbol{e} \end{bmatrix}}_{-\boldsymbol{F}(\boldsymbol{x}^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)} \tag{4.68}$$

with $\bar{\mu}^k$ from (4.65). In addition, the step size α^k in (4.67) must be determined by a suitable line search method that satisfies the strict constraints

$$\boldsymbol{x}^k > \boldsymbol{0}\,, \quad \boldsymbol{\mu}^k > \boldsymbol{0} \tag{4.69}$$

in each iteration k. This also explains the name *interior point method*. The reduction factor $\sigma^k \in [0, 1]$ also affects the step size α^k. Larger values of σ^k, i.e. $\sigma^k \approx 1$, typically imply a larger step α^k in the Newton direction $\boldsymbol{s}^k$ before the constraints $(\boldsymbol{x}^k, \boldsymbol{\mu}^k) > \boldsymbol{0}$ are violated.

There exist several more or less complex methods to compute α^k und σ^k, e.g. predictor-corrector schemes, that are not further detailed here. A critical point is the choice of the initial solution $(\boldsymbol{x}^0, \boldsymbol{\lambda}^0, \boldsymbol{\mu}^0)$ that must strictly satisfy the constraints $(\boldsymbol{x}^0, \boldsymbol{\mu}^0) > \boldsymbol{0}$ and the equations (4.63) at least with a not too large residual.

Interior point methods are well suited for large-scale problems. There are efficient algorithms that exploit the sparse structure of the linear set of equations (4.68). More details on this can e.g. be found in the textbooks [17, 14].

Relation to barrier methods

Interior point methods are closely related to barrier methods that use logarithmic penalty terms in the cost function to account for the constraints $\boldsymbol{x} \geq \boldsymbol{0}$, i.e.

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \quad \boldsymbol{c}^\mathsf{T}\boldsymbol{x} - \tau \sum_{i=1}^{n} \log x_i\,, \quad \tau > 0 \tag{4.70a}$$

$$\text{s.t.} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\,. \tag{4.70b}$$

A solution of the problem (4.70) must strictly satisfy the constraints $\boldsymbol{x} > \boldsymbol{0}$. Otherwise, the cost function (4.70a) would approach infinity due to the logarithmic functions. The penalty term τ affects the form of the barrier function as illustrated in Figure 4.8. For $\tau \to 0$, the log-terms approach a discontinuous behavior, whereas the penalty for interior points $\boldsymbol{x} > 0$ becomes negligible.

[4] Equation (4.68) results from the approximation $m_F^k(\boldsymbol{s}) = 0$ of (4.63) with the linear model $m_F^k(\boldsymbol{s}) = \boldsymbol{F}(\boldsymbol{x}^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k) + \nabla \boldsymbol{F}(\boldsymbol{x}^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)\,\boldsymbol{s}$. The computation of the Newton direction $\boldsymbol{s}$ is similar to the approach in Section 3.3.4, where a quadratic model (3.32) was derived for the cost function and the stationarity condition $\nabla m^k(\boldsymbol{s}) = 0$ for its minimum was a linear model in $\boldsymbol{s}$.

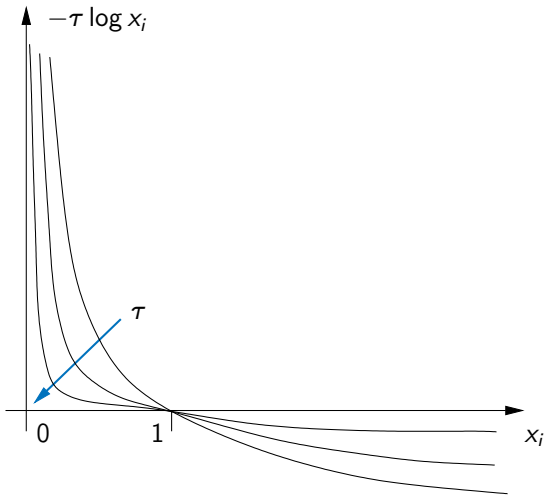


Figure 4.8: Logarithmic barrier function for different values of τ.

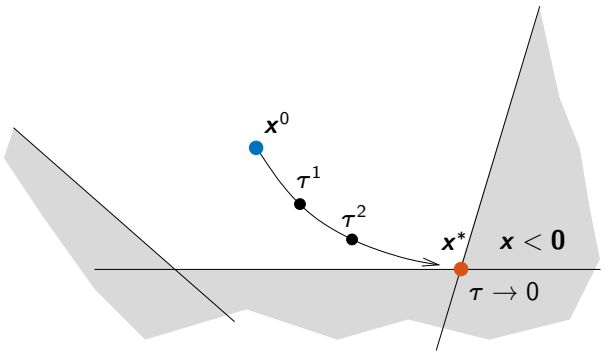


Figure 4.9: Central path for decreasing τ^k.

With the Lagrangian function

$$L(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{c}^\mathrm{T}\boldsymbol{x} - \tau \sum_{i=1}^{n} \log x_i + \boldsymbol{\lambda}^\mathrm{T}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}),$$

the KKT conditions (4.50a), (4.50b) for the barrier problem become

$$\begin{aligned} c_i - \frac{\tau}{x_i} + \boldsymbol{A}_i^\mathrm{T}\boldsymbol{\lambda} = 0, \quad i = 1, \dots, n \\ \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned} \tag{4.71}$$

with $\boldsymbol{A}_i$ denoting the i-th column of matrix $\boldsymbol{A}$. When we introduce the new variables $\mu_i = \tau / x_i$, we obtain the same set of equations as in (4.63)

$$\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \begin{bmatrix} \boldsymbol{c} + \boldsymbol{A}^\mathrm{T}\boldsymbol{\lambda} - \boldsymbol{\mu} \\ \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \\ \boldsymbol{X}\boldsymbol{M}\boldsymbol{e} \end{bmatrix} - \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \\ \tau \boldsymbol{e} \end{bmatrix} = \boldsymbol{0}, \quad (\boldsymbol{x}, \boldsymbol{\mu}) \geq \boldsymbol{0}.$$

This shows that the optimality conditions of the interior point method and the logarithmic barrier method are equivalent via the introduction of the new variables $\mu_i = \tau / x_i$.

Pure barrier methods as in (4.70) are numerically sensitive, because the terms in the KKT condition (4.71) can become very large, if $\boldsymbol{x}$ approaches the boundary of $\boldsymbol{x} \geq \boldsymbol{0}$ in the limit $\tau \to 0$. The formulation of the interior point methods is favorable in this regard as it can be shown that the set of equations (4.63) is better conditioned for $\tau \to 0$ than the optimality conditions (4.71) of barrier methods.

Nevertheless, the behavior of the barrier method can be used to illustrate the interior point method. Each value $\tau^k > 0$ defines a solution $(\boldsymbol{x}^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)$ and the set of all solutions parametrized by the barrier parameter τ describes the *central path* that approaches the optimal solution $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ of the original problem (4.44) for a sequence $\{\tau^k\}$ with $\tau^{k+1} < \tau^k$ and $\lim_{k \to \infty} \tau^k = 0$, see Figure 4.9. Interior point methods have the same behavior and therefore approach the optimal solution along the central path as the barrier parameter τ^k is successively reduced.

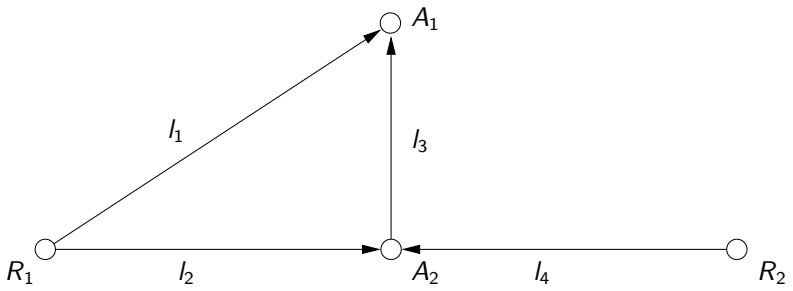


Figure 4.10: Transport network of the refinery example.

4.3.5 Example: Transport network

Figure 4.10 shows an abstract transport network of two refineries R_1 und R_2. The produced fuel is transported to two consumption areas A_1, A_2, (e.g. industry or towns) via four pipelines $l_1, \ldots, l_4$.[5] We assume that the consumption areas A_1, A_2 have the constant consumption rate R_1, R_2 and that the transport costs for each pipeline is proportional to its flow rate x_i with the proportionality factor K_i.

The goal is to find the optimal distribution of flows x_i for the minimization of the transport costs while maintaining the consumption rates R_1, R_2 and accounting for the maximum flow capacity C_i of each pipeline. Moreover, the refinery R_1 is supposed to have a maximum production rate P_1.

The following (dimension-free) parameter values are considered:

$$\begin{aligned} K_1 &= 40, & C_1 &= 30, & R_1 &= 50, & P_1 &= 45 \\ K_2 &= 20, & C_2 &= 20, & R_2 &= 40 \\ K_3 &= 10, & C_3 &= 30 \\ K_4 &= 10, & C_4 &= 60\,. \end{aligned}$$

Exercise 4.7 *Formulate the optimization problem of the transport network. Reduce the problem to two dimensions by substituting the equality constraints.*

Exercise 4.8 *Draw the admissible set as well as the contour lines of the cost function and determine the optimal solution graphically.*

The Optimization Toolbox of MATLAB provides the function

linprog(f, A, b, Aeq, beq, LB, UB, X0, options)

for solving linear optimization problems of the form

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \boldsymbol{f}^\mathsf{T}\boldsymbol{x} \quad \text{s.t.} \quad \boldsymbol{A}_{eq}\boldsymbol{x} = \boldsymbol{b}_{eq}\,, \quad \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\,, \quad \boldsymbol{LB} \leq \boldsymbol{x} \leq \boldsymbol{UB}\,.$$

The simplex method and the interior point method can be selected with the following options:

Simplex method:	opt = optimoptions('linprog','Algorithm','dual-simplex')
Interior point method:	opt = optimoptions('linprog','Algorithm','interior-point')

[5] This is a modified version of the refinery example in [15].

Note that MATLAB employs a simplex method for the *dual problem* of (4.44), see Section 4.6 and Example 4.8, which is the reason for the setting 'dual-simplex'.

Exercise 4.9 *Solve the transport network problem with* MATLAB *(see Figure 4.11) and compare the simplex method with the interior point method in terms of accuracy and number of iterations.*

```
function transport_problem(methodQ)
% ------------------------------------------------------
% methodQ: 1 - Simplex method
%          2 - Interior point method

opt = optimoptions('linprog','Display','iter'); % options for output

switch methodQ
 case 1,                                        % simplex method
  opt = optimoptions(opt,'Algorithm','dual-simplex');
 case 2,                                        % interior point method
  opt = optimoptions(opt,'Algorithm','interior-point');
end

K = [40; 20; 10; 10];                           % parameters for
C = [30; 20; 30; 60];                           % transport problem
R = [50; 40];

xopt = [20;10;30;60];                           % optimal point

c   = K;                                        % cost vector
Aeq = [1,0,1,0; 0,1,-1,1]; beq = R;             % equality constraints
LB  = [0;0;0;0];           UB  = C;             % constraints on x
A   = [1,1,0,0];           b   = 45;            % general constraints Ax <= b

% call of linprog
% -------------------
[x,f] = linprog(c,A,b,Aeq,beq,LB,UB,[],opt);
fprintf('\nComputed solution:   [%g, %g, %g, %g] with f(x) = %g',x,f);
fprintf('\nDistance to optimum: [%2.2g, %2.2g, %2.2g, %2.2g]\n',x-xopt);
```

Figure 4.11: MATLAB code for the transport network problem (linprog).

4.4 Quadratic optimization

Another special case is quadratic optimization – also known as *quadratic programming* – with quadratic cost function and linear constraints

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & \tfrac{1}{2}\,\boldsymbol{x}^\mathsf{T}\boldsymbol{G}\boldsymbol{x} + \boldsymbol{c}^\mathsf{T}\boldsymbol{x}\,, && \boldsymbol{G}\in\mathbb{R}^{n\times n} && \text{(4.72a)}\\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\,, && \boldsymbol{A}\in\mathbb{R}^{p\times n}, \quad \boldsymbol{b}\in\mathbb{R}^p && \text{(4.72b)}\\ & \boldsymbol{C}\boldsymbol{x} \leq \boldsymbol{d}\,, && \boldsymbol{C}\in\mathbb{R}^{q\times n}, \quad \boldsymbol{d}\in\mathbb{R}^q\,. && \text{(4.72c)} \end{aligned}$$

The weighting matrix $\boldsymbol{G}\in\mathbb{R}^{n\times n}$ is supposed to be symmetric. If it is positive (semi)definite, then the quadratic problem is (strictly) convex.

Similar to linear optimization (Section 4.3), this problem class is of special importance. An example is portfolio optimization, where an optimal compromise is sought between profit maximization and risk minimization (covariance). Moreover, many numerical methods for solving general nonlinear optimization problems (Section 4.5) are based on the sequential solution of quadratic problems.

4.4.1 Optimality conditions

The general nonlinear KKT conditions (4.29) have a simplified structure for quadratic problems. With the Lagrangian function

$$L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \tfrac{1}{2}\, \boldsymbol{x}^\mathsf{T} \boldsymbol{G} \boldsymbol{x} + \boldsymbol{c}^\mathsf{T} \boldsymbol{x} + \boldsymbol{\lambda}^\mathsf{T} (\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) + \boldsymbol{\mu}^\mathsf{T} (\boldsymbol{C}\boldsymbol{x} - \boldsymbol{d})$$

and the Lagrange multipliers $\boldsymbol{\lambda} \in \mathbb{R}^p$, $\boldsymbol{\mu} \in \mathbb{R}^q$, the KKT conditions for an optimal solution read

$$\boldsymbol{G}\boldsymbol{x}^* + \boldsymbol{c} + \boldsymbol{A}^\mathsf{T}\boldsymbol{\lambda}^* + \boldsymbol{C}^\mathsf{T}\boldsymbol{\mu}^* = \boldsymbol{0} \tag{4.73a}$$

$$\boldsymbol{A}\boldsymbol{x}^* - \boldsymbol{b} = \boldsymbol{0} \tag{4.73b}$$

$$\boldsymbol{C}\boldsymbol{x}^* - \boldsymbol{d} \leq \boldsymbol{0} \tag{4.73c}$$

$$\mu_i^* \geq 0\,, \quad i = 1, \ldots, n \tag{4.73d}$$

$$\mu_i^* (\boldsymbol{C}_i^\mathsf{T}\boldsymbol{x}^* - d_i) = 0\,, \quad i = 1, \ldots, n \tag{4.73e}$$

with $\boldsymbol{C}_i^\mathsf{T}$ denoting the i-th *row* of the constraint matrix $\boldsymbol{C}$. In the convex case, i.e. if $\boldsymbol{G}$ is positive semidefinite, a solution $\boldsymbol{x}^*$ (if it exists) is always a global solution of the quadratic problem and the KKT conditions (4.29) are necessary and sufficient for the existence of a global minimum point $\boldsymbol{x}^*$. Non-convex quadratic problems are usually more difficult to solve since multiple stationary points and minima may exist.

4.4.2 Equality-constrained quadratic problems

An important subclass are equality-constrained quadratic problems

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \quad \tfrac{1}{2}\, \boldsymbol{x}^\mathsf{T} \boldsymbol{G} \boldsymbol{x} + \boldsymbol{c}^\mathsf{T} \boldsymbol{x} \tag{4.74a}$$

$$\text{s.t.} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\,, \qquad \boldsymbol{A} \in \mathbb{R}^{p \times n}, \quad \boldsymbol{b} \in \mathbb{R}^p\,, \tag{4.74b}$$

for which the KKT conditions (4.73) reduce to the linear set of equations

$$\begin{bmatrix} \boldsymbol{G} & \boldsymbol{A}^\mathsf{T} \\ \boldsymbol{A} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}^* \\ \boldsymbol{\lambda}^* \end{bmatrix} = \begin{bmatrix} -\boldsymbol{c} \\ \boldsymbol{b} \end{bmatrix} \tag{4.75}$$

that consists of $n + p$ equations for the $n + p$ unknowns $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$. The block matrix in (4.75) is known as *Karush-Kuhn-Tucker matrix* or simply *KKT matrix*.

There exist different methods for solving (4.75) that are shortly summarized here. More informationen can be found, e.g. in [14].

- *Direct methods* solve the set of equations (4.75) by means of factorization, e.g. the *null-space method*. Direct methods are often used for small and mid-sized problems, but there are also efficient factorization methods for sparse large-scale problems.
- *Iterative methods* are well suited for large-scale problems and also allow for parallelization. A popular class of iterative methods are *conjugate gradient methods*.

Equality-constrained quadratic problems are important in optimization due to the linearity of the reduced KKT conditions (4.75) and their efficient solution by direct or iterative methods. Moreover, inequality-constrained problems as well as the solution of nonlinear constrained problems by means of *sequential quadratic programming* (see Section 4.5.1) break down to solving equality-constrained subproblems of the form (4.74).

4.4.3 Active set method

A popular method for solving general quadratic problems (4.72) is the *active set method*. Similar to the simplex method in linar optimization, the active set method identifies active constraints in each iteration and solves an equality-constrained problem of the form (4.74).

The basic idea of the active set method is to determine a number of active constraints from $\boldsymbol{C}\boldsymbol{x} \leq \boldsymbol{d}$ in each iteration k. The corresponding indices are captured in the *working set* $\mathcal{A}^k$, i.e.

$$\boldsymbol{C}_i^\mathsf{T}\boldsymbol{x}^k = d_i\,, \quad i \in \mathcal{A}^k\,.$$

The working set $\mathcal{A}^k$ does not necessarily consider all constraints that are active at point $\boldsymbol{x}^k$, but it is iteratively completed over the iterations until the active constraint set (see Definition 4.1) is reached at the optimal point $\boldsymbol{x}^*$

$$\mathcal{A}(\boldsymbol{x}^*) = \{i \in \{1,\dots,q\} \,|\, \boldsymbol{C}_i^\mathsf{T}\boldsymbol{x}^* = d_i\}\,.$$

The active set method typically assumes that the constraint qualification (4.26) is satified at each iteration point $\boldsymbol{x}^k$, i.e.

$$\operatorname{rank}\begin{bmatrix}\boldsymbol{A}\\ [\boldsymbol{C}_i^\mathsf{T}]_{i\in\mathcal{A}^k}\end{bmatrix} = p + |\mathcal{A}^k| \quad \forall k \tag{4.76}$$

and that the matrix $\boldsymbol{G}$ is positive definite.

In each iteration k, the active set method verifies if the point $\boldsymbol{x}^k$ with the working set $\mathcal{A}^k$ is the optimal solution of the quadratic problem. If this is not the case, a direction $\boldsymbol{s}$ is computed by solving an equality-constrained problem that accounts for all active constraints according to the working set $\mathcal{A}^k$. This problem is derived from the original one (4.72) by replacing $\boldsymbol{x} = \boldsymbol{x}^k + \boldsymbol{s}$ and using the current working set [6]

$$\min_{\boldsymbol{s}\in\mathbb{R}^n} \quad \tfrac{1}{2}\,\boldsymbol{s}^\mathsf{T}\boldsymbol{G}\boldsymbol{s} + (\boldsymbol{G}\boldsymbol{x}^k + \boldsymbol{c})^\mathsf{T}\boldsymbol{s} \tag{4.77a}$$

$$\text{s.t.} \quad \boldsymbol{A}\boldsymbol{s} = \boldsymbol{0}\,, \quad \boldsymbol{C}_i^\mathsf{T}\boldsymbol{s} = 0\,, \quad i \in \mathcal{A}^k\,. \tag{4.77b}$$

Since we assume that $\boldsymbol{G}$ is positive definite, the problem (4.77) can be solved with the numerical methods mentioned in Section 4.4.2.

If the solution of (4.77) satisfies $\boldsymbol{s}^k \neq \boldsymbol{0}$ and $\boldsymbol{x}^k + \boldsymbol{s}^k$ is feasible in terms of the constraints $\boldsymbol{C}(\boldsymbol{x}^k + \boldsymbol{s}^k) \leq \boldsymbol{d}$, then $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \boldsymbol{s}^k$ is defined as new iteration point. If $\boldsymbol{x}^k + \boldsymbol{s}^k$ is not feasible, the step size $\alpha^k < 1$ for

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha^k\boldsymbol{s}^k \tag{4.78}$$

must be reduced until all constraints are satisfied. An important property in this regard is that each active constraint $i \in \mathcal{A}^k$ remains active in direction $\boldsymbol{s}^k$, because

$$\boldsymbol{C}_i^\mathsf{T}(\boldsymbol{x}^k + \alpha\boldsymbol{s}^k) = \boldsymbol{C}_i^\mathsf{T}\boldsymbol{x}^k = d_i\,, \quad \forall i \in \mathcal{A}^k \quad \forall \alpha \geq 0\,. \tag{4.79}$$

[6] When substituting $\boldsymbol{x} = \boldsymbol{x}^k + \boldsymbol{s}$ in the cost function (4.72a), an additional term $\frac{1}{2}(\boldsymbol{x}^k)^\mathsf{T}\boldsymbol{G}\boldsymbol{x}^k + \boldsymbol{c}^\mathsf{T}\boldsymbol{x}^k$ appears that is independent of $\boldsymbol{s}$ and therefore omitted in the new cost function (4.77a).

It is therefore sufficient to consider those constraints that are not yet included in $\mathcal{A}^k$, especially those with $\boldsymbol{C}_i^\mathsf{T}\boldsymbol{s}^k > 0$. In this case, the original constraint $\boldsymbol{C}_i^\mathsf{T}(\boldsymbol{x}_k + \alpha^k\boldsymbol{s}^k) \leq d_i$ is only satisfied if

$$\alpha^k \leq \frac{d_i - \boldsymbol{C}_i^\mathsf{T}\boldsymbol{x}^k}{\boldsymbol{C}_i^\mathsf{T}\boldsymbol{s}^k}\,. \tag{4.80}$$

This shows that the largest possible step size α^k is given by

$$\alpha^k = \min\left\{1,\ \min_{i\notin\mathcal{A}^k,\boldsymbol{C}_i^\mathsf{T}\boldsymbol{s}^k>0} \frac{d_i - \boldsymbol{C}_i^\mathsf{T}\boldsymbol{x}^k}{\boldsymbol{C}_i^\mathsf{T}\boldsymbol{s}^k}\right\}\,. \tag{4.81}$$

If $\alpha^k < 1$ follows from (4.81), a new constraint becomes active and its index is added to the new working set $\mathcal{A}^{k+1}$.

This procedure is iterated until $\boldsymbol{s}^k = \boldsymbol{0}$, which implies that point $\boldsymbol{x}^k$ minimizes the quadratic cost function (4.77a). To verify the KKT conditions (4.73) at this point $\boldsymbol{x}^k$, the Lagrange multipliers associated with the active constraints must be non-negative, i.e. $\mu_i^k \geq 0$, $i \in \mathcal{A}^k$. If this is the case, then $\boldsymbol{x}^k$ is the optimal and indeed global solution $\boldsymbol{x}^*$ due to the positive definiteness of $\boldsymbol{G}$. Otherwise, if at least one multiplier μ_i^k, $i \in \mathcal{A}^k$ is negative, the cost function (4.72a) can be further reduced if the corresponding constraint is removed from the working set, see Section 4.1.2 and 4.1.3. The reduction of $\mathcal{A}^k$ by one element $i \in \mathcal{A}^k$ with $\mu_i^k < 0$ and the new solution of the problem (4.77) then generates a new feasible direction $\boldsymbol{s}^{k+1}$.

The algorithm of the active set method is summarized in Table 4.2. For stricty convex quadratic problems, it can be shown that the active set method converges in a finite number of iterations.

Initialization: $\boldsymbol{x}^0$ & $\mathcal{A}^0$	Initial point & working set
for $k = 0, 1, 2, \ldots$ **do**	
$\boldsymbol{s}^k$ & μ_i^k, $i \in \mathcal{A}^k \leftarrow$ (4.77) & (4.73)	Equality-constrained problem (4.77)
if $\boldsymbol{s}^k = \boldsymbol{0}$ **then**	Minimum found ($\boldsymbol{s}^k = \boldsymbol{0}$)?
if $\mu_i^k \geq 0\ \forall i \in \mathcal{A}^k$ **then**	
return $\boldsymbol{x}^* = \boldsymbol{x}^k$	Optimal solution
else	
$i^k \leftarrow \arg\min_{\mathcal{A}^k} \mu_i^k$	
$\mathcal{A}^{k+1} \leftarrow \mathcal{A}^k \backslash \{i^k\}$, $\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x}^k$	Remove active constraint i^k
end	
else	Minimum not found ($\boldsymbol{s}^k \neq \boldsymbol{0}$)
α^k & $i^k \leftarrow$ (4.81)	Step size computation
if $\alpha^k < 1$ **then**	
$\mathcal{A}^{k+1} \leftarrow \mathcal{A}^k \cup \{i^k\}$	Add new active constraint i^k
else	
$\mathcal{A}^{k+1} \leftarrow \mathcal{A}^k$	
end	
$\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x}^k + \alpha^k\boldsymbol{s}^k$	Next iteration point
end	
end	

Table 4.2: Active set method.

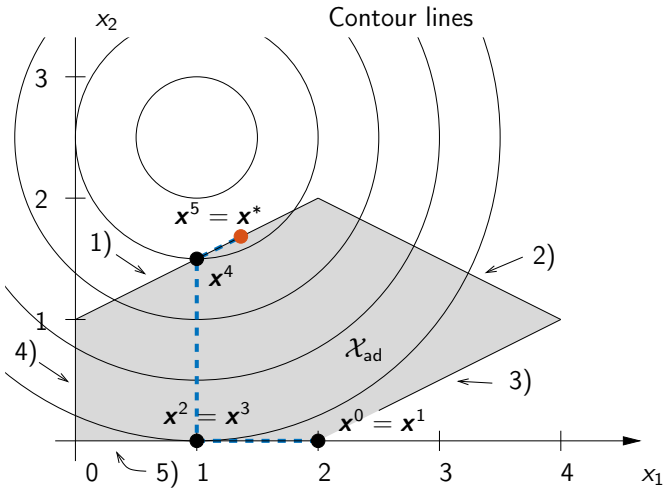


Figure 4.12: Iterations of the active set method in Example 4.6.

Example 4.6 ([14]) *We consider the quadratic problem*

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^2} \quad & (x_1-1)^2+(x_2-2.5)^2 \\ \text{s.t.} \quad 1) \quad & -x_1+2x_2-2\le 0 \\ 2) \quad & x_1+2x_2-6\le 0 \\ 3) \quad & x_1-2x_2-2\le 0 \\ 4) \quad & x_1\ge 0 \\ 5) \quad & x_2\ge 0\,. \end{aligned}$$

The admissible set $\mathcal{X}_{ad}$ and the contour lines of the cost function are plotted in Figure 4.12. The constraints are numbered with the indices 1 bis 5.

***Start**: The initial point for the active set method is chosen as $\boldsymbol{x}^0 = [2,0]^\mathsf{T}$. Since the constraints 3 and 5 are active at $\boldsymbol{x}^0$, the working set is initialized with $\mathcal{A}^0 = \{3,5\}$. The solution of problem (4.77) is $\boldsymbol{s}^0 = \boldsymbol{0}$ and the KKT condition (4.73a) with $\mu_i^0 = 0$, $i = 1,2,4$ leads to the multipliers $(\mu_3^0, \mu_5^0) = (-2,-1)$ of the active constraints 3 and 5.*

***Iteration 1:** Constraint 3 is removed from the working set $\mathcal{A}^0$ because of $\mu_3^0 < \mu_5^0 < 0$, i.e. $\mathcal{A}^1 = \{5\}$, and the next solution of (4.77) is $\boldsymbol{s}^1 = [-1,0]^\mathsf{T}$. The step size $\alpha^k = 1$ follows from (4.81), leading to a "full" step $\boldsymbol{x}^2 = \boldsymbol{x}^1 + \boldsymbol{s}^1 = [1,0]^\mathsf{T}$ without a further constraint becoming active. The working set therefore remains unchanged with $\mathcal{A}^2 = \mathcal{A}^1$, see Figure 4.12.*

***Iteration 2:** The solution of (4.77) now gives $\boldsymbol{s}^2 = \boldsymbol{0}$. The evaluation of the KKT conditions (4.73a) yields $\mu_5^2 = -5$ for the remaining active constraint in $\mathcal{A}^2 = \{5\}$. After removing the constraint, we get the empty working set $\mathcal{A}^3 = \{\}$.*

***Iteration 3:** The problem (4.77) is now unconstrained and leads to $\boldsymbol{s}^3 = [0,2.5]^\mathsf{T}$ with the step size $\alpha^3 = 0.6$ and the new iteration point $\boldsymbol{x}^4 = [1,1.5]^\mathsf{T}$. The constraint 1 blocks a full step of the algorithm and becomes active. The working set for the next iteration is therefore updated to $\mathcal{A}^4 = \{1\}$.*

***Iteration 4:** We obtain $\boldsymbol{s}^4 = [0.4,0.2]^\mathsf{T}$ with $\alpha^4 = 1$ and $\boldsymbol{x}^5 = [1.4,1.7]^\mathsf{T}$, see Figure 4.12. The working set remains unchanged $\mathcal{A}^5 = \mathcal{A}^4 = \{1\}$ since no new constraint becomes active.*

***Iteration 5:** In the last iteration, (4.77) gives $\boldsymbol{s}^5 = \boldsymbol{0}$ with the Lagrange multiplier $\mu_1^5 = 0.8 > 0$ from (4.73a). Thus, $\boldsymbol{x}^5 = \boldsymbol{x}^*$ is the optimal solution and the algorithm terminates.*

4.4.4 Further numerical methods

The active set method is well suited for low-dimensional or mid-sized problems but can show slow convergence properties for large-scale problems due to the fact that the working set $\mathcal{A}^k$ is only updated by one element in each iteration. An alternative for large-scale problems are *projected gradient methods* as they allow for a rapid adaptation of the working set $\mathcal{A}^k$.

A further class of methods for solving quadratic problems are interior point methods that were already introduced in terms of linear programming, see Section 4.3.4. The inequality constraints (4.72c) can be transformed into the standard form (4.44b)-(4.44c) by means of *slack variables*.

More information on projected gradient methods and interior points methods for quadratic problems can be found in the textbooks at the end of the chapter.

Exercise 4.10 *Verify the solution of the quadratic problem in Example 4.6 numerically by using the* MATLAB *function quadprog of the Optimization Toolbox, also see Figure 4.13.*

```
% Matlab solution of a quadratic example problem
% ----------------------------------------------
opt = optimoptions('quadprog','Display','iter','Algorithm','interior-point-convex');

G = [2,0; 0,2];                                          % cost function: 0.5*x'*G*x + c'*x
c = [-2;-5];
C = [-1,2; 1,2; 1,-2; -1,0; 0,-1];                       % constraints: C*x <= d
d = [   2;   6;    2;    0;    0];

x0   = [2.0;0.0];                                        % initial guess

[x,f] = quadprog(G,c,C,d,[],[],[],[],[],opt);            % solution with quadprog
fprintf('\nComputed solution: [%g, %g] mit f(x) = %g\n',x,f);
```

Figure 4.13: MATLAB code for Exercise 4.10 (quadprog).

4.5 Nonlinear optimization

After the special cases of linear and quadratic optimization, we focus on the solution of general nonlinear problems (4.1) that are repeated here for the sake of completeness:

$$\boxed{\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^n} \quad & f(\boldsymbol{x}) && && \text{(4.82a)} \\ \text{s.t.} \quad & g_i(\boldsymbol{x}) = 0, && i = 1,\dots,p && \text{(4.82b)} \\ & h_i(\boldsymbol{x}) \leq 0, && i = 1,\dots,q\,. && \text{(4.82c)} \end{aligned}}$$

4.5.1 Sequential quadratic programming (SQP)

An efficient class of algorithms for nonlinear problems is *sequential quadratic programming (SQP)* that solves a sequence of linear-quadratic approximations of the constrained problem (4.82). SQP methods therefore rely on methods from quadratic programming that are iteratively applied to the nonlinear problem.

General idea and algorithm

To illustrate the working principle of SQP, we consider an equality-constrained problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \quad f(\boldsymbol{x}) \tag{4.83a}$$

$$\text{s.t.} \quad \boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0} \tag{4.83b}$$

with $\boldsymbol{g} = [g_1, \dots, g_p]^\mathsf{T}$. The functions $f(\boldsymbol{x})$ and $\boldsymbol{g}(\boldsymbol{x})$ are supposed to be continuously differentiable. The idea of SQP methods is to approximate the nonlinear problem (4.83) by a quadratic problem at the current iteration point $\boldsymbol{x}^k$ and to use its solution to compute the next iterate $\boldsymbol{x}^{k+1}$.

The quadratic problem to be solved can be derived by looking at the optimality conditions for (4.83). With the Lagrangian function $L(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^\mathsf{T}\boldsymbol{g}(\boldsymbol{x})$, the KKT conditions (4.29) simplify to the equations

$$\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{\lambda}) = \begin{bmatrix} \nabla f(\boldsymbol{x}) + \boldsymbol{A}(\boldsymbol{x})^\mathsf{T}\boldsymbol{\lambda} \\ \boldsymbol{g}(\boldsymbol{x}) \end{bmatrix} = \boldsymbol{0} \quad \text{with} \quad \boldsymbol{A}(\boldsymbol{x}) = [\nabla g_1(\boldsymbol{x}), \dots, \nabla g_p(\boldsymbol{x})]^\mathsf{T} \tag{4.84}$$

of order $n + p$ for the $n + p$ unknowns $(\boldsymbol{x}, \boldsymbol{\lambda})$. Newton's method for solving the equations is based on the iteration scheme

$$\begin{bmatrix} \boldsymbol{x}^{k+1} \\ \boldsymbol{\lambda}^{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}^k \\ \boldsymbol{\lambda}^k \end{bmatrix} + \boldsymbol{s}^k, \quad \boldsymbol{s}^k = \begin{bmatrix} \Delta\boldsymbol{x}^k \\ \Delta\boldsymbol{\lambda}^k \end{bmatrix} \tag{4.85}$$

with the Newton direction $\boldsymbol{s}^k$ as the solution of

$$\underbrace{\begin{bmatrix} \nabla^2_{\boldsymbol{xx}} L(\boldsymbol{x}^k, \boldsymbol{\lambda}^k) & \boldsymbol{A}(\boldsymbol{x}^k)^\mathsf{T} \\ \boldsymbol{A}(\boldsymbol{x}^k) & \boldsymbol{0} \end{bmatrix}}_{\nabla \boldsymbol{F}(\boldsymbol{x}^k, \boldsymbol{\lambda}^k)} \begin{bmatrix} \Delta\boldsymbol{x}^k \\ \Delta\boldsymbol{\lambda}^k \end{bmatrix} = \underbrace{\begin{bmatrix} -\nabla f(\boldsymbol{x}^k) - \boldsymbol{A}(\boldsymbol{x}^k)^\mathsf{T}\boldsymbol{\lambda}^k \\ -\boldsymbol{g}(\boldsymbol{x}^k) \end{bmatrix}}_{-\boldsymbol{F}(\boldsymbol{x}^k, \boldsymbol{\lambda}^k)}. \tag{4.86}$$

The matrix $\nabla \boldsymbol{F}(\boldsymbol{x}^k, \boldsymbol{\lambda}^k)$ represents the Jacobian of $\boldsymbol{F}(\boldsymbol{x}^k, \boldsymbol{\lambda}^k)$ w.r.t. $(\boldsymbol{x}, \boldsymbol{\lambda})$.

An alternative view on the Newton iteration (4.85), (4.86) is obtained if we consider the quadratic problem

$$\min_{\boldsymbol{s}\in\mathbb{R}^n} \quad f(\boldsymbol{x}^k) + \nabla f(\boldsymbol{x}^k)^\mathsf{T}\boldsymbol{s} + \tfrac{1}{2}\boldsymbol{s}^\mathsf{T}\nabla^2_{\boldsymbol{xx}} L(\boldsymbol{x}^k, \boldsymbol{\lambda}^k)\boldsymbol{s} \tag{4.87a}$$

$$\text{s.t.} \quad \boldsymbol{A}(\boldsymbol{x}^k)\boldsymbol{s} + \boldsymbol{g}(\boldsymbol{x}^k) = \boldsymbol{0} \tag{4.87b}$$

with the optimality conditions

$$\nabla^2_{\boldsymbol{xx}} L(\boldsymbol{x}^k, \boldsymbol{\lambda}^k)\boldsymbol{s}^k + \nabla f(\boldsymbol{x}^k) + \boldsymbol{A}(\boldsymbol{x}^k)^\mathsf{T}\boldsymbol{\lambda}^k_{\boldsymbol{s}} = \boldsymbol{0} \tag{4.88a}$$

$$\boldsymbol{A}(\boldsymbol{x}^k)\boldsymbol{s} + \boldsymbol{g}(\boldsymbol{x}^k) = \boldsymbol{0} \tag{4.88b}$$

for the solution $(\boldsymbol{s}^k, \boldsymbol{\lambda}^k_{\boldsymbol{s}})$. Adding the term $\boldsymbol{A}(\boldsymbol{x}^k)^\mathsf{T}\boldsymbol{\lambda}^k$ on both sides of the first equation in (4.86) and with $\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \Delta\boldsymbol{\lambda}^k$ we obtain

$$\begin{bmatrix} \nabla^2_{\boldsymbol{xx}} L(\boldsymbol{x}^k, \boldsymbol{\lambda}^k) & \boldsymbol{A}(\boldsymbol{x}^k)^\mathsf{T} \\ \boldsymbol{A}(\boldsymbol{x}^k) & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \Delta\boldsymbol{x}^k \\ \boldsymbol{\lambda}^{k+1} \end{bmatrix} = \begin{bmatrix} -\nabla f(\boldsymbol{x}^k) \\ -\boldsymbol{g}(\boldsymbol{x}^k) \end{bmatrix}. \tag{4.89}$$

Initialization:	
$k \leftarrow 0$	Iteration counter
$\boldsymbol{x}^0$, $\boldsymbol{H}^0$	Initial point and Hessian approx. (pos. def.)
ε	Convergence tolerance
repeat	
$\boldsymbol{s}^k \leftarrow$ Solution of (4.90)	
$\alpha^k \leftarrow$ e.g. Armijo condition (3.21)	Step size α^k for descent in penalty function $P(\boldsymbol{x}^k + \alpha^k \boldsymbol{s}^k, \sigma) < P(\boldsymbol{x}^k, \sigma)$, e.g. with (4.91)
$\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x}^k + \alpha^k \boldsymbol{s}^k$	Next iteration point
$k \leftarrow k + 1$	
until $\|\|\nabla_{\boldsymbol{x}} L(\boldsymbol{x}^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)\|\| \leq \varepsilon$	Or other tolerance criterion

Table 4.3: Structure of SQP methods.

It is straightforward to see that (4.89) and the optimality conditions (4.88) for the quadratic problem (4.87) are equivalent for $\Delta \boldsymbol{x}^k = \boldsymbol{s}^k$ and $\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}_{\boldsymbol{s}}^k$. The iteration variables $(\boldsymbol{x}^{k+1}, \boldsymbol{\lambda}^{k+1})$ therefore can be computed via the Newton iteration (4.85), (4.86) or as the solution of the quadratic problem (4.87) with $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \boldsymbol{s}^k$ and the multipliers $\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}_{\boldsymbol{s}}^k$.

This interpretation can be extended to inequality-constrained problems as the equality constraints (4.87b) of the quadratic problem represent a linearization of $\boldsymbol{g}(\boldsymbol{x})$ at point $\boldsymbol{x}^k$. In the case of the general nonlinear problem (4.82), SQP methods consider the following quadratic problem in each iteration k

$$\min_{\boldsymbol{s} \in \mathbb{R}^n} \quad \tfrac{1}{2} \boldsymbol{s}^\mathsf{T} \nabla^2_{\boldsymbol{xx}} L(\boldsymbol{x}^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k) \boldsymbol{s} + \nabla f(\boldsymbol{x}^k)^\mathsf{T} \boldsymbol{s} \tag{4.90a}$$

$$\text{s.t.} \quad \nabla g_i(\boldsymbol{x}^k)^\mathsf{T} \boldsymbol{s} + g_i(\boldsymbol{x}^k) = 0, \quad i = 1, \ldots, p \tag{4.90b}$$

$$\nabla h_i(\boldsymbol{x}^k)^\mathsf{T} \boldsymbol{s} + h_i(\boldsymbol{x}^k) \leq 0, \quad i = 1, \ldots, q\,, \tag{4.90c}$$

whereby the term $f(\boldsymbol{x}^k)$ is independent of $\boldsymbol{s}$ and therefore omitted in the cost function. The equations (4.90) are again solved with the methods described in Section 4.4. The Hessian matrix $\nabla^2_{\boldsymbol{xx}} L(\boldsymbol{x}^k, \boldsymbol{\lambda}^k, \boldsymbol{\mu}^k)$ of the Lagrangian function can be determined directly (i.e. either analytically, numerically, or by using automatic differentiation) or by a quasi-Newton approximation, see e.g. [14].

The next iteration point is $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha^k \boldsymbol{s}^k$ with the solution $\boldsymbol{s}^k$ of (4.90). The step size α^k must be chosen as a compromise between a sufficient decrease of the cost function $f(\boldsymbol{x}^{k+1}) < f(\boldsymbol{x}^k)$ while complying with the inequality constraints (4.82c) within a certain tolerance. This point will be touched upon in Section 4.5.1. The general algorithm of an SQP method is listed in Table 4.3.

Numerical methods for solving (4.90) can be classified as follows :

- *EQP methods* (from "equality-constrained QP") choose a working set of active constraints at each iteration point $\boldsymbol{x}^k$ and reduce (4.90) to an equality-constrained problem, where all constraints that are not part of the working set are discarded. The working set is updated in each iteration. This method has the advantage that the solution of high-dimensional equality-constrained quadratic problems requires less computational effort than the solution of the general problem (4.90).

- *IQP methods* (from "inequality-constrained QP") use the active constraint set of (4.90) as the estimate of $\mathcal{A}(\boldsymbol{x}^*)$ at the optimal point $\boldsymbol{x}^*$. If the set $\mathcal{A}(\boldsymbol{x}^*)$ is determined correctly, the SQP method behaves similar to Newton's method for equality-constrained problems and converges rapidly to the optimal point $\boldsymbol{x}^*$. This approach has proven successful in practice. The disadvantage is that the solution of (4.90) is computationally demanding for large-scale problems, although information from previous iterations can be used as warm-start strategy in the neighborhood of the optimum in order to start with a good estimate of the solution in the current iteration.

Step size computation

It is not sufficient to choose the step size α^k in view of the maximum descent of the cost function $f(\boldsymbol{x}^k+\alpha^k\boldsymbol{s}^k) < f(\boldsymbol{x}^k)$ because the compliance with the nonlinear constraints $g_i(\boldsymbol{x}) = 0$ and $h_i(\boldsymbol{x}) \leq 0$ must be taken into account as well. Both aspects can be combined in a *penalty function*. In particular, the l_1*-penalty function*

$$P(\boldsymbol{x},\sigma) = f(\boldsymbol{x}) + \sigma\left[\sum_{i=1}^{p}|g_i(\boldsymbol{x})| + \sum_{i=1}^{q}\max\{0, h_i(\boldsymbol{x})\}\right] \tag{4.91}$$

has the characteristics that each solution $\boldsymbol{x}^*$ of the nonlinear problem (4.82) is also a minimizing point of P under the assumption that $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ satisfies the sufficient optimality conditions (4.37) and that the penalty parameter σ is larger than the absolute value of the multipliers, i.e. $\sigma > |\lambda_i^*|$, $i = 1,\ldots,p$ and $\sigma > |\mu_i^*|$, $i = 1,\ldots,q$. In this case, the function (4.91) is also called *exact penalty function*. The step size α^k can be determined with one of the line search methods from Section 3.3.1, e.g. the Armijo condition (3.21) to satisfy $P(\boldsymbol{x}^k + \alpha^k\boldsymbol{s}^k, \sigma) < P(\boldsymbol{x}^k, \sigma)$.

Example 4.7 (Optimized controller parameters) *We consider the optimization of controller parameters for a time-discrete system with delay*

$$x(k) = ax(k-1) + bu(k-10) \tag{4.92}$$

and the system parameters $a = b = 0.5$ *[15]. The control law is given by*

$$u(k) = c_1u(k-1) + c_2e(k) + c_3e(k-1)\,, \tag{4.93}$$

where $e(k) = 1 - x(k)$ *denotes the output error for a unit step as reference signal. The controller parameters* (c_1, c_2, c_3) *are to be determined such that the discrete-time* control surface

$$f(\boldsymbol{c}) = \sum_{k=1}^{N}|e(k)| \tag{4.94}$$

or the time-weighted control surface

$$f(\boldsymbol{c}) = \sum_{k=1}^{N}\frac{k}{N}|e(k)| \tag{4.95}$$

are minimized. The latter case puts more weight on control errors over the time horizon. In addition, overshoot in the step response shall be bounded in form of an inequality constraint.

The Optimization Toolbox of MATLAB *offers the function fmincon for the solution of nonlinear optimization problems. The SQP method is chosen with the option optimset('Algorithm','active-set') or optimset('Algorithm','sqp').*

The MATLAB *code to compute the optimal controller parameters with fmincon is shown in Figure 4.14.*[7] *The limitation of the overshoot is implemented as nonlinear constraint. The initial guess for the controller parameters is* $\boldsymbol{c}^0 = [1, 0.5, -0.5]^\mathsf{T}$.

The optimized step response for the cases (4.94) and (4.95) with 10% and 0% overshoot are shown in Figure 4.15 and 4.16. The corresponding controller parameters are listed in Table 4.4. Note that the optimization returns $c_1 = 1$ *in all cases, which shows that the controller (4.93) has integral action. Figure 4.16 also shows that the time-dependent weighting in the cost function (4.95) leads to a reduction of the oscillations after the setpoint is reached.*

```
function controllerparameters(caseQ)
% ------------------------------------------------------
global x cold
switch caseQ
 case 1,   p.costQ=1; p.ov=0.1;    % control surface,               max. overshoot 10%
 case 2,   p.costQ=1; p.ov=0;      % control surface,               max. overshoot  0%
 case 3,   p.costQ=2; p.ov=0.1;    % time-weighted control surface, max. overshoot 10%
 case 4,   p.costQ=2; p.ov=0;      % time-weighted control surface, max. overshoot  0%
end

p.N = 70;                                                         % number of steps
p.a = 0.5; p.b = 0.5;                                             % system parameters
c0  = [1;0.5;-0.5];                                               % initial guess

opt = optimoptions('fmincon','Algorithm','sqp', ...               % SQP method for fmincon
                   'Display','iter','MaxFunEvals',500);

[c,f] = fmincon(@costfct,c0,[],[],[],[],[],[],@nlconstr,opt,p);
fprintf('\nOptimal parameters c=[%2.3g,%2.3g,%2.3g] with f(c)=%2.3g\n',c,f);

x = stepresponse(c,p);                                            % optimized step response

figure(1); clf                                                    % plotting
h(1) = stairs(ones(1,p.N),'r--'); hold on
h(2) = stairs(x,'k');
h(3) = stairs(1-x,'m'); xlabel('Time steps');
axis([0, p.N, -0.1, 1.1+p.ov]);
legend(h,'Reference','State x','Error e')

function x = stepresponse(c,p)
% ------------------------------------------------------
z  = tf('z');                                                     % z-transformation:
G0 = (c(2)+c(3)*z^(-1))/(1-c(1)*z^(-1)) * p.b/(z^10-p.a*z^9);     % Gcontrol * Gsystem
x  = step( G0/(1+G0), p.N-1 );                                    % step response

function f = costfct(c,p)
% ------------------------------------------------------
global x cold
x = stepresponse(c,p);                                            % step response
if p.costQ==1, f = sum( abs(1-x) );                               % cost function
else           f = 1/p.N*sum( [1:p.N]*abs(1-x) ); end
cold = c;

function [C,Ceq] = nlconstr(c,p)
% ------------------------------------------------------
global x cold
if any(c~=cold), x=stepresponse(c,p); end                         % if new c-values
C = max(x-(1+p.ov));  Ceq = [];                                   % nonlin. inequ. constr.
```

Figure 4.14: MATLAB code for optimizing the controller parameters (fmincon).

[7] The control loop (4.92), (4.93) is implemented as z-transfer function. The step response is calculated with the function step.

Cost fct.	max. overshoot	c_1^*	c_2^*	c_3^*	$f(\boldsymbol{c}^*)$
(4.94)	10%	1.000	0.439	-0.369	14.6
(4.94)	0%	1.000	0.439	-0.372	14.8
(4.95)	10%	1.000	0.367	-0.297	1.93
(4.95)	0%	1.000	0.400	-0.333	2.06

Table 4.4: Optimized controller parameters for the control loop (4.92)-(4.93).

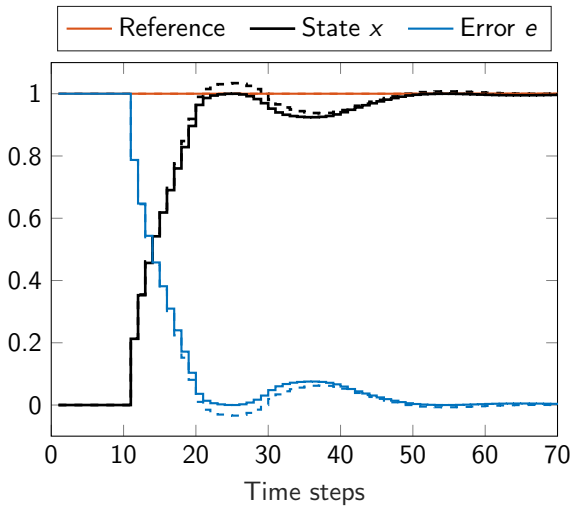


Figure 4.15: Step response of the optimized control loop with cost function (4.94) (overshoot 10%: - -, 0%: –).

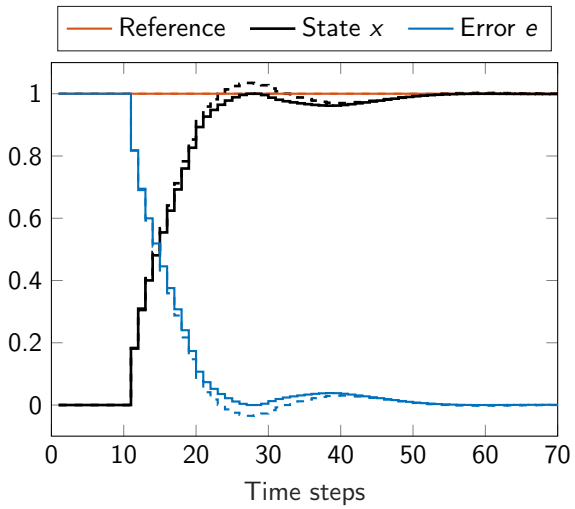


Figure 4.16: Step response of the optimized control loop with cost function (4.95) (overshoot 10%: - -, 0%: –).

4.5.2 Further numerical methods

An alternative to SQP is to replace the original problem (4.82) by one or several unconstrained problems that are solved in a successive manner. This section will give a brief introduction for selected methods, more information can be found in the textbooks at the end of the chapter.

External penalty functions

A straightforward approach is to replace the constrained problem (4.82) by an unconstrained one by adding the constraints (4.82b) und (4.82c) as penalty terms to the cost function

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} P(\boldsymbol{x},\tau) = f(\boldsymbol{x}) + \frac{1}{\tau}\left[\sum_{i=1}^{p}\phi(g_i(\boldsymbol{x})) + \sum_{i=1}^{q}\psi(h_i(\boldsymbol{x}))\right]. \tag{4.96}$$

The functions ϕ and ψ are *external penalty functions* that penalize a violation of the constraints. The penalty function ϕ for the equality constraints $g_i(\boldsymbol{x}) = 0$ is often designed as a power function $\phi(g_i(\boldsymbol{x})) = |g_i(\boldsymbol{x})|^r$ with $r \geq 1$, whereas ψ is typically defined as

$$\psi(h_i(\boldsymbol{x})) = \Big(\max\{0, h_i(\boldsymbol{x})\}\Big)^r, \quad r \geq 1\,. \tag{4.97}$$

Figure 4.17a illustrates the behavior of ψ and the influence of the penalty parameter τ.

The solution $\boldsymbol{x}(\tau)$ of the new unconstrained problem (4.96) will differ from the optimal solution $\boldsymbol{x}^*$ of the original problem (4.82) because the penalty term in (4.96) alters the original cost function (4.82a).

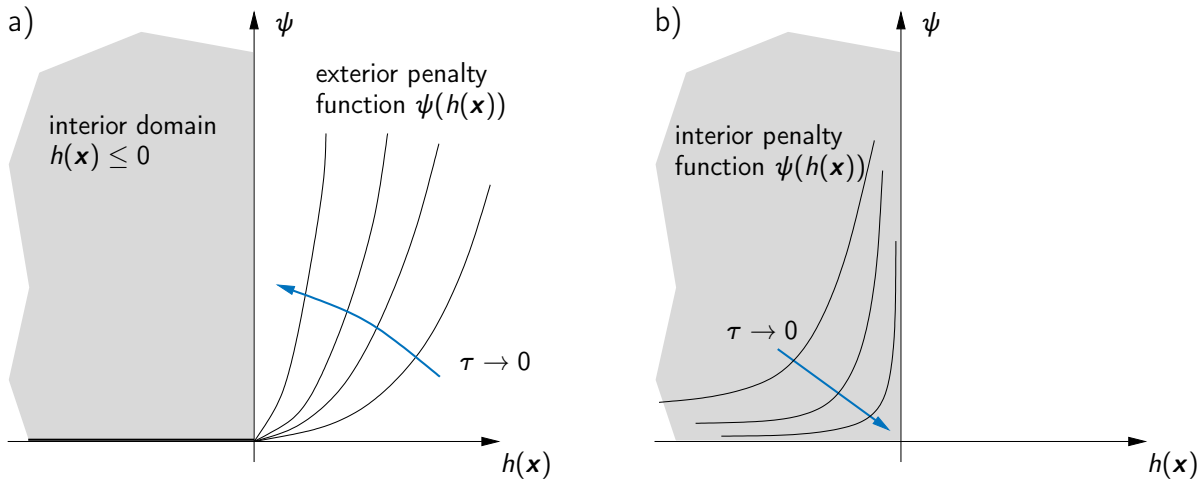


Figure 4.17: Illustration of external and interior penalty function $\psi(h(\boldsymbol{x}))$.

The problem (4.96) is therefore solved in a successive manner for a decreasing sequence $\{\tau^k\}$ of penalty parameters corresponding to an increasingly restrictive penalization of the contraint violation, see Figure 4.17a. Under certain assumptions, it can be shown that the sequence $\{\tau^k\}$ with $\lim_{k\to\infty} \tau^k = 0$ will approach the optimal solution in the limit, i.e.

$$\lim_{k\to\infty} \boldsymbol{x}(\tau^k) = \boldsymbol{x}^* \,. \tag{4.98}$$

A well-known drawback is that problem (4.96) is increasingly ill-conditioned for $\tau \to 0$, which makes it difficult in practice to achieve an accurate approximation of the optimal solution $\boldsymbol{x}^*$.

Interior penalty functions (barrier functions)

Interior penalty functions or *barrier functions* penalize the constraints $h_i(\boldsymbol{x}) \leq 0$ from the interior of the admissible set and approach infinity if the boundary $h_i(\boldsymbol{x}) = 0$ is touched, see Figure 4.17b as well as Figure 4.8. Examples of interior penalty functions are

$$\psi(h_i(\boldsymbol{x})) = \frac{-1}{h_i(\boldsymbol{x})} \quad \text{or} \quad \psi(h_i(\boldsymbol{x})) = -\ln\left(-h_i(\boldsymbol{x})\right) \,. \tag{4.99}$$

Since barrier functions are defined on the interior of the admissible set, only inequality constraints (4.82) can be considered within the unconstrained problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} P(\boldsymbol{x}, \tau) = f(\boldsymbol{x}) + \tau \sum_{i=1}^{q} \psi(h_i(\boldsymbol{x})) \,. \tag{4.100}$$

The successive reduction of the penalty parameter τ reduces the penalization of the admissible set for $h_i(\boldsymbol{x}) < 0$, whereas ψ remains unbounded for $h_i(\boldsymbol{x}) \nearrow 0$, see Figure 4.17b. It can be shown under certain assumptions that the optimal solution $\boldsymbol{x}^*$ is reached in the limit (4.98) for a decreasing sequence $\{\tau^k\}$ with $\lim_{k\to\infty} \tau^k = 0$. For barrier methods, it is important to find a feasible starting point $\boldsymbol{x}^0$ that lies in the interior of the constraints $h_i(\boldsymbol{x}) \leq 0$.

Interior point methods

Interior point methods were already introduced in connection with linear programming in Section 4.3.4. Their extension to nonlinear problems is therefore only touched upon. Interior point methods rely on the idea of perturbing the complementary condition (4.29e) by a parameter

$\tau > 0$, i.e. $\mu_i^* h_i(\boldsymbol{x}^*) + \tau = 0$, in order to ease the numerical solution of the KKT conditions (4.29). As already mentioned in Section 4.3.4, there exists a relation between interior point methods and logarithmic barrier functions in (4.99). However, interior point methods are numerically better conditioned and belong to the most powerful methods in nonlinear optimization in particular for large-scale problems. Further information and details can be found in the comprehensive survey paper [9] and the textbooks [17, 14].

4.6 Lagrangian duality

The previous sections focused on constrained optimization problems from the perspective of the KKT conditions and their numerical or algorithmic solution. In contrast to this, the Lagrangian duality theory considers an alternative optimization problem that is *dual* to the original one. The dual problem has several interesting characteristics that makes the dual problem in some cases easier to solve than the original constrained optimization problem.

This section gives an introduction to Lagrangian duality and the dual formulation and afterwards presents a popular numerical method for solving dual problems, the so-called *augmented Lagrangian method*.

4.6.1 Dual formulation

We consider the nonlinear optimization problem

$$\min_{\boldsymbol{x} \in \mathcal{X}} \quad f(\boldsymbol{x}) \tag{4.101a}$$

$$\text{s.t.} \quad \boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0} \tag{4.101b}$$

$$\boldsymbol{h}(\boldsymbol{x}) \leq \boldsymbol{0} \tag{4.101c}$$

similar to (4.82) with the exception that the constraint functions are written in vector notation $\boldsymbol{g} = [g_1, \dots, g_p]^\mathsf{T}$ and $\boldsymbol{h} = [h_1, \dots, h_q]^\mathsf{T}$ for compactness reasons and the domain of definition $\mathcal{X} \subseteq \mathbb{R}^n$ is introduced for the optimization variables $\boldsymbol{x}$.

According to Definition 4.2, the Lagrangian for the problem (4.101) reads

$$L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^\mathsf{T} \boldsymbol{g}(\boldsymbol{x}) + \boldsymbol{\mu}^\mathsf{T} \boldsymbol{h}(\boldsymbol{x}) \tag{4.102}$$

with the Lagrange multipliers $\boldsymbol{\lambda} \in \mathbb{R}^p$ and $\boldsymbol{\mu} \in \mathbb{R}^q$. The so-called *dual function* is defined as the infimum[8] of the Lagrangian w.r.t. $\boldsymbol{x}$.

Definition 4.4 (Dual function) *The* Lagrangian dual function *or simply* dual function *is defined as*

$$\theta(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \inf_{\boldsymbol{x} \in \mathcal{X}} L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \,. \tag{4.103}$$

[8] The infimum $\inf \mathcal{X}$ of a nonlinear set $\mathcal{X} \subset \mathbb{R}$ denotes the largest lower bound of $\mathcal{X}$. This means that there exists some ξ such that $x \geq \xi$ for all $x \in \mathcal{X}$ and, moreover, for all $\xi' > \xi$ there exists a $x \in \mathcal{X}$ such that $x < \xi'$. The infimum of $\mathcal{X}$ is not necessarily contained in the set $\mathcal{X}$, see e.g. $\inf \mathcal{X} = 0$ with $\mathcal{X} = (0, 1)$ and $0 \notin \mathcal{X}$. Applied to a function $f : \mathcal{U} \to \mathcal{V}$ we have $\inf_{u \in \mathcal{U}} f(u) = \inf \mathcal{V}$. The supremum $\sup \mathcal{X}$ as the smallest upper bound of a set $\mathcal{X}$ is defined accordingly.

Depending on the problem at hand, the dual function $\theta(\boldsymbol{\lambda}, \boldsymbol{\mu})$ can take on the value $-\infty$, e.g. if L is linear in $\boldsymbol{x}$ and $\mathcal{X} = \mathbb{R}^n$, see Example 4.8 below. For this reason, we define the domain on which θ is finite

$$D = \{(\boldsymbol{\lambda}, \boldsymbol{\mu}) \in \mathbb{R}^{p \times q} \,|\, \theta(\boldsymbol{\lambda}, \boldsymbol{\mu}) > -\infty\}\,. \tag{4.104}$$

The dual function has some interesting structural properties on D irrespective of the convexity properties of the cost $f(\boldsymbol{x})$ and constraint functions $\boldsymbol{g}(\boldsymbol{x})$, $\boldsymbol{h}(\boldsymbol{x})$ of the original problem (4.82).

Theorem 4.5 *The set D is convex and the dual function θ is concave on D.*

Proof: For each $\boldsymbol{x}$, $(\boldsymbol{\lambda}, \boldsymbol{\mu})$, $(\boldsymbol{\lambda}', \boldsymbol{\mu}')$, and $\alpha \in [0, 1]$, the Lagrangian satisfies

$$L(\boldsymbol{x}, \alpha\boldsymbol{\lambda} + (1-\alpha)\boldsymbol{\lambda}', \alpha\boldsymbol{\mu} + (1-\alpha)\boldsymbol{\mu}') = \alpha L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) + (1-\alpha) L(\boldsymbol{x}, \boldsymbol{\lambda}', \boldsymbol{\mu}')\,.$$

If we consider the infimum, we can derive the inequality

$$\begin{aligned}\inf_{\boldsymbol{x} \in \mathbb{R}^n} L\Big(\boldsymbol{x}, \alpha\boldsymbol{\lambda} + (1-\alpha)\boldsymbol{\lambda}', \alpha\boldsymbol{\mu} + (1-\alpha)\boldsymbol{\mu}'\Big) &= \inf_{\boldsymbol{x} \in \mathbb{R}^n} \Big(\alpha L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) + (1-\alpha) L(\boldsymbol{x}, \boldsymbol{\lambda}', \boldsymbol{\mu}')\Big) \\ &\geq \alpha \inf_{\boldsymbol{x} \in \mathbb{R}^n} L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) + (1-\alpha) \inf_{\boldsymbol{x} \in \mathbb{R}^n} L(\boldsymbol{x}, \boldsymbol{\lambda}', \boldsymbol{\mu}')\end{aligned}$$

or in terms of the dual function

$$\theta\Big(\alpha\boldsymbol{\lambda} + (1-\alpha)\boldsymbol{\lambda}', \alpha\boldsymbol{\mu} + (1-\alpha)\boldsymbol{\mu}'\Big) \geq \alpha\theta(\boldsymbol{\lambda}, \boldsymbol{\mu}) + (1-\alpha)\theta(\boldsymbol{\lambda}', \boldsymbol{\mu}')\,,$$

which proves the concavity of θ. If $(\boldsymbol{\lambda}, \boldsymbol{\mu})$, $(\boldsymbol{\lambda}', \boldsymbol{\mu}')$ belong to D, then the same holds for $\alpha\boldsymbol{\lambda} + (1-\alpha)\boldsymbol{\lambda}' \in D$ due to $\theta\Big(\alpha\boldsymbol{\lambda} + (1-\alpha)\boldsymbol{\lambda}', \alpha\boldsymbol{\mu} + (1-\alpha)\boldsymbol{\mu}'\Big) > -\infty$, which shows that D is convex. ■

Definition 4.5 (Dual problem) *The* (Lagrangian) dual problem *of (4.101) is defined by*

$$\max_{\boldsymbol{\lambda}, \boldsymbol{\mu}} \quad \theta(\boldsymbol{\lambda}, \boldsymbol{\mu}) \tag{4.105a}$$

$$\text{s.t.} \quad \boldsymbol{\mu} \geq \mathbf{0} \tag{4.105b}$$

with the dual variables $(\boldsymbol{\lambda}, \boldsymbol{\mu})$.

In the sense of the duality theory, the original optimization problem (4.101) is denoted as the *primal problem* with the *primal variables* $\boldsymbol{x}$.

The dual problem is always convex, even if the primal problem (4.101) is non-convex. Another advantage is that the dual problem (4.105) has only linear inequality constraints in contrast to the primal problem (4.82). Hence, it might be advantageous to solve the dual problem (4.105) instead of the primal one. The *augmented Lagrangian method* is a suitable algorithm in this regard that is presented in Section 4.6.2.

As mentioned before, the dual function $\theta(\boldsymbol{\lambda}, \boldsymbol{\mu})$ might evaluate to $-\infty$. In many cases, however, the set D in (4.104) can be determined explicitly and incorporated in the dual formulation as the following example shows.

Example 4.8 (Dual standard form of linear optimization problems) *We consider the linear problem in standard form (also see Section 4.3)*

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \quad \boldsymbol{c}^\mathsf{T}\boldsymbol{x} \tag{4.106a}$$

$$\text{s.t.} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\,, \quad \boldsymbol{x} \geq \boldsymbol{0} \tag{4.106b}$$

with the Lagrangian function

$$L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \boldsymbol{c}^\mathsf{T}\boldsymbol{x} + \boldsymbol{\lambda}^\mathsf{T}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) + \boldsymbol{\mu}^\mathsf{T}(-\boldsymbol{x}) = (\boldsymbol{A}^\mathsf{T}\boldsymbol{\lambda} - \boldsymbol{\mu} + \boldsymbol{c})^\mathsf{T}\boldsymbol{x} - \boldsymbol{\lambda}^\mathsf{T}\boldsymbol{b}\,.$$

The dual problem (4.105) then reads

$$\max_{\boldsymbol{\lambda},\boldsymbol{\mu}} \quad \theta(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \begin{cases} -\boldsymbol{\lambda}^\mathsf{T}\boldsymbol{b} & \text{if } \boldsymbol{A}^\mathsf{T}\boldsymbol{\lambda} - \boldsymbol{\mu} + \boldsymbol{c} = \boldsymbol{0} \\ -\infty & \text{else} \end{cases} \tag{4.107a}$$

$$\text{s.t.} \quad \boldsymbol{\mu} \geq \boldsymbol{0}\,. \tag{4.107b}$$

The dual function $\theta(\boldsymbol{\lambda}, \boldsymbol{\mu})$ follows from Definition (4.103) and is only finite for $\boldsymbol{A}^\mathsf{T}\boldsymbol{\lambda} - \boldsymbol{\mu} + \boldsymbol{c} = \boldsymbol{0}$. This condition can be considered in the formulation of the dual problem (4.107)

$$\max_{\boldsymbol{\lambda}} \quad -\boldsymbol{\lambda}^\mathsf{T}\boldsymbol{b} \tag{4.108a}$$

$$\text{s.t.} \quad \boldsymbol{A}^\mathsf{T}\boldsymbol{\lambda} + \boldsymbol{c} \geq \boldsymbol{0}\,, \tag{4.108b}$$

where the inequality constraint (4.108b) results from combining $\boldsymbol{A}^\mathsf{T}\boldsymbol{\lambda} - \boldsymbol{\mu} + \boldsymbol{c} = \boldsymbol{0}$ and $\boldsymbol{\mu} \geq \boldsymbol{0}$. The formulation (4.108) is often referred to as Lagrangian dual standard form *or in short* dual standard form *in linear optimization.*

Exercise 4.11 *Consider the linear problem (4.106) with the difference that the inequality constraint $\boldsymbol{x} \geq \boldsymbol{0}$ is integrated in the domain of definition $\boldsymbol{x} \in \mathcal{X}$ with $\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}^n \,|\, \boldsymbol{x} \geq \boldsymbol{0}\}$. Determine the dual function (4.103) and show that the resulting dual problem is equivalent to (4.108).*

The following theorem establishes the correspondence between the KKT conditions (4.29) of the primal problem (4.101) and the solution of the dual problem – at least for the case without equality constraints (4.101b) and under the assumption of convexity.

Theorem 4.6 ([14]) *Suppose that the primal (inequality-constrained) optimization problem (4.101a), (4.101c) has a solution $\boldsymbol{x}^*$ and that the functions f and h_i, $i = 1, \ldots, q$ are convex on $\mathbb{R}^n$ and differentiable at point $\boldsymbol{x}^*$. Then, every multiplier $\boldsymbol{\lambda}^*$ satisfying the corresponding KKT conditions at point $\boldsymbol{x}^*$ is a solution of the dual problem.*

If the dual formulation is used for the solution of an optimization problem, then the inverse question related to Theorem 4.6 is when does the solution of (4.105) correspond to the optimal solution of (4.82)? This fundamental question is closely related to the property of *weak and strong duality*.

Theorem 4.7 (Weak duality) *For each primal-dual feasible point $\boldsymbol{x} \in \mathcal{X}_{ad} = \{\boldsymbol{x} \in \mathcal{X} \,|\, \boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0}, \boldsymbol{h}(\boldsymbol{x}) \leq \boldsymbol{0}\}$ and $(\boldsymbol{\lambda}, \boldsymbol{\mu})$ with $\boldsymbol{\mu} \geq \boldsymbol{0}$, the dual function is a lower bound of the cost function, i.e. $\theta(\boldsymbol{\lambda}, \boldsymbol{\mu}) \leq f(\boldsymbol{x})$.*

In particular, weak duality implies that the maximum $\theta^* = \max_{\boldsymbol{\lambda}, \boldsymbol{\mu} \geq \boldsymbol{0}} \theta(\boldsymbol{\lambda}, \boldsymbol{\mu})$ of the dual function is always a lower bound for the optimal cost value $f^* = f(\boldsymbol{x}^*)$ of the primal optimization problem (4.82), i.e.

$$\theta^* \leq f^* \,.$$

The difference $f^* - \theta^*$ between both values is the so-called *duality gap*. If the duality gap is zero, we speak of *strong duality* with

$$\theta^* = f^* \,.$$

If $\boldsymbol{x}^*$ and $(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ is a primal-dual optimal solution and strong duality holds, then the Lagrangian satisfies the *saddle point condition*

$$L(\boldsymbol{x}^*, \boldsymbol{\lambda}, \boldsymbol{\mu}) \leq L(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \leq L(\boldsymbol{x}, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \quad \forall\, \boldsymbol{x} \in \mathcal{X},\, \boldsymbol{\lambda},\, \boldsymbol{\mu} \geq 0 \,. \tag{4.109}$$

Thus, an optimal solution minimizes the Lagrangian w.r.t. the primal variables and maximizes it w.r.t. the dual variables. Vice versa, if $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ satisfies the saddle point condition (4.109), then $\boldsymbol{x}^*$ is primal optimal, $(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ is dual optimal and the duality gap is zero.

Theorem 4.8 (Saddle point) *A primal-dual point $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ with $\boldsymbol{\mu}^* \geq \boldsymbol{0}$ is optimal if and only if the saddle point condition (4.109) is satisfied for all $\boldsymbol{x} \in \mathcal{X}$ and $(\boldsymbol{\lambda}, \boldsymbol{\mu}) \in \mathbb{R}^{p \times q}$ with $\boldsymbol{\mu} \geq \boldsymbol{0}$.*

Example 4.9 *We consider the non-convex optimization problem*

$$\min_{x \in \mathbb{R}} \quad f(x) = 2x - \tfrac{1}{2}x^2 - \tfrac{2}{3}x^3 + \tfrac{1}{4}x^4 \tag{4.110a}$$

$$\text{s.t.} \quad h(x) = 4x^2 - 1 \leq 0 \tag{4.110b}$$

with the Lagrangian $L(x, \mu) = 2x - \frac{1}{2}x^2 - \frac{2}{3}x^3 + \frac{1}{4}x^4 + \mu(4x^2 - 1)$. The left side of Figure 4.18 shows $f(x)$ and $h(x)$ as well as the admissible set $\mathcal{X}_{ad} = \{x \in \mathbb{R} \,|\, -\frac{1}{2} \leq x \leq \frac{1}{2}\}$ as the grey shaded area. In addition, $L(\cdot, \mu)$ is plotted for various values of $\mu \in [0, 3]$ along with the respective minima that correspond to the profile of the dual function $\theta(\mu)$. In accordance with Theorem 4.7, $\theta(\mu)$ is smaller than the cost function $f(x)$ in the admissible set $\mathcal{X}_{ad}$.

The right side of Figure 4.18 plots the dual function $\theta(\mu)$ over μ as well as the optimal value f^* of the cost function as the solution of (4.110). The maximum $\theta^* = \max_{\mu \geq 0} \theta(\mu)$ equals the optimal cost f^*, which shows that the duality gap is zero and strong duality holds. In addition, Figure 4.19 shows a surface plot of the Lagrangian $L(x, \lambda)$. The red lines illustrate that the saddle point condition (4.109) is satisfied at the primal-dual optimal point (x^*, μ^*).

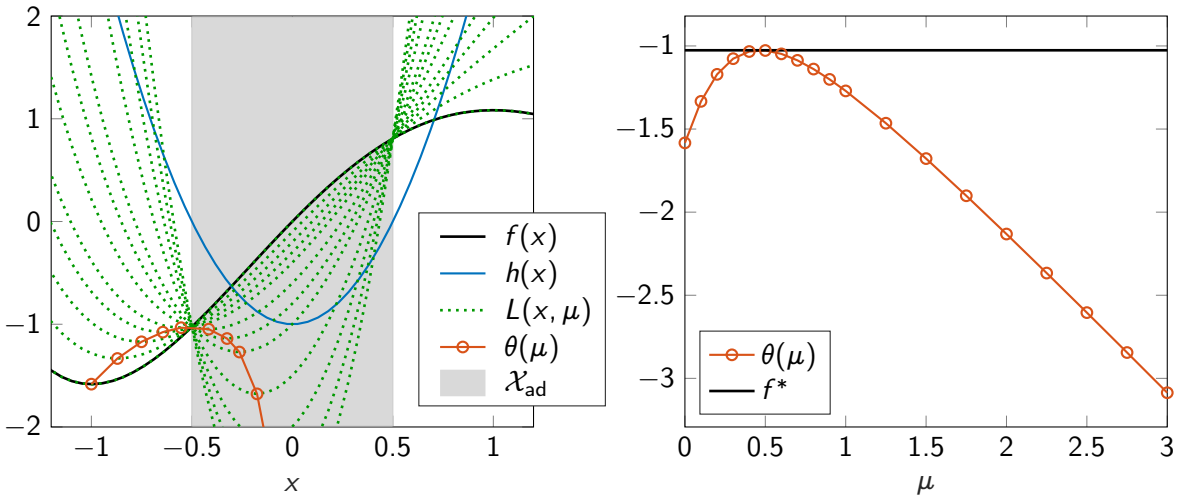


Figure 4.18: Functions $f(x)$, $h(x)$, and $L(x,\mu)$ for discrete values of $\mu \in [0,3]$ as well as the dual function $\theta(\mu)$ as lower bound on the optimal cost f^* (Example 4.9).

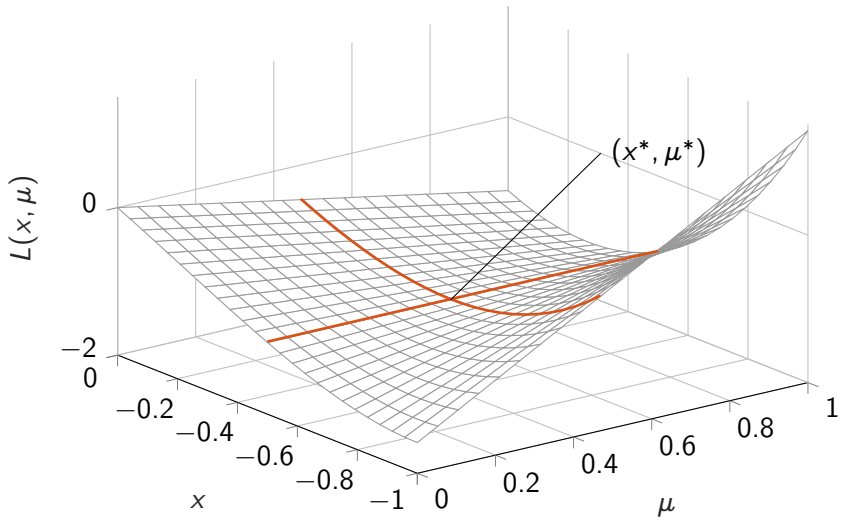


Figure 4.19: Graphical validation of the saddle point condition (4.109) for Example 4.9.

The proof of strong duality typically relies on convexity, although Example 4.9 shows that strong duality can also hold for non-convex problems. Remember that a nonlinear optimization problem with linear equality constraints

$$\min_{\boldsymbol{x}\in\mathcal{X}} \quad f(\boldsymbol{x}) \tag{4.111a}$$
$$\text{s.t.} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{4.111b}$$
$$\boldsymbol{h}(\boldsymbol{x}) \leq \boldsymbol{0} \tag{4.111c}$$

is convex if the functions f and h_i, $i = 1,\dots,q$ are convex and if $\mathcal{X}$ is convex as well, see Definition 2.7 and Theorem 2.5. A further assumption that is often used in relation with strong duality is the constraint qualification of *Slater*

$$\exists\, \boldsymbol{x} \in \text{relint}(\mathcal{X}): \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \quad h_i(\boldsymbol{x}) < 0, \quad i = 1,\dots,q \tag{4.112}$$

that implies that at least one interior point $\boldsymbol{x}$ of the primal problem must exist. [9]

[9] If $\mathcal{X}$ contains at least one interior point, then $\text{relint}(\mathcal{X}) = \text{int}(\mathcal{X})$.

Theorem 4.9 (Strong duality) *Let the primal optimization problem (4.111) be convex for $\boldsymbol{x} \in \mathbb{R}^n$ and feasible. Moreover, suppose that the minimum $f^* = f(\boldsymbol{x}^*)$ is finite and the constraint qualification (4.112) is satisfied. Then, the following statements are true:*

- *There exists no duality gap, i.e. $\theta^* = f^*$.*
- *There exists at least one pair of dual variables with $\theta(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) = \theta^*$.*
- *The saddle point condition (4.109) is satisfied.*
- *If $\boldsymbol{x}^*$ is a global primal solution, then $(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ is a global solution of the dual problem (4.105).*

The assumptions in Theorem 4.9 do not automatically imply the uniqueness of the primal solution $\boldsymbol{x}^*$. This can be achieved by additionally assuming strict convexity of the Lagrangian $L(\cdot, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ [14]. Moreover, convexity alone is not sufficient to guarantee strong duality. The following example illustrates this point.

Example 4.10 (Convex problem with duality gap [3]) *Consider the problem*

$$\min_{\boldsymbol{x} \in \mathcal{X}} \quad e^{-\sqrt{x_1 x_2}} \tag{4.113a}$$

$$\text{s.t.} \quad x_1 = 0 \tag{4.113b}$$

with the convex cost function $f(\boldsymbol{x}) = e^{-\sqrt{x_1 x_2}}$ on the domain of definition $\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}^2 \,|\, \boldsymbol{x} \geq \boldsymbol{0}\}$. The minimum point is $\boldsymbol{x}^* = [0, x_2^*]^\mathsf{T}$ with $f^* = f(\boldsymbol{x}^*) = 1$. With the Lagrangian $L(\boldsymbol{x}, \lambda) = e^{-\sqrt{x_1 x_2}} + \lambda x_1$, the dual function is defined by

$$\theta(\lambda) = \inf_{\boldsymbol{x} \in \mathcal{X}} \left(e^{-\sqrt{x_1 x_2}} + \lambda x_1 \right) = \begin{cases} 0 & \text{if } \lambda \geq 0 \\ -\infty & \text{else}. \end{cases} \tag{4.114}$$

The first case implies $x_1 x_2 \to \infty$ and in addition $x_1 \to 0$ if $\lambda > 0$. This becomes clear in view of $\boldsymbol{x} = [1/\xi, \xi^2]^\mathsf{T}$ and

$$\lim_{\xi \to \infty} L(\boldsymbol{x}, \lambda \geq 0) = \lim_{\xi \to \infty} e^{-\sqrt{\xi}} + \frac{\lambda}{\xi} = 0\,.$$

We thus get $\theta^* = \max_\lambda \theta(\lambda) = 0$ and $f^* - \theta^* = 1 > 0$, which shows that a duality gap exists although the problem is convex. The reason for this is that Slater's constraint qualification (4.112) is not satified.

Exercise 4.12 *Investigate strong duality for the optimization problems*

$$\min_{\boldsymbol{x} \in \mathbb{R}} \quad x^4 \qquad\qquad \min_{\boldsymbol{x} \in \mathbb{R}} \quad x^2$$
$$\text{s.t.} \quad x^2 \geq 1 \qquad\qquad \text{s.t.} \quad x^4 \geq 1\,.$$

If strong duality holds, the dual formulation can be used for the solution of the primal optimization problem (4.101) or (4.111). Formally, the computation of an optimal solution $\boldsymbol{x}^*$ requires to determine the dual function as the solution of (4.103) and its maximization as the dual problem (4.105) to obtain $(\boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$. The optimal primal variables $\boldsymbol{x}^*$ then follow as the minimizing arguments of (4.103). The augmented Lagrangian method solves this max-min-problem in an iterative manner.

4.6.2 Augmented Lagrangian method

The augmented Lagrangian method can be applied to equality-constrained nonlinear optimization problems

$$\min_{x \in \mathcal{X}} \quad f(\boldsymbol{x}) \tag{4.115a}$$

$$\text{s.t.} \quad \boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0} \tag{4.115b}$$

with the constraint functions $\boldsymbol{g} = [g_1 \dots q_p]^\mathrm{T}$. The extension to inequality constraints is described in Section 4.6.3. The *augmented Lagrangian function* is defined by

$$L_c(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^\mathrm{T} \boldsymbol{g}(\boldsymbol{x}) + \frac{c}{2} \|\boldsymbol{g}(\boldsymbol{x})\|^2 \tag{4.116}$$

with the Lagrange multipliers $\boldsymbol{\lambda}^\mathrm{T} \in \mathbb{R}^p$ and an additional quadratic penalty term with parameter $c \geq 0$ that penalizes the deviation of the constraint $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{0}$. The dual problem based on the augmented Lagrangian function (4.116) reads

$$\max_{\boldsymbol{\lambda}} \inf_{x \in \mathcal{X}} L_c(\boldsymbol{x}, \boldsymbol{\lambda}) \,. \tag{4.117}$$

If strong duality holds and $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ is a primal-dual solution, then the augmented Lagrangian function satisfies the saddle point condition

$$L_c(\boldsymbol{x}^*, \boldsymbol{\lambda}) \leq L_c(\boldsymbol{x}^*, \boldsymbol{\lambda}^*) \leq L_c(\boldsymbol{x}, \boldsymbol{\lambda}^*) \quad \forall\, \boldsymbol{x}, \boldsymbol{\lambda} \,. \tag{4.118}$$

The inverse statement is particularly important for the augmented Lagrangian method: if (4.118) is satisfied, then $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ is a primal-dual solution and the duality gap is zero.

The additional quadratic penalty of the constraint in the Lagrangian function (4.116) has a positive effect on the numerical conditioning and also affects the duality of the problem. In particular, it can be shown under certain assumptions that strong duality holds even for non-convex problems if the penalty parameter $c > 0$ is sufficiently large [16]. The following theorem [14] provides the interesting result that the penalty term does not need to be chosen unnecessarily large.

Theorem 4.10 *Let $\boldsymbol{x}^*$ be a local solution of (4.115), for which the constraint qualification (4.26) as well as the KKT conditions (4.29) and the sufficient optimality conditions (4.37) for $\boldsymbol{\lambda} = \boldsymbol{\lambda}^*$ are satisfied. Then, there exists a lower bound $\bar{c} > 0$ such that $L_c(\boldsymbol{x}, \boldsymbol{\lambda}^*)$ has a strict local minimum at $\boldsymbol{x} = \boldsymbol{x}^*$ for all $c \geq \bar{c}$.*

The augmented Lagrangian algorithm is based on the idea to compute $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ by iteratively approaching the saddle point (4.118). Table 4.5 summarizes the basic algorithm of the augmented Lagrangian method. If $\boldsymbol{x}$ is unconstrained, i.e. $\mathcal{X} = \mathbb{R}^n$, numerical methods of unconstrained optimization can be used for the solution, see Chapter 3. Box constraints of the form $\mathcal{X} = [\boldsymbol{x}^-, \boldsymbol{x}^+]$ can be considered via projection onto the respective boundary. The minimization problem is often solved approximately, e.g. by using a convergence criterion of the form

$$\|\nabla_{\boldsymbol{x}} L_{c^k}(\boldsymbol{x}^k, \boldsymbol{\lambda}^k)\| \leq \varepsilon_L^k \quad \text{with} \quad 0 < \varepsilon_L^k < \varepsilon_L^{k-1} \tag{4.119}$$

or by monitoring the convergence according to $\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\| \leq \varepsilon_x \|\boldsymbol{x}^{k-1}\|$ with $\varepsilon_x < 1$.

Initialization:

$k_{\max}$	Max. number of iterations
$\boldsymbol{\lambda}^1$	Initial dual point
$c^1 > \mathbf{0}$	Initial penalty parameter
$\varepsilon_g > 0$	Convergence tolerance

for $k = 1, \ldots, k_{\text{mcax}}$ **do**

$\boldsymbol{x}^k = \arg\min_{\boldsymbol{x}\in\mathcal{X}} L_{c^k}(\boldsymbol{x}, \boldsymbol{\lambda}^k)$	Update of primal variables (minimization)
if $\|\boldsymbol{g}^k\| \leq \varepsilon_g$ **or** $k = k_{\max}$ **then**	Convergence check with $\boldsymbol{g}^k = \boldsymbol{g}(\boldsymbol{x}^k)$
return $\boldsymbol{x}^k$	
end	
$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + c^k \boldsymbol{g}^k$	Update of dual variables (steepest ascent)
if $k > 1$ **then**	
$c^{k+1} = \psi_c\left(c^k, \|\boldsymbol{g}^k\|, \|\boldsymbol{g}^{k-1}\|\right)$	Update of penalty parameter
end	

end

Table 4.5: Structure of augmented Lagrangian algorithm.

The dual variables $\boldsymbol{\lambda}$ in Table 4.5 are updated in the direction of steepest ascent with $\nabla_{\boldsymbol{\lambda}} L_c = \boldsymbol{g}(\boldsymbol{x}^k)$. The step size is not computed via a line search because L_c is linear in λ which implies that no finite maximum point exists. Instead, the step size corresponds to the penalty parameter c, which can be illustrated by comparing the stationarity conditions for the minimizer $\boldsymbol{x}^k$ of the augmented Lagrangian (4.116)

$$\nabla_{\boldsymbol{x}} L_{c^k}(\boldsymbol{x}^k, \boldsymbol{\lambda}^k) = \nabla f(\boldsymbol{x}^k) + \sum_{i=1}^{p} \left(\lambda_i^k + c^k g_i(\boldsymbol{x}^k)\right) \nabla g_i(\boldsymbol{x}^k) = \mathbf{0} \tag{4.120}$$

with the optimal solution of (4.115)

$$\nabla_{\boldsymbol{x}} L(\boldsymbol{x}^*, \boldsymbol{\lambda}^*) = \nabla f(\boldsymbol{x}^*) + \sum_{i=1}^{p} \lambda_i^* \nabla g_i(\boldsymbol{x}^*) = \mathbf{0}\,. \tag{4.121}$$

This shows that the update law $\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + c^k \boldsymbol{g}(\boldsymbol{x}^k)$ in Table 4.5 corresponds to the optimal multiplier $\boldsymbol{\lambda}^*$. Another illustration for the choice of c as step size follows from considering the *proximal maximization problem*

$$\boldsymbol{\lambda}^{k+1} = \arg\max_{\boldsymbol{\lambda}} \left(L_{c^k}(\boldsymbol{x}^k, \boldsymbol{\lambda}) - \frac{1}{2c^k}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^k\|^2\right) \tag{4.122}$$

that penalizes the distance from the previous iteration point as a trade-off to the maximization of L_c. The solution of this strictly convex problem corresponds to the update law in Table 4.5.

The update of the penalty parameter is often done heuristically. If the constraint violation $\|\boldsymbol{g}^k\| = \|\boldsymbol{g}(\boldsymbol{x}^k)\|$ is not reduced sufficiently fast, it is reasonable to increase c^k. A simple update rule therefore is

$$c^{k+1} = \psi_c\left(c^k, \|\boldsymbol{g}^k\|, \|\boldsymbol{g}^{k-1}\|\right) = \begin{cases} \beta c^k & \text{if } \|\boldsymbol{g}^k\| > \gamma \|\boldsymbol{g}^{k-1}\| \\ c^k & \text{else} \end{cases} \tag{4.123}$$

with $\gamma < 1$ and $\beta > 1$. Recommended values are $\gamma = 0.25$ and $\beta \in [5, 10]$, see [3]. As already mentioned, the minimization problem $\min_{\boldsymbol{x} \in \mathcal{X}} L_{c^k}(\boldsymbol{x}, \boldsymbol{\lambda}^k)$ in Table 4.5 is often solved approximately. In this case, an adaptive update of the penalty parameter c^k in both directions that also allows a temporary reduction of $c^{k+1} < c^k$ has shown to be advantageous in practice, see e.g. LANCELOT [5] or the MPC toolbox GRAMPC [6]. In particular, a decrease of c^k makes sense if the constraints already satisfy the convergence criterion $\|\boldsymbol{g}^k\| \leq \varepsilon_g$ but the minimization of $L_{c^k}(\boldsymbol{x}, \boldsymbol{\lambda}^k)$ is not yet sufficiently accurate in terms of the criterion (4.119).

The following theorem [3, 14] proves that the augmented Lagrangian algorithm has linear convergence properties at least locally around an optimal solution $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ and for sufficiently large penalty parameters.

Theorem 4.11 (Augmented Lagrangian algorithm) *Let the assumptions of Theorem 4.10 be satisfied at the point $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ und suppose that $\bar{c} > 0$ is chosen according to Theorem 4.10. Then, there exist constants $\delta_c, \delta_x, C > 0$ such that for all $\boldsymbol{\lambda}^k$ and c^k with $\|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^*\| \leq \delta_c \bar{c}$ and $c^k \geq \bar{c}$ the problem*

$$\begin{aligned} \min_{\boldsymbol{x} \in \mathcal{X}} \quad & L_{c^k}(\boldsymbol{x}, \boldsymbol{\lambda}^k) \\ \text{s.t.} \quad & \|\boldsymbol{x} - \boldsymbol{x}^*\| \leq \delta_x \end{aligned}$$

has a unique solution $\boldsymbol{x}^k$. Moreover, $\boldsymbol{x}^k$ and $\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + c^k \boldsymbol{g}(\boldsymbol{x}^k)$ satisfy

$$\|\boldsymbol{x}^k - \boldsymbol{x}^*\| \leq C \|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^*\| / c^k \,, \quad \|\boldsymbol{\lambda}^{k+1} - \boldsymbol{\lambda}^*\| \leq C \|\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^*\| / c^k \,. \tag{4.124}$$

Exercise 4.13 ([3]) *Investigate the solution behavior of the augmented Lagrangian algorithm for the example problems*

$$\begin{aligned} \min_{\boldsymbol{x} \in \mathbb{R}^2} \quad & \tfrac{1}{2}(x_1^2 + x_2^2) \\ \text{s.t.} \quad & x_1 = 1 \end{aligned} \qquad \begin{aligned} \min_{\boldsymbol{x} \in \mathbb{R}^2} \quad & \tfrac{1}{2}(-x_1^2 + x_2^2) \\ \text{s.t.} \quad & x_1 = 1 \end{aligned}$$

and determine the lower bound $\bar{c} > 0$ of the penalty parameter $c^k \geq \bar{c}$ for the convergence of the algorithm.

4.6.3 Consideration of inequality constraints

In the last section, the augmented Lagrangian method was introduced for equality-constrained problems (4.115). Inequality constraints

$$\boldsymbol{h}(\boldsymbol{x}) \leq \boldsymbol{0} \tag{4.125}$$

with $\boldsymbol{h} = [h_1, \ldots, h_q]^\mathrm{T}$ can be added to the equality constraints (4.115b) by introducing slack variables $\boldsymbol{s} \geq 0$

$$\hat{\boldsymbol{g}}(\boldsymbol{x}, \boldsymbol{s}) = \begin{bmatrix} \boldsymbol{g}(\boldsymbol{x}) \\ \boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{s} \end{bmatrix} = \boldsymbol{0} \,. \tag{4.126}$$

The augmented Lagrangian function then reads

$$L_c(\boldsymbol{x}, \boldsymbol{s}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^\mathrm{T} \hat{\boldsymbol{g}}(\boldsymbol{x}, \boldsymbol{s}) + \frac{c}{2} \|\hat{\boldsymbol{g}}(\boldsymbol{x}, \boldsymbol{s})\|^2 \tag{4.127}$$

with the multipliers $\boldsymbol{\lambda}^\mathsf{T} = \left[\boldsymbol{\lambda}_{\boldsymbol{g}}^\mathsf{T}, \boldsymbol{\lambda}_{\boldsymbol{h}}^\mathsf{T}\right] \in \mathbb{R}^{p+q}$ and the penalty parameter $c > 0$. The corresponding dual problem (4.117) is defined by

$$\max_{\boldsymbol{\lambda}} \inf_{\boldsymbol{x}\in\mathcal{X},\boldsymbol{s}\geq\mathbf{0}} L_c(\boldsymbol{x}, \boldsymbol{s}, \boldsymbol{\lambda})\,. \tag{4.128}$$

The inner minimization w.r.t. $\boldsymbol{s} \geq \mathbf{0}$ reduces to the strictly convex problem ($c > 0$)

$$\min_{\boldsymbol{s}\geq\mathbf{0}} \left(\boldsymbol{\lambda}_{\boldsymbol{h}}^\mathsf{T}\Big(\boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{s}\Big) + \frac{c}{2}\|\boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{s}\|^2\right)\,. \tag{4.129}$$

Its solution follows from the stationarity condition and projection onto the admissible set $\boldsymbol{s} \geq \mathbf{0}$

$$\boldsymbol{s} = \mathbf{max}\left\{\mathbf{0}, -\boldsymbol{\lambda}_{\boldsymbol{h}}/c - \boldsymbol{h}(\boldsymbol{x})\right\}\,, \tag{4.130}$$

where the vector-valued **max**-function is to be understood elementwise. Inserting (4.130) into (4.126) leads to the new form of the equality constraints

$$\bar{\boldsymbol{g}}_c(\boldsymbol{x}, \boldsymbol{\lambda}) = \begin{bmatrix} \boldsymbol{g}(\boldsymbol{x}) \\ \mathbf{max}\left\{\boldsymbol{h}(\boldsymbol{x}), -\boldsymbol{\lambda}_{\boldsymbol{h}}/c\right\} \end{bmatrix} = \mathbf{0} \tag{4.131}$$

as well as the adapted augmented Lagrangian function

$$\bar{L}_c(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^\mathsf{T}\bar{\boldsymbol{g}}_c(\boldsymbol{x}, \boldsymbol{\lambda}) + \frac{c}{2}\|\bar{\boldsymbol{g}}_c(\boldsymbol{x}, \boldsymbol{\lambda})\|^2 \tag{4.132}$$

for the dual problem

$$\max_{\boldsymbol{\lambda}} \inf_{\boldsymbol{x}\in\mathcal{X}} \bar{L}_c(\boldsymbol{x}, \boldsymbol{\lambda})\,. \tag{4.133}$$

In summary, inequality constraints can be incorporated into the augmented Lagrangian algorithm by replacing $\boldsymbol{g}$ und L_c in Table 4.5 with (4.131) and (4.132) and adapting the dimension of $\boldsymbol{\lambda}$ accordingly. Note that the new equality constraints (4.131) now depend on the multipliers $\boldsymbol{\lambda}$ and the penalty parameter c.

4.6.4 Remarks

The advantage of the augmented Lagrangian method is that the inner minimization problem

$$\inf_{\boldsymbol{x}\in\mathcal{X}} L_c(\boldsymbol{x}, \boldsymbol{\lambda})$$

is often easier zu solve than the original constrained optimization problem due to the fact that it is only constrained to the set $\mathcal{X}$. Over the last years, the dual formulation became popular in connection with variants of the gradient method, especially the accelerated gradient method [13]. A survey over different primal-dual solution methods is given in [11].

The rather simple structure of the augmented Lagrangian method also offers advantages for the implementation on embedded systems or hardware with limited ressources such as electronic control units (ECU) in the automotive sector or programmable logic controllers (PLC) in automation systems. Embedded systems require algorithms that allow for a fast computation of single iterations (at the price of suboptimality) with numerical robustness despite limited numerical accuracy [7, 8].

The duality theory is also important for distributed optimization problems and networked systems as it enables the decoupling of central optimization problem and their distributed or parallelized solution. A popular method in this regard is the *Alternating Direction Method of Multipliers (ADMM)* [10, 4] for optimization problems of the form

$$\min_{\boldsymbol{x},\boldsymbol{y}} \quad f_1(\boldsymbol{x}) + f_2(\boldsymbol{y}) \tag{4.134a}$$

$$\text{s.t.} \quad \underbrace{\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{b}}_{\boldsymbol{g}(\boldsymbol{x},\boldsymbol{y})} = \boldsymbol{0} \tag{4.134b}$$

with the optimization variables $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}^{n_1 \times n_2}$, the cost functions $f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{y})$ and the coupling conditions (4.134b). With the augmented Lagrangian function

$$L_c(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\lambda}) = f_1(\boldsymbol{x}) + f_2(\boldsymbol{y}) + \boldsymbol{\lambda}^\mathsf{T}\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}) + \frac{c}{2}\|\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y})\|^2,$$

the ADMM iteration scheme is defined by

$$\boldsymbol{x}^{k+1} = \arg\min_{\boldsymbol{x}} L_{c^k}(\boldsymbol{x}, \boldsymbol{y}^k, \boldsymbol{\lambda}^k) \tag{4.135a}$$

$$\boldsymbol{y}^{k+1} = \arg\min_{\boldsymbol{y}} L_{c^k}(\boldsymbol{x}^{k+1}, \boldsymbol{y}, \boldsymbol{\lambda}^k) \tag{4.135b}$$

$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + c^k\boldsymbol{g}^{k+1} \tag{4.135c}$$

$$c^{k+1} = \psi_c\left(c^k, \|\boldsymbol{g}^k\|, \|\boldsymbol{g}^{k+1}\|\right) \tag{4.135d}$$

with $\boldsymbol{g}^k = \boldsymbol{g}(\boldsymbol{x}^k, \boldsymbol{y}^k)$. The sequential solution of the optimization problems for $\boldsymbol{x}$ and $\boldsymbol{y}$ ("alternating directions") enables the decomposition of the overall problem. ADMM has become popular in the field of distributed model predictive control of networked systems or in image processing and digital communication for solving high-dimensional optimization problems in a distributed manner.

4.7 Software overview

The following lines give a selection of software tools for solving linear, quadratic, and nonlinear optimization problems.

Linear optimization

- linprog: MATLABOptimization Toolbox (commercial)
- CPLEX (commercial)
  `http://www.ilog.com/products/cplex`
- GLPK: "GNU Linear Programming Kit" (free)
  `http://www.gnu.org/software/glpk`
- lp_solve: Mixed-Integer Lineare Optimierung (free)
  `http://lpsolve.sourceforge.net`
- CVXGEN (commercial, academic licence available)
  `http://cvxgen.com`

Quadratic optimization

- quadprog: MATLAB Optimization Toolbox (commercial)
- qpOASES (free)
  `http://github.com/coin-or/qpOASES`
- OOQP (free)
  `http://pages.cs.wisc.edu/~swright/ooqp`
- CPLEX (commercial)
  `http://www.ilog.com/products/cplex`
- LOQO (commercial)
  `http://vanderbei.princeton.edu/loqomenu.html`
- CVXGEN (commercial, academic licence available)
  `http://www.cvxgen.com`

Nonlinear optimization

- fmincon: MATLABOptimization Toolbox (commercial)
- IPOPT (free)
  `https://github.com/coin-or/Ipopt`
- SNOPT (commercial, student licence available)
  `http://ccom.ucsd.edu/~optimizers`
- LOQO (commercial)
  `http://vanderbei.princeton.edu`
- MINOS (commercial)
  `http://www.sbsi-sol-optimize.com/asp/sol_product_minos.htm`

Modeling languages & automatic code generation

Most of the mentioned optimization tools have an interface for MATLAB or for the modeling language AMPL (e.g. LOQO, GLPK, IPOPT) or GAMS (e.g. MINOS). These languages offer a symbolic syntax to formulate optimization problems. Moreover, automatic code generation as e.g. offered by CasADi is of growing importance in parameter and dynamic optimization.

- AMPL: "A Mathematical Programming Language"
  `http://www.ampl.com`
- GAMS: "General Algebraic Modeling System"
  `http://www.gams.com`
- CasADi: Code generation including automatic differentiation
  `https://web.casadi.org`

4.8 References

[1] G. Allaire. *Numerical Analysis and Optimization*. Oxford: Oxford University Press, 2007.

[2] J.A.E. Andersson, J. Gillis, G. Horn, J.B. Rawlings, and M. Diehl. "CasADi – A software framework for nonlinear optimization and optimal control". In: *Mathematical Programming Computation* (2018). Online verfügbar.

[3] D.P. Bertsekas. *Nonlinear Programming*. Belmont. Athena Scientific, 1999.

[4] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. "Distributed optimization and statistical learning via the alternating direction method of multipliers". In: *Foundations and Trends in Machine Learning* 3.1 (2011), pp. 1–122.

[5] A. R. Conn, G.I.M. Gould, and P. L. Toint. *LANCELOT: A Fortran Package for Large-Scale Nonlinear Optimization (Release A)*. Berlin, Germany: Springer-Verlag, 1992.

[6] T. Englert, A. Völz, F. Mesmer, S. Rhein, and K. Graichen. "A software framework for embedded nonlinear model predictive control using a gradient-based augmented Lagrangian approach (GRAMPC)". In: *Optimization and Engineering* 20.3 (2019), pp. 769–809. DOI: 10.1007/s11081-018-9417-2.

[7] H.J. Ferreau, S. Almér, R. Verschueren, M. Diehl, D. Frick, A. Domahidi, J.L. Jerez, G. Stathopoulos, and C. Jones. "Embedded optimization methods for industrial automatic control". In: *Proc. 20th IFAC World Congress*. 2017, pp. 13736–13751.

[8] R. Findeisen, K. Graichen, and M. Mönnigmann. "Eingebettete Optimierung in der Regelungstechnik – Grundlagen und Herausforderungen". In: *at – Automatisierungstechnik* 66.11 (2018), pp. 877–902.

[9] A. Forsgren, P.E. Gill, and M.H. Wright. "Interior methods for nonlinear optimization". In: *SIAM Review* 44.4 (2002), pp. 525–597.

[10] M. Fortin and R. Glowinski, eds. *Augmented Lagrangian Methods: Applications to the Solution of Boundary-Value Problems*. Amsterdam: North-Holland, 1983.

[11] D. Kouzoupis, H.J. Ferreau, H. Peyrl, and M. Diehl. "First-order methods in embedded nonlinear model predictive control". In: *Proc. 2015 European Control Conference (ECC)*. 2015, pp. 2617–2622.

[12] J. Mattingley and S. Boyd. "CVXGEN: A code generator for embedded convex optimization". In: *Optimization and Engineering* 13.1 (2012), pp. 1–27.

[13] Y. Nesterov. "Introductory lectures on convex optimization: A basic course". In: *Applied Optimization*. Ed. by P.M. Paradalos and D.W. Hearn. Vol. 87. Applied Optimization. Norwell, MA, USA: Kluwer Academic Publishers, 2004.

[14] J. Nocedal and S.J. Wright. *Numerical Optimization*. New York: Springer, 2006.

[15] M. Papageorgiou. *Optimierung*. München: Oldenbourg, 1991.

[16] R.T. Rockafellar. "Augmented Lagrange multiplier functions and duality in nonconvex programming". In: *SIAM Journal on Control* 12.2 (1974), pp. 268–285.

[17] S.J. Wright. *Primal-Dual Interior-Point Methods*. Society for Industrial and Applied Mathematics (SIAM), 1997.

5 Dynamic optimization

The previous chapters focused on parameter optimization problems with the optimization variables $\boldsymbol{x}$ being elements of the Euclidean space $\mathbb{R}^n$. In what follows, we focus on dynamic optimization problems, where a *functional* is minimized w.r.t. a function of an independent variable, in most cases a function of time. The most common application of dynamic optimization is the computation of an optimal control trajectory $\boldsymbol{u}^*(t)$ for a dynamical system based on the solution of an *optimal control problem*.

5.1 Problem statement

A quite general class of optimal control problems is represented by

$$\boxed{\begin{aligned} \min_{\boldsymbol{u}(\cdot)} \quad & J(\boldsymbol{u}) = V(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} l(\boldsymbol{x}(t), \boldsymbol{u}(t), t)\,\mathrm{d}t && \text{(5.1a)} \\ \text{s.t.} \quad & \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t)\,, \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0 && \text{(5.1b)} \\ & \boldsymbol{g}(\boldsymbol{x}(t_f), t_f) = \boldsymbol{0} && \text{(5.1c)} \\ & \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \leq \boldsymbol{0} \quad \forall t \in [t_0, t_f]\,. && \text{(5.1d)} \end{aligned}}$$

The problem is characterized by the cost functional (5.1a), the system dynamics (5.1b) with state $\boldsymbol{x} \in \mathbb{R}^n$ and control $\boldsymbol{u} \in \mathbb{R}^m$, the terminal condition (5.1c), the constraints (5.1d) as well as the end time t_f. A standing assumption we have to impose – similar to the parameter optimization case – is that all functions are at least continuously differentiable in their arguments.

Note that the cost functional (5.1a) formally depends on the control and the state. However, $J(\boldsymbol{u})$ is usually written in sole dependency of the control, if it is assumed that the state trajectory $\boldsymbol{x}(t)$ is uniquely determined by the dynamics (5.1b) for a given control trajectory $\boldsymbol{u}(t)$, $t \in [0, T]$.

Terminal condition (5.1c)

The dynamics and boundary conditions in (5.1b) and (5.1c) define a *transition problem* from the initial state $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$ to the terminal condition (5.1c) at the end time $t = t_f$. The general nonlinear form (5.1c) [1] often breaks down to partial terminal conditions of the form

$$\boxed{x_i(t_f) = x_{f,i}\,, \quad i \in \mathcal{I}_f\,.} \tag{5.2}$$

The set $\mathcal{I}_f$ includes the indices of the state variables x_i that are fixed at the end time t_f. If the full state vector $\boldsymbol{x}(t_f) = \boldsymbol{x}_f$ is fixed at t_f, we have $\mathcal{I}_f = \{1, \ldots, n\}$. The other special

[1] An example of a nonlinear terminal condition is the optimization of the flight trajectory of a space launcher rocket that brings a satellite to a geostationary orbit.

case is a free terminal state with $\mathcal{I}_f = \{\}$. In this chapter, we will mainly focus on the partial terminal condition (5.2). The general form (5.1c) is briefly addressed in Section 5.3.6.

Cost functional (5.1a)

In general, the transition problem (5.1b), (5.1c) has infinitely many solutions $\boldsymbol{u}(t)$, $t \in [t_0, t_f]$, from which an *optimal control* $\boldsymbol{u}^*(t)$ is sought that minimizes the cost functional (5.1a). The general form (5.1a) includes a *terminal cost* or *Mayer term* $V(\boldsymbol{x}(t_f), t_f)$ and an *integral cost* $l(\boldsymbol{x}, \boldsymbol{u}, t)$. Their combination leads to the following cases

$$J(\boldsymbol{u}) = \begin{cases} V(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} l(\boldsymbol{x}(t), \boldsymbol{u}(t), t)\,\mathrm{d}t & \text{Bolza form} \\ \int_{t_0}^{t_f} l(\boldsymbol{x}(t), \boldsymbol{u}(t), t)\,\mathrm{d}t & \text{Lagrange form} \\ V(\boldsymbol{x}(t_f), t_f) & \text{Mayer form}\,. \end{cases} \tag{5.3}$$

The Bolza and Lagrange forms can always be expressed in Mayer form by introducing the integral term as new state variable

$$\dot{x}_{n+1} = l(\boldsymbol{x}, \boldsymbol{u}, t)\,, \quad x_{n+1}(t_0) = 0 \tag{5.4}$$

and adding $x_{n+1}(t_f)$ to the terminal cost $V(\boldsymbol{x}(t_f), t_f)$. The most common examples of the integral cost are

$J(\boldsymbol{u}) = \frac{1}{2}\int_{t_0}^{t_f} \boldsymbol{x}^\mathsf{T}(t)\boldsymbol{Q}\boldsymbol{x}(t)\,\mathrm{d}t$	quadratic in the state ($\boldsymbol{Q}$ symmetric & $\boldsymbol{Q} > \boldsymbol{0}$)
$J(\boldsymbol{u}) = \frac{1}{2}\int_{t_0}^{t_f} \boldsymbol{u}^\mathsf{T}(t)\boldsymbol{R}\boldsymbol{u}(t)\,\mathrm{d}t$	energy optimal criterion ($\boldsymbol{R}$ symmetric & $\boldsymbol{R} > \boldsymbol{0}$)
$J(\boldsymbol{u}) = \frac{1}{2}\int_{t_0}^{t_f} \boldsymbol{x}^\mathsf{T}(t)\boldsymbol{Q}\boldsymbol{x}(t) + \boldsymbol{u}^\mathsf{T}(t)\boldsymbol{R}\boldsymbol{u}(t)\,\mathrm{d}t$	general quadratic form ($\boldsymbol{Q}, \boldsymbol{R}$ symmetric & $\boldsymbol{Q} \geq \boldsymbol{0}, \boldsymbol{R} > 0$)
$J(\boldsymbol{u}) = \int_{t_0}^{t_f} 1\,\mathrm{d}t = t_f - t_0$	time optimality
$J(\boldsymbol{u}) = \int_{t_0}^{t_f} 1 + \boldsymbol{u}^\mathsf{T}(t)\boldsymbol{R}\boldsymbol{u}(t)\,\mathrm{d}t$	combination of time/energy optimality.

The denomination *energy optimality* for a quadratic control term $\boldsymbol{u}^\mathsf{T}\boldsymbol{R}\boldsymbol{u}$ stems from interpreting the control as an electrical signal.

Example 5.1 *Several examples of optimal control problems were considered in Chapter 1. In case of the Goddard rocket (Example 1.4), the terminal cost $V(\boldsymbol{x}(t_f), t_f) = -h(t_f)$ maximizes the altitude. For the economic model in Example 1.5, $V(\boldsymbol{x}(t_f), t_f) = -K^\kappa(t_f)$ describes the capital of inheritance. The combination of time/energy optimality was considered for the inverted pendulum in Example 1.3.*

End time t_f

The end time or optimization horizon t_f is either pre-defined (fixed) or unspecified (free). In the free end time case, t_f is part of the solution of the problem (5.1). The end time t_f plays a major role in this regard and often decides upon whether a solution to an optimal control problem exists or not (also see Example 1.3).

Constraints (5.1d)

Constraints are present in many practical problems. In most cases, the general nonlinear form (5.1d) can be reduced to input constraints

$$\boldsymbol{u}(t) \in U \subseteq \mathbb{R}^m \quad \forall t \in [t_0, t_f] \tag{5.5}$$

and/or state constraints

$$\boldsymbol{x}(t) \in X \subseteq \mathbb{R}^n \quad \forall t \in [t_0, t_f]\,. \tag{5.6}$$

Examples of input constraints are voltage or current limits or a maximum force or torque of an actuator. This chapter focuses on the unconstrained and the input-constrained case (5.5). The rigorous mathematical treatment of state constraints (5.6) or general constraints of the form (5.1d) is significantly harder than the input-constrained case and is only briefly addressed in Section 5.5.5.

5.2 Calculus of variations

The consideration of dynamic optimization problems is more complex than in the parameter optimization case. An elegant framework for solving dynamic optimization problems is the *calculus of variations* that was developed at the end of the 17th and at the beginning of the 18th century. The most prominent names that have contributed to the calculus of variations are *Bernoulli*, *Euler*, and *Lagrange*.

For a general introduction into the calculus of variations, we consider a general functional to be minimized before the results are applied to unconstrained optimal control problems (Section 5.3).

5.2.1 Fundamentals

The previous parameter or finite-dimensional optimization was concerned with the minimization of a function $f(\boldsymbol{x})$ subject to optional constraints, where the optimization variables $\boldsymbol{x}$ are elements of the Euclidean space $\mathbb{R}^n$. Contrary to this, dynamic optimization minimizes a *functional*

$$J(\boldsymbol{x}^*) = \min_{\boldsymbol{x} \in \mathcal{X}} J(\boldsymbol{x}) \tag{5.7}$$

with $J : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X}$ is a suitable *function space*. An example is the functional

$$J(\boldsymbol{x}) = \int_{t_0}^{t_f} \phi(\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), t)\,\mathrm{d}t \tag{5.8}$$

with the optimal time function $\boldsymbol{x}^*(t)$, $t \in [t_0, t_f]$ that minimizes $J(\boldsymbol{x})$. The functional (5.8) is also known as *Lagrange problem* in the calculus of variations and will be investigated in Section 5.2.2.

In the spirit of dynamic optimization, the time function $\boldsymbol{x}(t)$ is also called *trajectory*. The set $\mathcal{X}_{ad}$ denotes the respective admissible set. For instance, if the minimization of the functional (5.7) is subject to the boundary conditions

$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0\,, \quad \boldsymbol{x}(t_f) = \boldsymbol{x}_f\,, \tag{5.9}$$

the admissible set is defined by

$$\mathcal{X}_{ad} = \{\boldsymbol{x} \in \mathcal{X} \,|\, \boldsymbol{x}(t_0) = \boldsymbol{x}_0\,, \boldsymbol{x}(t_f) = \boldsymbol{x}_f\}\,. \tag{5.10}$$

A point $\boldsymbol{x}^* = \boldsymbol{x}^*(t)$ (i.e. trajectory) is a global minimum point of a functional $J(\boldsymbol{x})$ if

$$J(\boldsymbol{x}^*) \leq J(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \mathcal{X}_{ad}\,. \tag{5.11}$$

The definition of a *local* minimum, however, is based on a suitable norm $\|\cdot\|$ of $\mathcal{X}$

$$\exists \alpha > 0: \quad J(\boldsymbol{x}^*) \leq J(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \{\boldsymbol{x} \in \mathcal{X}_{ad} \,|\, \|\boldsymbol{x} - \boldsymbol{x}^*\| \leq \alpha\}\,. \tag{5.12}$$

The calculus of variations often considers the class of continuous (piecewise) differentiable functions as function space with

$\mathcal{C}^k[t_0, t_f]$: Class of k-times continuously differentiable functions on $[t_0, t_f]$
$\bar{\mathcal{C}}^k[t_0, t_f]$: Class of k-times piecewise continuously differentiable functions on $[t_0, t_f]$
$\mathcal{C}_n^k[t_0, t_f]$: Class of functions $\boldsymbol{x} = [x_1, \ldots, x_n]^\mathsf{T}$ with $x_i \in \mathcal{C}^k[t_0, t_f]$, $i = 1, \ldots, n$
$\bar{\mathcal{C}}_n^k[t_0, t_f]$: Class of functions $\boldsymbol{x} = [x_1, \ldots, x_n]^\mathsf{T}$ with $x_i \in \bar{\mathcal{C}}^k[t_0, t_f]$, $i = 1, \ldots, n$.

A suitable function norm to equip $\mathcal{X}$ with is the supremum norm

$$|x|_\infty = |x(\cdot)|_\infty = \max_{t \in [t_0, t_f]} |x(t)| \quad \text{or} \quad \|\boldsymbol{x}\|_\infty = \|\boldsymbol{x}(\cdot)\|_\infty = \max_{t \in [t_0, t_f]} \|\boldsymbol{x}(t)\| \tag{5.13}$$

with an associated vector norm $\|\boldsymbol{x}(t)\|$ in $\mathbb{R}^n$.

The generalization of the gradient $\nabla f(\boldsymbol{x})$ of a function $f : \mathbb{R}^n \to \mathbb{R}$ to the function space setting is the *Gâteaux derivative*, also known as *first variation* of the functional $J(\boldsymbol{x})$.

Definition 5.1 (First variation / Gâteaux derivative) *The first variation or Gâteaux derivative of the functional $J : \mathcal{X} \to \mathbb{R}$ at point $\boldsymbol{x} \in \mathcal{X}$ in direction of $\boldsymbol{y} \in \mathcal{X}$ is defined by*

$$\delta J(\boldsymbol{x}; \boldsymbol{y}) = \lim_{\varepsilon \to 0} \frac{J(\boldsymbol{x} + \varepsilon \boldsymbol{y}) - J(\boldsymbol{x})}{\varepsilon} = \left.\frac{\mathrm{d} J(\boldsymbol{x} + \varepsilon \boldsymbol{y})}{\mathrm{d}\varepsilon}\right|_{\varepsilon = 0}. \tag{5.14}$$

If $\delta J(\boldsymbol{x}; \boldsymbol{y})$ exists for all $\boldsymbol{y} \in \mathcal{X}$, then J is Gâteaux differentiable at point $\boldsymbol{x} \in \mathcal{X}$.

Example 5.2 *For the functional $J(x) = \int_{t_0}^{t_f} x(t)^2 + x(t)\,\mathrm{d}t$ with $x \in \mathcal{C}^1[t_0, t_f]$, the Gâteaux derivative in direction $y \in \mathcal{C}^1[t_0, t_f]$ is*

$$\begin{aligned}
\delta J(x; y) &= \lim_{\varepsilon \to 0} \frac{J(x + \varepsilon y) - J(x)}{\varepsilon} \\
&= \lim_{\varepsilon \to 0} \left[\frac{1}{\varepsilon} \int_{t_0}^{t_f} (x(t) + \varepsilon y(t))^2 + x(t) + \varepsilon y(t) - x(t)^2 - x(t) \mathrm{d}t\right] \\
&= \lim_{\varepsilon \to 0} \left[\int_{t_0}^{t_f} 2x(t)y(t) + y(t) + \varepsilon y(t)^2\,\mathrm{d}t\right] \\
&= \int_{t_0}^{t_f} 2x(t)y(t) + y(t)\,\mathrm{d}t\,.
\end{aligned} \tag{5.15}$$

The same result is obtained by directly differentiating J w.r.t. ε

$$\begin{aligned}\delta J(x;y) &= \left.\frac{\mathrm{d}J(x+\varepsilon y)}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} \\ &= \left.\int_{t_0}^{t_f} \frac{\mathrm{d}}{\mathrm{d}\varepsilon}\left[(x(t)+\varepsilon y(t))^2 + x(t) + \varepsilon y(t)\right]\mathrm{d}t\right|_{\varepsilon=0} \\ &= \left.\int_{t_0}^{t_f} 2(x(t)+\varepsilon y(t))y(t) + y(t)\,\mathrm{d}t\right|_{\varepsilon=0} \\ &= \int_{t_0}^{t_f} 2x(t)y(t) + y(t)\,\mathrm{d}t\,. \end{aligned} \tag{5.16}$$

This shows that the functional J is Gâteaux differentiable in all points $y \in \mathcal{C}^1[t_0, t_f]$.

According to Definition (5.14), the existence of the Gâteaux derivative $\delta J(\boldsymbol{x};\boldsymbol{y})$ requires that the functionals $J(\boldsymbol{x})$ and $J(\boldsymbol{x}+\varepsilon\boldsymbol{y})$ are defined for all sufficiently small ε and that the derivative of $J(\boldsymbol{x}+\varepsilon\boldsymbol{y})$ w.r.t. ε must exist. The following example illustrates the necessity behind this.

Example 5.3 (Non-existence of Gâteaux derivative [6]) *The functional $J(x) = \int_0^1 |x(t)|\,\mathrm{d}t$ is defined and bounded for all $x \in \mathcal{C}^1[0,1]$. For $x(t) = 0$ and $y(t) = t$ we get*

$$\frac{J(x+\varepsilon y) - J(x)}{\varepsilon} = \frac{1}{\varepsilon}\int_0^1 |\varepsilon t|\,\mathrm{d}t = \operatorname{sign}(\varepsilon)\int_0^1 t\,\mathrm{d}t = \begin{cases} \frac{1}{2} & \text{for } \varepsilon > 0 \\ -\frac{1}{2} & \text{for } \varepsilon < 0\,. \end{cases} \tag{5.17}$$

Thus, the limit value for $\varepsilon \to 0$ is not unique and therefore the Gâteaux derivative at point $x(t) = 0$ in direction of $y(t) = t$ does not exist.

If the Gâteaux derivative of a functional $J(\boldsymbol{x})$ at a point $\boldsymbol{x} \in \mathcal{X}$ exists and $\delta J(\boldsymbol{x};\boldsymbol{y}) < 0$ holds, then $\boldsymbol{y} \in \mathcal{X}$ is a descent direction, i.e.

$$J(\boldsymbol{x} + \alpha\boldsymbol{y}) < J(\boldsymbol{x})$$

for a sufficiently small step size $\alpha > 0$. Hence, the condition $\delta J(\boldsymbol{x};\boldsymbol{y}) < 0$ is a generalization of the descent condition (3.18) in parameter optimization. A necessary condition for a minimum $J(\boldsymbol{x}^*)$ at point $\boldsymbol{x}^*$ is that no direction $\boldsymbol{y}$ is allowed to exist with $\delta J(\boldsymbol{x}^*;\boldsymbol{y}) < 0$. This leads to the concept of $\mathcal{X}_{ad}$–admissible directions and the necessary first-order optimality conditions.

Definition 5.2 ($\mathcal{X}_{ad}$–admissible direction) *Consider the functional $J : \mathcal{X} \to \mathbb{R}$ with the admissible set $\mathcal{X}_{ad} \subseteq \mathcal{X}$. A direction $\boldsymbol{y} \in \mathcal{X}$, $\boldsymbol{y} \neq \boldsymbol{0}$ is called $\mathcal{X}_{ad}$–admissible at point $\boldsymbol{x} \in \mathcal{X}$ if $\delta J(\boldsymbol{x};\boldsymbol{y})$ exists and $\boldsymbol{x} + \varepsilon\boldsymbol{y} \in \mathcal{X}_{ad}$ for sufficiently small values of $\varepsilon \in \mathbb{R}$.*

Theorem 5.1 (Necessary first-order optimality conditions) *Suppose that the functional $J : \mathcal{X} \to \mathbb{R}$ has a minimum at point $\boldsymbol{x}^* \in \mathcal{X}_{ad} \subseteq \mathcal{X}$. Then,*

$$\delta J(\boldsymbol{x}^*;\delta\boldsymbol{x}) = 0 \tag{5.18}$$

for all $\mathcal{X}_{ad}$–admissible directions $\delta\boldsymbol{x} \in \mathcal{X}$ at point $\boldsymbol{x}^*$.

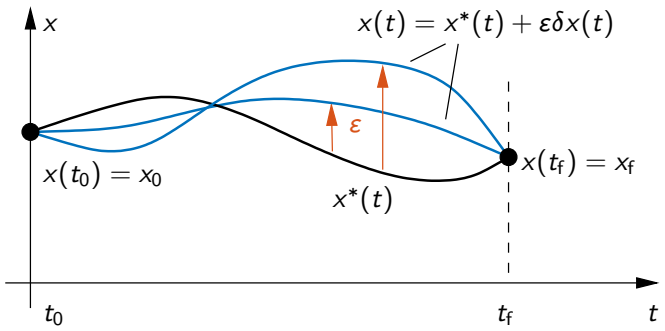


Figure 5.1: Illustration of $\mathcal{X}_{ad}$–admissible directions δx at $x^* \in \mathcal{X}_{ad}$.

The concept of $\mathcal{X}_{ad}$–admissible directions $\delta\boldsymbol{x}$ is relevant in connection with additional side constraints that are accounted for in the admissible set $\mathcal{X}_{ad}$. For instance, if boundary conditions (5.9) are given, also see (5.10), then $\mathcal{X}_{ad}$–admissible directions $\delta\boldsymbol{x}(t)$ must satisfy the homogeneous boundary conditions $\delta\boldsymbol{x}(t_0) = \boldsymbol{0}$ and $\delta\boldsymbol{x}(t_f) = \boldsymbol{0}$, such that $\boldsymbol{x} = \boldsymbol{x}^* + \varepsilon\delta\boldsymbol{x}$ as the argument in the first variation

$$\delta J(\boldsymbol{x}^*; \delta\boldsymbol{x}) = \left.\frac{\mathrm{d}J(\boldsymbol{x} + \varepsilon\boldsymbol{y})}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0}$$

satisfies $\boldsymbol{x} \in \mathcal{X}_{ad}$ (similar to $\boldsymbol{x}^* \in \mathcal{X}_{ad}$), see Figure 5.1. In this light, Theorem 5.1 states that no $\mathcal{X}_{ad}$–admissible direction $\delta\boldsymbol{x}(t)$ is allowed to exist that leads to a reduction of the cost functional $J(\boldsymbol{x})$.

In order to make the optimality condition (5.18) more explicit, we invoke the *fundamental lemma of the calculus of variations*. Its proof is omitted here but can e.g. be found in [2]. As it will turn out in the next section, the fundamental lemma is essential to derive the famous *Euler-Lagrange equations*.

Theorem 5.2 (Fundamental lemma of the calculus of variations) *Suppose that a continuous function $\boldsymbol{h} \in \mathcal{C}_n^0[t_0, t_f]$ satisfies*

$$\int_{t_0}^{t_f} \boldsymbol{h}(t)^\mathsf{T} \delta\boldsymbol{x}(t)\,\mathrm{d}t = 0 \tag{5.19}$$

for all continuous functions $\delta\boldsymbol{x} \in \mathcal{C}_n^0[t_0, t_f]$. Then, $\boldsymbol{h}(t) = \boldsymbol{0}$ for all $t \in [t_0, t_f]$.

5.2.2 Euler-Lagrange equations

We now focus on the Lagrange problem (5.8) with the boundary conditions (5.9) and repeat it for the sake of completeness

$$J(\boldsymbol{x}) = \int_{t_0}^{t_f} \phi(\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), t)\,\mathrm{d}t \tag{5.20a}$$

with

$$\boldsymbol{x} \in \mathcal{X}_{ad} = \{\boldsymbol{x} \in \mathcal{X} \,|\, \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \boldsymbol{x}(t_f) = \boldsymbol{x}_f\}\,. \tag{5.20b}$$

The end time t_f is supposed to be fixed and we assume that an optimal solution $\boldsymbol{x}^* \in \mathcal{X}_{ad} \subset \mathcal{X} = \mathcal{C}_n^1[t_0, t_f]$ exists that minimizes the functional (5.20a). The necessary condition

(5.18) requires that the Gâteaux derivative (or first variation) vanishes for all $\mathcal{X}_{ad}$–admissible variations $\delta\boldsymbol{x} \in \mathcal{C}_n^1[t_0, t_f]$ with the homogeneous boundary conditions

$$\delta\boldsymbol{x}(t_0) = \boldsymbol{0}\,, \quad \delta\boldsymbol{x}(t_f) = \boldsymbol{0}\,. \tag{5.21}$$

The Gâteaux derivative of J at the optimal point $\boldsymbol{x}^*$ in direction of $\delta\boldsymbol{x}$ is computed by differentiation

$$\begin{aligned}
\delta J(\boldsymbol{x}^*; \delta\boldsymbol{x}) &= \left.\frac{\mathrm{d}J(\boldsymbol{x}^* + \varepsilon\delta\boldsymbol{x})}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} \\
&= \int_{t_0}^{t_f} \left.\frac{\mathrm{d}}{\mathrm{d}\varepsilon}\phi(\boldsymbol{x}^*(t) + \varepsilon\delta\boldsymbol{x}(t), \dot{\boldsymbol{x}}^*(t) + \varepsilon\delta\dot{\boldsymbol{x}}(t), t)\,\mathrm{d}t\right|_{\varepsilon=0} \\
&= \int_{t_0}^{t_f} \phi_{\boldsymbol{x}}(\boldsymbol{x}^*(t), \dot{\boldsymbol{x}}^*(t))^\mathsf{T}\delta\boldsymbol{x}(t) + \phi_{\dot{\boldsymbol{x}}}(\boldsymbol{x}^*(t), \dot{\boldsymbol{x}}^*(t))^\mathsf{T}\delta\dot{\boldsymbol{x}}(t)\,\mathrm{d}t\,.
\end{aligned} \tag{5.22}$$

The functions $\phi_{\boldsymbol{x}}$ and $\phi_{\dot{\boldsymbol{x}}}$ denote the (transposed) partial derivatives $\phi_{\boldsymbol{x}} = (\frac{\partial\phi}{\partial\boldsymbol{x}})^\mathsf{T}$ and $\phi_{\dot{\boldsymbol{x}}} = (\frac{\partial\phi}{\partial\dot{\boldsymbol{x}}})^\mathsf{T}$. The integrand of (5.22) is continuous because the function ϕ is continuously differentiable by assumption and $\delta\boldsymbol{x} \in \mathcal{C}_n^1[t_0, t_f]$. This implies that $J(\boldsymbol{x})$ is Gâteaux differentiable at all points $\boldsymbol{x} \in \mathcal{X} = \mathcal{C}_n^1[t_0, t_f]$.

Partial integration of the second term in (5.22) leads to [2] (neglecting arguments for readability)

$$\int_{t_0}^{t_f} \phi_{\dot{\boldsymbol{x}}}(\boldsymbol{x}^*, \dot{\boldsymbol{x}}^*, t)^\mathsf{T}\delta\dot{\boldsymbol{x}}\,\mathrm{d}t = \left[\phi_{\dot{\boldsymbol{x}}}(\boldsymbol{x}^*, \dot{\boldsymbol{x}}^*, t)^\mathsf{T}\delta\boldsymbol{x}\right]_{t_0}^{t_f} - \int_{t_0}^{t_f} \frac{\mathrm{d}}{\mathrm{d}t}\phi_{\dot{\boldsymbol{x}}}(\boldsymbol{x}^*, \dot{\boldsymbol{x}}^*, t)^\mathsf{T}\delta\boldsymbol{x}\,\mathrm{d}t\,. \tag{5.23}$$

The boundary terms vanish due to the fact that only $\mathcal{X}_{ad}$–admissible variations with $\delta\boldsymbol{x}(t_0) = \delta\boldsymbol{x}(t_f) = \boldsymbol{0}$ are considered. Hence, the Gâteaux derivative (5.22) simplifies to

$$\delta J(\boldsymbol{x}^*; \delta\boldsymbol{x}) = \int_{t_0}^{t_f} \left[\phi_{\boldsymbol{x}}(\boldsymbol{x}^*, \dot{\boldsymbol{x}}^*, t) - \frac{\mathrm{d}}{\mathrm{d}t}\phi_{\dot{\boldsymbol{x}}}(\boldsymbol{x}^*, \dot{\boldsymbol{x}}^*, t)\right]^\mathsf{T}\delta\boldsymbol{x}\,\mathrm{d}t \tag{5.24}$$

in dependence of $\delta\boldsymbol{x}(t)$. For an optimal trajectory $\boldsymbol{x}^*(t)$, the Gâteaux derivative $\delta J(\boldsymbol{x}^*; \delta\boldsymbol{x})$ must vanish for all $\mathcal{X}_{ad}$–admissible variations $\delta\boldsymbol{x}(t)$, see Theorem 5.1. Thus, the fundamental lemma of the calculus of variations (Theorem 5.2) yields

$$\boxed{\phi_{\boldsymbol{x}}(\boldsymbol{x}^*(t), \dot{\boldsymbol{x}}^*(t), t)) - \frac{\mathrm{d}}{\mathrm{d}t}\phi_{\dot{\boldsymbol{x}}}(\boldsymbol{x}^*(t), \dot{\boldsymbol{x}}^*(t), t) = \boldsymbol{0}\,.} \tag{5.25}$$

The differential equations (5.25) are known as *Euler-Lagrange equations*. They are a *necessary condition* for an optimal solution $\boldsymbol{x}^*(t)$ to minimize the functional (5.20a) subject to the boundary conditions in (5.20b).

With the *second variation* of J

$$\delta^2 J = \left.\frac{\mathrm{d}^2 J(\boldsymbol{x}^* + \varepsilon\delta\boldsymbol{x})}{\mathrm{d}\varepsilon^2}\right|_{\varepsilon=0} \geq 0\,, \tag{5.26}$$

it can be shown that the Hessian matrix of $\phi(\boldsymbol{x}, \dot{\boldsymbol{x}})$ w.r.t. $\dot{\boldsymbol{x}}$ must be positive semidefinite, i.e.

$$\phi_{\dot{\boldsymbol{x}}\dot{\boldsymbol{x}}}(\boldsymbol{x}^*(t), \dot{\boldsymbol{x}}^*(t), t) \geq 0 \quad \forall t \in [t_0, t_f]\,. \tag{5.27}$$

This so-called *Legendre condition* is a necessary second-order optimality condition for an optimal solution $\boldsymbol{x}^*(t)$, $t \in [t_0, t_f]$ in addition to the Euler-Lagrange equations (5.25). The derivation of (5.27) is similar to the Gâteaux derivative or first variation (5.14) by means of partial integration and is not further detailed here.

[2] Remember that $\int \boldsymbol{u}^\mathsf{T}\dot{\boldsymbol{v}}\,\mathrm{d}t = \boldsymbol{u}^\mathsf{T}\boldsymbol{v} - \int \dot{\boldsymbol{u}}^\mathsf{T}\boldsymbol{v}\,\mathrm{d}t$.

There are two special cases of the Euler-Lagrange equations (5.25) that are worth mentioning:

- If ϕ does not explicitly depend on $\boldsymbol{x}$, i.e. $\phi = \phi(\dot{\boldsymbol{x}}, t)$, a first integration of (5.25) shows that
$$\phi_{\dot{x}}(\dot{\boldsymbol{x}}^*(t), t) = const. \tag{5.28}$$
- If ϕ does not explicitly depend on t, i.e. $\phi = \phi(\boldsymbol{x}, \dot{\boldsymbol{x}})$, it can be shown that the Euler-Lagrange equations (5.25) simplify to
$$\phi(\boldsymbol{x}^*(t), \dot{\boldsymbol{x}}^*(t)) - \dot{\boldsymbol{x}}^*(t)^\mathsf{T} \phi_{\dot{x}}(\boldsymbol{x}^*(t), \dot{\boldsymbol{x}}^*(t)) = const. \tag{5.29}$$

Exercise 5.1 *Compute the minimum distance between two points A and B in the (x, y)-plane. To this end, determine the cost functional to be minimized in dependence of the arc length s and use the relation*
$$\mathrm{d}s = \sqrt{\mathrm{d}x^2 + \mathrm{d}y^2} = \sqrt{1 + y'(x)^2}\,\mathrm{d}x. \tag{5.30}$$

5.2.3 Example: Galileo's hanging chain

In 1683, Galileo formulated the question how the form of a hanging chain that is fixed at two points can be mathematically described. Galileo (wrongly) postulated that the form is described by a parabola. The problem was solved in 1690 by *Jakob and Johannes Bernoulli* and *Leibnitz* using the calculus of variations.

We consider a chain of length L with both ends being fixed at two points A and B, see Figure 5.2. Hamilton's principle in mechanics states that the potential energy of the chain in stationary conditions must be minimal. The potential energy can be formulated with the arc length s and the relation (5.30)
$$V = \rho g \int_0^L y\,\mathrm{d}s = \rho g \int_{x_A}^{x_B} y\sqrt{1 + y'(x)^2}\,\mathrm{d}x\,, \tag{5.31}$$
where g denotes the acceleration due to gravity and ρ is the mass per chain length. Since the minimum of the potential energy V is sought, the constant factor ρg can be discarded, leading to the cost functional along with the boundary conditions
$$J(y) = \int_{x_A}^{x_B} y\sqrt{1 + y'(x)^2}\,\mathrm{d}x\,, \quad y(x_A) = y_A\,, \quad y(x_B) = y_B\,. \tag{5.32}$$
Due to the fact that the spatial variable x does not explicitly appear in $\phi(y, y') = y\sqrt{1 + y'(x)^2}$, we can refer to the special case (5.29) of the Euler-Lagrange equations (again omitting arguments)
$$\begin{aligned} \phi(y, y') - y'\phi_{y'}(y, y') &= y\sqrt{1 + y'^2} - \frac{y\,y'^2}{\sqrt{1 + y'^2}} \\ &= \frac{y}{\sqrt{1 + y'^2}} = c_1 = const. \end{aligned} \tag{5.33}$$
This equation can be written as
$$y' = \sqrt{\frac{y^2}{c_1^2} - 1} \tag{5.34}$$

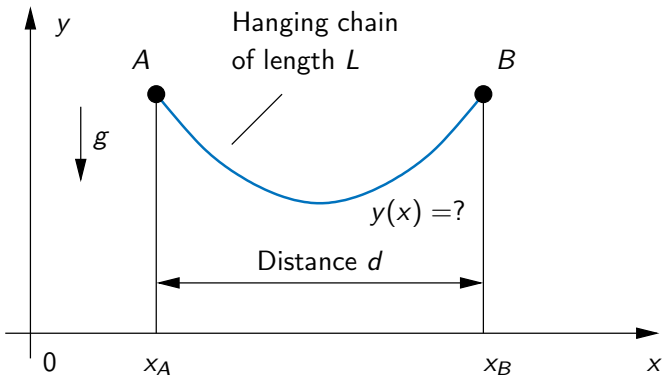


Figure 5.2: Galileo's problem: Determination of the form $y(x)$ of a hanging chain.

if we assume that $c_1 \neq 0$. With $y' = \frac{\mathrm{d}y}{\mathrm{d}x}$ and using integration, the spatial coordinate x can be expressed as function of y

$$x = \int \frac{1}{\sqrt{y^2/c_1^2 - 1}} \,\mathrm{d}y = c_1 \ln\left(\frac{y + \sqrt{y^2 - c_1^2}}{c_1}\right) + c_2$$

with the integration constant c_2. By transforming and squaring this equation

$$\left(c_1 e^{(x-c_2)/c_1} - y\right)^2 = y^2 - c_1^2 \,,$$

we can express y as

$$y = \frac{1}{2} c_1 \frac{e^{2(x-c_2)/c_1} + 1}{e^{(x-c_2)/c_1}} = \frac{1}{2} c_1 \left(e^{(x-c_2)/c_1} + e^{-(x-c_2)/c_1}\right).$$

With $\cosh b = \frac{1}{2}(e^b + e^{-b})$, we eventually obtain the famous *catenary* of the hanging chain

$$y(x) = c_1 \cosh\left(\frac{x - c_2}{c_1}\right) . \tag{5.35}$$

Optionally, a third constant c_3 can be added to (5.35) in order to compute the catenary for a chain of length L hanging between the points A and B by (numerically) solving the equations

$$y(x_A) = y_A \,, \quad y(x_B) = y_B \,, \quad \int_{x_A}^{x_B} \sqrt{1 + y'(x)^2} \,\mathrm{d}x = L \,. \tag{5.36}$$

Figure 5.3 shows the catenary $y(x)$ with length $L = 1.5$ for different hang points compared to Galileo's postulated parabola.

5.3 Unconstrained optimal control problems

In the next step, the calculus of variations is applied to unconstrained optimal control problems

$$\begin{aligned}
&\min_{\boldsymbol{u}(\cdot)} \quad J(\boldsymbol{u}) = V(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} l(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \,\mathrm{d}t && \text{(5.37a)}\\
&\text{s.t.} \quad \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t), \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0 && \text{(5.37b)}\\
&\qquad x_i(t_f) = x_{\mathrm{f},i} \,, \quad i \in \mathcal{I}_f && \text{(5.37c)}
\end{aligned}$$

to derive necessary conditions for an optimal solution.

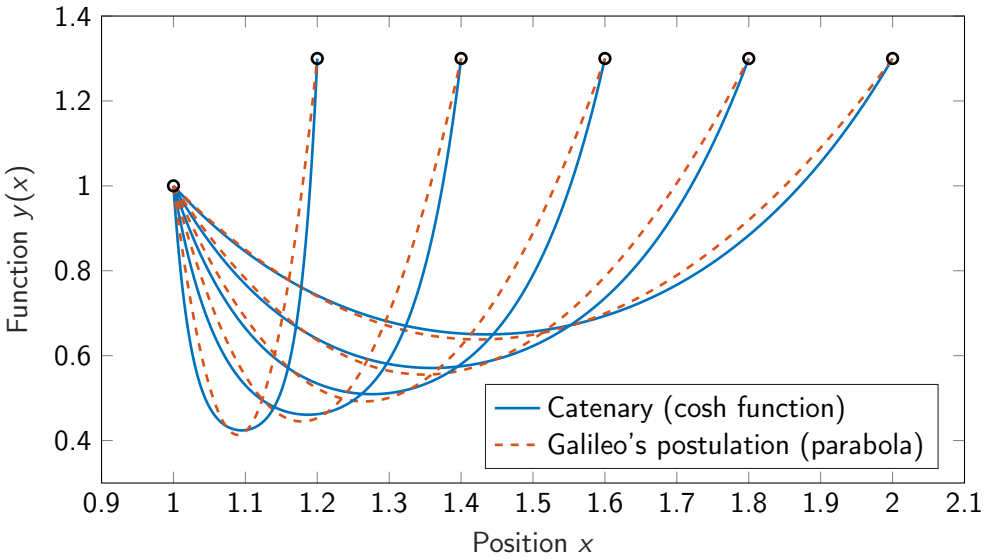


Figure 5.3: Hanging chain of length $L = 1.5$ for different hang points.

5.3.1 Application of the calculus of variations

The considerations for the Euler-Lagrange equations are successively extended to account for the system dynamics (5.37b) and the partial terminal conditions (5.37c). The end time t_f is supposed to be fixed, i.e. $V = V(\boldsymbol{x}(t_f))$. The free end time case will be considered afterwards.

Similar to the constrained optimization case in Chapter 4, the system dynamics (5.37b) are treated as (dynamic) equality constraints that are added to the cost functional via Lagrange multipliers

$$\bar{J}(\boldsymbol{u}, \boldsymbol{x}) = V(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} l(\boldsymbol{x}(t), \boldsymbol{u}(t), t) + \boldsymbol{\lambda}^\mathsf{T}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t) - \dot{\boldsymbol{x}}) \, \mathrm{d}t \tag{5.38}$$

with the optimization variables $\boldsymbol{u} \in \mathcal{U} = \mathcal{C}_m^0[t_0, t_f]$ and $\boldsymbol{x} \in \mathcal{X} = \mathcal{C}_n^1[t_0, t_f]$. The boundary conditions of (5.37) are incorporated in the admissible set

$$\mathcal{X}_{ad} = \{\boldsymbol{x} \in \mathcal{X} \,|\, \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \; x_i(t_f) = x_{f,i}, \; i \in \mathcal{I}_f\} \, . \tag{5.39}$$

Note that the multipliers $\boldsymbol{\lambda}(t), t \in [t_0, t_f]$ are functions of time in contrast to the finite-dimensional (parameter optimization) case, where the multipliers $\boldsymbol{\lambda}$ for equality constraints are constant vectors.

Following the procedure of the Euler-Lagrange equations, we compute the Gâteaux derivative of $\bar{J}$ at an optimal point $\boldsymbol{u}^* \in \mathcal{U}$ and $\boldsymbol{x}^* \in \mathcal{X}_{ad}$ in direction of $\delta\boldsymbol{u}$ and $\delta\boldsymbol{x}$, i.e. (without time arguments)

$$\begin{aligned}
\delta\bar{J}(\boldsymbol{u}^*, \boldsymbol{x}^*; \delta\boldsymbol{u}, \delta\boldsymbol{x}) &= \left.\frac{\mathrm{d}\bar{J}(\boldsymbol{u}^* + \varepsilon\delta\boldsymbol{u}, \boldsymbol{x}^* + \varepsilon\delta\boldsymbol{x})}{\mathrm{d}\varepsilon}\right|_{\varepsilon=0} \\
&= V_{\boldsymbol{x}}(\boldsymbol{x}^*(t_f), t_f)^\mathsf{T}\delta\boldsymbol{x}(t_f) + \int_{t_0}^{t_f} \left[l_{\boldsymbol{x}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{f}_{\boldsymbol{x}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T}\boldsymbol{\lambda}\right]^\mathsf{T}\delta\boldsymbol{x} \\
&\quad + \left[l_{\boldsymbol{u}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{f}_{\boldsymbol{u}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T}\boldsymbol{\lambda}\right]^\mathsf{T}\delta\boldsymbol{u} - \boldsymbol{\lambda}^\mathsf{T}\delta\dot{\boldsymbol{x}} \, \mathrm{d}t \, .
\end{aligned} \tag{5.40}$$

Partial integration of the latter integral term gives

$$\delta\bar{J}(\boldsymbol{u}^*, \boldsymbol{x}^*; \delta\boldsymbol{u}, \delta\boldsymbol{x}) = V_{\boldsymbol{x}}(\boldsymbol{x}^*(t_f), t_f)^\mathsf{T}\delta\boldsymbol{x}(t_f) - \Big[\boldsymbol{\lambda}^\mathsf{T}\delta\boldsymbol{x}\Big]_{t_0}^{t_f}$$
$$+ \int_{t_0}^{t_f} \Big[l_{\boldsymbol{x}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{f}_{\boldsymbol{x}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T}\boldsymbol{\lambda} + \dot{\boldsymbol{\lambda}}\Big]^\mathsf{T}\delta\boldsymbol{x} + \Big[l_{\boldsymbol{u}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{f}_{\boldsymbol{u}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T}\boldsymbol{\lambda}\Big]^\mathsf{T}\delta\boldsymbol{u}\,\mathrm{d}t\,.$$

The boundary terms can be simplified according to

$$\Big[\boldsymbol{\lambda}^\mathsf{T}\delta\boldsymbol{x}\Big]_{t_0}^{t_f} = \boldsymbol{\lambda}(t_f)^\mathsf{T}\delta\boldsymbol{x}(t_f) - \boldsymbol{\lambda}(t_0)^\mathsf{T}\delta\boldsymbol{x}(t_0) = \boldsymbol{\lambda}(t_f)^\mathsf{T}\delta\boldsymbol{x}(t_f)$$

because $\mathcal{X}_{ad}$–admissible directions must satisfy $\delta\boldsymbol{x}(t_0) = \boldsymbol{0}$.

A necessary optimality condition is $\delta\bar{J}(\boldsymbol{u}^*, \boldsymbol{x}^*; \delta\boldsymbol{u}, \delta\boldsymbol{x}) = 0$. The fundamental lemma of the calculus of variations implies that the following equations must be satisfied

$$\dot{\boldsymbol{\lambda}}^* = -l_{\boldsymbol{x}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) - \boldsymbol{f}_{\boldsymbol{x}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T}\boldsymbol{\lambda}^*\,, \qquad t \in (t_0, t_f) \tag{5.41a}$$

$$\boldsymbol{0} = l_{\boldsymbol{u}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{f}_{\boldsymbol{u}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T}\boldsymbol{\lambda}^*\,, \qquad t \in (t_0, t_f) \tag{5.41b}$$

$$0 = \Big(V_{\boldsymbol{x}}(\boldsymbol{x}^*(t_f)) - \boldsymbol{\lambda}^*(t_f)\Big)^\mathsf{T}\delta\boldsymbol{x}(t_f) = \sum_{i=1}^{n}\Big(V_{x_i}(\boldsymbol{x}^*(t_f)) - \lambda_i^*(t_f)\Big)\delta x_i(t_f)\,. \tag{5.41c}$$

In particular, a Lagrange multiplier $\boldsymbol{\lambda}^*(t)$, $t \in [t_0, t_f]$ corresponding to $\boldsymbol{x}^*(t)$, $\boldsymbol{u}^*(t)$ must satisfy the differential equation (5.41a). In addition, (5.41b) is an algebraic equation that must be satisfied over the whole time interval.

The terminal condition (5.41c) must hold for all $\mathcal{X}_{ad}$–admissible variations $\delta\boldsymbol{x}(t_f)$. They can be further simplified in light of the partial terminal conditions (5.37c). For the states x_i, $i \in \mathcal{I}_f$ that are fixed at the end time $t = t_f$, see (5.37c), an admissible variation must satisfy $\delta x_i(t_f) = 0$, whereas for all free terminal states $x_i(t_f)$, $i \notin \mathcal{I}_f$ the corresponding variation at the end time is arbitrary. Hence, if

$$\lambda_i^*(t_f) = V_{x_i}(\boldsymbol{x}^*(t_f))\,, \quad i \notin \mathcal{I}_f\,, \tag{5.42}$$

holds, then (5.41c) is satisfied for all admissible variations at $t = t_f$.

Consideration of free end time

If the end time t_f of the optimal control problem (5.1) is unspecified, we additionally have to consider the variation around the optimal end time t_f^*. The Gâteaux derivative at the optimal point $(\boldsymbol{u}^*, \boldsymbol{x}^*, t_f^*)$ in direction of $(\delta\boldsymbol{u}, \delta\boldsymbol{x}, \delta t_f)$ is an extention of (5.40) and reads (again omitting time arguments)

$$\delta\bar{J}(\boldsymbol{u}^*, \boldsymbol{x}^*, t_f^*; \delta\boldsymbol{u}, \delta\boldsymbol{x}, \delta t_f) = \frac{\mathrm{d}\bar{J}(\boldsymbol{u}^* + \varepsilon\delta\boldsymbol{u}, \boldsymbol{x}^* + \varepsilon\delta\boldsymbol{x}, t_f^* + \varepsilon\delta t_f)}{\mathrm{d}\varepsilon}\bigg|_{\varepsilon=0}$$
$$= V_{\boldsymbol{x}}(\boldsymbol{x}^*(t_f^*), t_f^*)^\mathsf{T}\Big(\overbrace{\delta\boldsymbol{x}(t_f^*) + \dot{\boldsymbol{x}}^*(t_f^*)\,\delta t_f}^{=\Delta\boldsymbol{x}_\mathrm{f}}\Big) + V_t(\boldsymbol{x}^*(t_f^*), t_f^*)\,\delta t_f$$
$$+ \Big(l(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{\lambda}^\mathsf{T}\big(\boldsymbol{f}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) - \dot{\boldsymbol{x}}\big)\Big)_{t=t_f^*}\delta t_f$$
$$+ \int_{t_0}^{t_f^*}\Big[l_{\boldsymbol{x}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{f}_{\boldsymbol{x}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T}\boldsymbol{\lambda}\Big]^\mathsf{T}\delta\boldsymbol{x}$$
$$+ \Big[l_{\boldsymbol{u}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{f}_{\boldsymbol{u}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T}\boldsymbol{\lambda}\Big]^\mathsf{T}\delta\boldsymbol{u} - \boldsymbol{\lambda}^\mathsf{T}\delta\dot{\boldsymbol{x}}\;\mathrm{d}t\,. \tag{5.43}$$

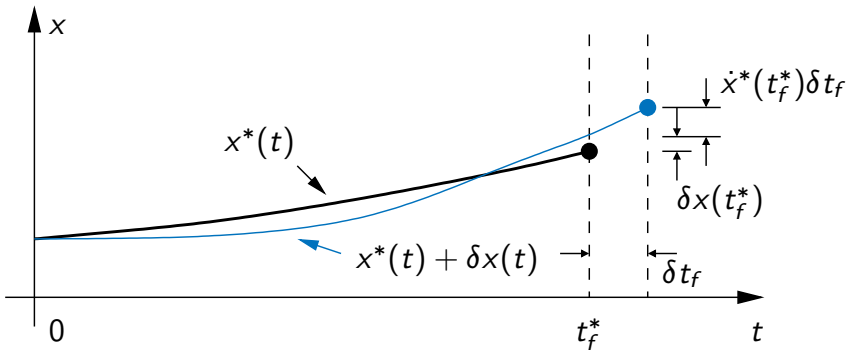


Figure 5.4: Total variation $\Delta \boldsymbol{x}_f = \delta \boldsymbol{x}(t_f^*) + \dot{\boldsymbol{x}}^*(t_f^*)\delta t_f$ of the terminal state for $x \in \mathbb{R}$.

The *total variation* $\Delta \boldsymbol{x}_f$ accounts for the additional influence of the end time variation δt_f on the change of the terminal state, see Figure 5.4. The variation of the integral term due to a small variation of the end time δt_f results in the term $(l(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{\lambda}^\mathsf{T}(\boldsymbol{f}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) - \dot{\boldsymbol{x}}))_{t=t_f^*}\delta t_f$ in (5.43). Partial integration of (5.43) with $\boldsymbol{\lambda}(t_0)^\mathsf{T}\delta \boldsymbol{x}(t_0) = 0$ yields

$$
\begin{aligned}
\delta \bar{J}(\boldsymbol{u}^*, \boldsymbol{x}^*, t_f^*; \delta \boldsymbol{u}, \delta \boldsymbol{x}, \delta t_f) = V_x(\boldsymbol{x}^*(t_f^*), t_f^*)^\mathsf{T} \Delta \boldsymbol{x}_f - \boldsymbol{\lambda}(t_f^*)^\mathsf{T} \Big(\overbrace{\delta \boldsymbol{x}(t_f^*) + \dot{\boldsymbol{x}}(t_f^*)\, \delta t_f}^{= \Delta \boldsymbol{x}_\mathrm{f}} \Big) \\
+ \Big(V_t(\boldsymbol{x}^*, t) + l(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{\lambda}^\mathsf{T} \boldsymbol{f}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) \Big)_{t=t_f^*} \delta t_f \\
+ \int_{t_0}^{t_f^*} \Big[l_x(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{f}_x(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T} \boldsymbol{\lambda} + \dot{\boldsymbol{\lambda}} \Big]^\mathsf{T} \delta \boldsymbol{x} \\
+ \Big[l_u(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{f}_u(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T} \boldsymbol{\lambda} \Big]^\mathsf{T} \delta \boldsymbol{u} \, \mathrm{d}t \, .
\end{aligned}
$$

If $\boldsymbol{x}^*(t)$, $\boldsymbol{u}^*(t)$, t_f^* is an optimal solution, then $\delta \bar{J} = 0$ must hold for all admissible variations $\delta \boldsymbol{x}(t)$, $\delta \boldsymbol{u}(t)$, δt_f. The fundamental lemma of the calculus of variations thus yields

$$
\begin{aligned}
\dot{\boldsymbol{\lambda}}^* &= -l_x(\boldsymbol{x}^*, \boldsymbol{u}^*, t) - \boldsymbol{f}_x(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T} \boldsymbol{\lambda}^* \, , \quad t \in (t_0, t_f) && \text{(5.44a)} \\
\boldsymbol{0} &= \quad l_u(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{f}_u(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T} \boldsymbol{\lambda}^* \, , \quad t \in (t_0, t_f) && \text{(5.44b)} \\
0 &= \Big(V_x(\boldsymbol{x}^*(t_f^*), t_f^*) - \boldsymbol{\lambda}^*(t_f) \Big)^\mathsf{T} \Delta \boldsymbol{x}_f && \text{(5.44c)} \\
0 &= \Big(V_t(\boldsymbol{x}^*, t) + l(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{\lambda}^\mathsf{T} \boldsymbol{f}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) \Big)_{t=t_f^*} . && \text{(5.44d)}
\end{aligned}
$$

In analogy to the previous fixed end time case, the condition (5.44c) is satisfied for (5.42). Moreover, the free end time t_f leads to the additional (scalar) terminal condition (5.44d).

5.3.2 Optimality conditions

The relations of the last section are necessary conditions for a solution of the optimal control problem (5.37). These conditions can be formalized by introducing the *Hamiltonian function* or in short *Hamiltonian*.

Definition 5.3 (Hamiltonian function) *The Hamiltonian (function) for the optimal control problem (5.37) is defined by*

$$H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, t) = l(\boldsymbol{x}, \boldsymbol{u}, t) + \boldsymbol{\lambda}^\mathsf{T} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t) \tag{5.45}$$

with the Lagrange multipliers $\boldsymbol{\lambda}(t) \in \mathbb{R}^n$.

The Hamiltonian allows to express the necessary first-order optimality conditions for (5.37) in a compact form as follows.

Theorem 5.3 (Necessary first-order optimality conditions) *Suppose that the functions l and $\boldsymbol{f}$ of the optimal control problem (5.37) are continuously differentiable in $(\boldsymbol{x}, \boldsymbol{u})$ and continuous in t. Suppose that $\boldsymbol{u}^* \in \mathcal{C}_m^0[t_0, t_f]$ is a (local) optimal solution of (5.37) with associated state trajectory $\boldsymbol{x}^* \in \mathcal{C}_n^1[t_0, t_f]$. Then, there exist time functions $\boldsymbol{\lambda}^* \in \mathcal{C}_n^1[t_0, t_f]$ with $\boldsymbol{\lambda}^* = [\lambda_1^*, \dots, \lambda_n^*]^\mathsf{T}$ such that the following conditions are satisfied:*

a) $(\boldsymbol{u}^*, \boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ *satisfy*

$$\dot{\boldsymbol{x}}^* = \quad H_{\boldsymbol{\lambda}}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*, t) = \boldsymbol{f}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) \tag{5.46a}$$

$$\dot{\boldsymbol{\lambda}}^* = -H_{\boldsymbol{x}}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*, t) = -l_{\boldsymbol{x}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) - \boldsymbol{f}_{\boldsymbol{x}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T} \boldsymbol{\lambda}^* \tag{5.46b}$$

$$\boldsymbol{0} = \quad H_{\boldsymbol{u}}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*, t) = l_{\boldsymbol{u}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t) + \boldsymbol{f}_{\boldsymbol{u}}(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T} \boldsymbol{\lambda}^* \tag{5.46c}$$

for $t \in [t_0, t_f]$ with the boundary conditions

$$\boldsymbol{x}^*(t_0) = \boldsymbol{x}_0 \tag{5.47a}$$

$$x_i^*(t_f) = x_{f,i}\,, \quad i \in \mathcal{I}_f \tag{5.47b}$$

and the transversality condition

$$\lambda_i^*(t_f) = V_{x_i}(\boldsymbol{x}(t_f), t_f)\,, \quad i \notin \mathcal{I}_f\,. \tag{5.47c}$$

b) If the end time t_f is unspecified and t_f^* is (locally) optimal, then (5.46) is satisfied for $t \in [t_0, t_f^*]$ and (5.47b), (5.47c) for $t_f = t_f^*$. In addition, the Hamiltonian satisfies the transversality condition

$$H(\boldsymbol{x}^*(t_f^*), \boldsymbol{u}^*(t_f^*), \boldsymbol{\lambda}(t_f^*), t_f^*) = -V_t(\boldsymbol{x}^*(t_f^*), t_f^*)\,. \tag{5.48}$$

The differential equations (5.46a), (5.46b) together are known as *canonical equations* and (5.46c) is sometimes called *stationarity condition*. The dynamics (5.46b) are called the *adjoint system* with the *adjoint state* $\boldsymbol{\lambda}$. The optimality conditions define a *two-point boundary value problem* (BVP) for the optimal states $\boldsymbol{x}^*(t)$ and $\boldsymbol{\lambda}^*(t)$ as well as for the optimal control $\boldsymbol{u}^*(t)$.

If t_f is fixed, the BVP (5.46), (5.47) has $2n$ dynamical equations with $2n$ boundary conditions and m algebraic equations for the $2n + m$ unknowns $\boldsymbol{x}^*(t)$, $\boldsymbol{\lambda}^*(t)$, and $\boldsymbol{u}^*(t)$. If t_f is free, the transversality condition (5.48) has to be satisfied as well, i.e. one scalar equation is added for one additional unknown variable t_f^*. Thus, the optimality conditions are well defined in terms of dimensions. The terminal conditions in (5.47) include the following special cases:

- **Fixed terminal state** $\boldsymbol{x}(t_f) = \boldsymbol{x}_f$, i.e. $\mathcal{I}_f = \{1, \dots, n\}$: The full state $\boldsymbol{x}$ is fixed at the initial time t_0 and final time t_f, the adjoint terminal state $\boldsymbol{\lambda}(t_f)$ is free.
- **Free terminal state** $\boldsymbol{x}(t_f)$, i.e. $\mathcal{I}_f = \{\}$: In this case, the full adjoint state is fixed at $t = t_f$, i.e. $\boldsymbol{\lambda}(t_f) = V_{\boldsymbol{x}}(\boldsymbol{x}(t_f), t_f)$. Thus, the boundary conditions separate into initial conditions for $\boldsymbol{x}(t_0)$ and terminal conditions for $\boldsymbol{\lambda}(t_f)$.
- **Free end time** t_f with mayer term $V = 0$ or $V = V(\boldsymbol{x}(t_f))$: In this case, the transversality condition (5.48) is homogeneous, i.e. $H|_{t=t_f^*} = 0$.

The extension of the second-order Legendre condition (5.27) to the optimal control problem (5.1) requires the Hessian of the Hamiltonian w.r.t. the control $\boldsymbol{u}$ to be positive semidefinite:

$$H_{\boldsymbol{uu}}(\boldsymbol{x}^*(t), \boldsymbol{u}^*(t), \boldsymbol{\lambda}^*(t), t) \geq 0 \quad \forall t \in [t_0, t_f]\,. \tag{5.49}$$

The function $H_{\boldsymbol{uu}}$ thereby denotes the second partial derivative $\frac{\partial^2 H}{\partial \boldsymbol{u}^2}$ of the Hamiltonian.

An interesting property of the Hamiltonian (5.45) is its behavior along an optimal solution:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t} H(\boldsymbol{x}^*(t), \boldsymbol{u}^*(t), \boldsymbol{\lambda}^*(t), t)) &= H_t^* + \overbrace{(H_{\boldsymbol{x}}^*)^\mathsf{T} \boldsymbol{f}^*}^{(H_{\boldsymbol{x}}^*)^\mathsf{T} \boldsymbol{f}^*} + \overbrace{(H_{\boldsymbol{u}}^*)^\mathsf{T} \dot{\boldsymbol{u}}^*}^{\mathbf{0}} + \overbrace{(H_{\boldsymbol{\lambda}}^*)^\mathsf{T} \dot{\boldsymbol{\lambda}}^*}^{-(\boldsymbol{f}^*)^\mathsf{T} H_{\boldsymbol{x}}^*} \\ &= H_t(\boldsymbol{x}^*(t), \boldsymbol{u}^*(t), \boldsymbol{\lambda}^*(t), t)\,,\end{aligned} \tag{5.50}$$

where the notation $()^*$ denotes the evaluation of the respective function along the optimal solution. In particular, for time-invariant problems (5.37), i.e. l as well as $\boldsymbol{f}$ are independent of t, the Hamiltonian H is constant along the optimal trajectory.

Exercise 5.2 (Fixed end time & terminal condition) *Consider the problem*

$$\begin{aligned}\min_{u(\cdot)} \quad & \int_0^1 \frac{1}{2} u^2 + \frac{a}{2} x^2 \,\mathrm{d}t\,, \quad a > 0 \\ \text{s.t.} \quad & \dot{x} = u\,, \quad x(0) = 1, \quad x(1) = 0\,.\end{aligned}$$

Show that the solution is given by

$$x^*(t) = \frac{1}{1 - e^{2\sqrt{a}}} \left(e^{\sqrt{a}t} - e^{\sqrt{a}(2-t)}\right)\,, \quad u^*(t) = \frac{\sqrt{a}}{1 - e^{2\sqrt{a}}} \left(e^{\sqrt{a}t} + e^{\sqrt{a}(2-t)}\right) \tag{5.51}$$

in dependence of the parameter $a \geq 0$. Interpret the results shown in Figure 5.5.

Exercise 5.3 (Fixed end time & partial terminal condition) *Consider the problem*

$$\begin{aligned}\min_{u(\cdot)} \quad & \frac{a}{2} x_2(1)^2 + \int_0^1 \frac{1}{2} u^2 \,\mathrm{d}t\,, \quad a \geq 0 \\ \text{s.t.} \quad & \dot{x}_1 = x_2\,, \quad x_1(0) = 1, \quad x_1(1) = 0 \\ & \dot{x}_2 = u\,, \quad x_2(0) = 0\end{aligned}$$

Show that the (free) terminal state is $x_2^*(1) = -6/(4+a)$ and that the optimal solution is given by

$$x_1^*(t) = \frac{2(1+a)}{4+a} t^3 - \frac{3(2+a)}{a+4} t^2 + 1\,, \quad u^*(t) = \frac{12(1+a)}{4+a} t - \frac{6(2+a)}{a+4} \tag{5.52}$$

in dependence of the parameter $a \geq 0$. Interpret the results shown in Figure 5.6.

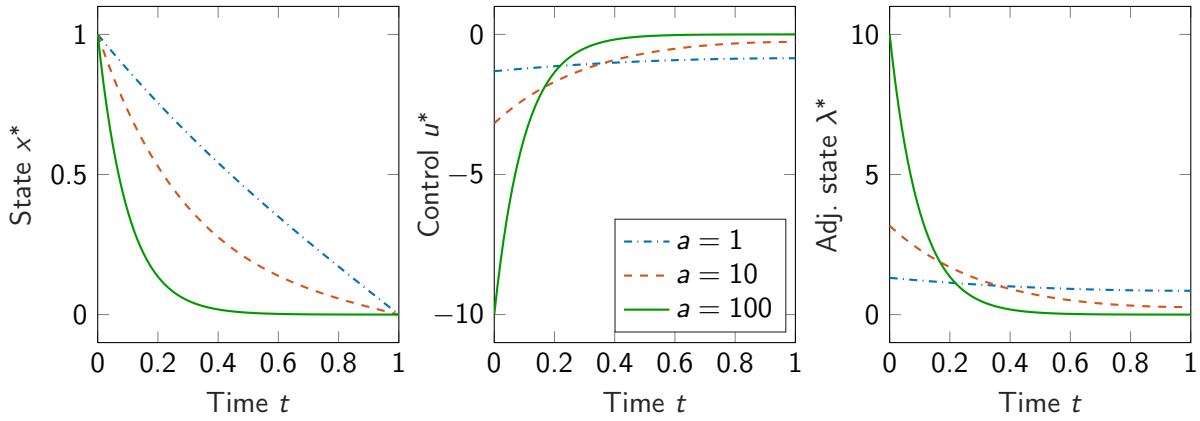


Figure 5.5: Optimal trajectories in Exercise 5.2.

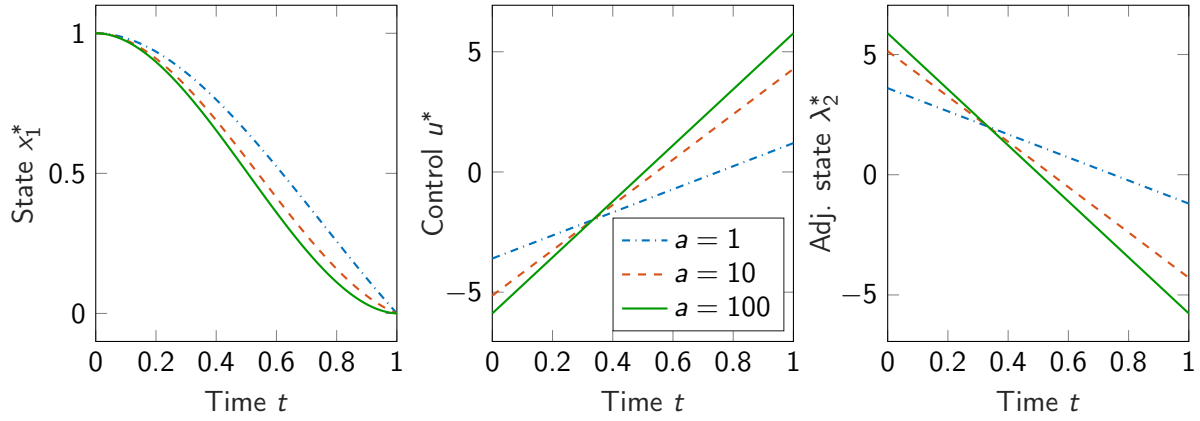


Figure 5.6: Optimal trajectories in Exercise 5.3.

5.3.3 Example with free end time

We consider a particle with mass m in the (x, y)-plane with the constant thrust force $F = ma$. The control input u is the angle between the thrust and the x-axis, see Figure 5.7. The goal is to steer the particle in minimum time t_f to the target point (x_f, y_f). Under the assumption that no other forces act on the particle besides the thrust, the optimal control problem can be formulated as follows

$$\begin{aligned}
\min_{u(\cdot)} \quad & t_f && && && \text{(5.53a)}\\
\text{s.t.} \quad & \dot{x} = v\,, && x(0) = x_0\,, && x(t_f) = x_f && \text{(5.53b)}\\
& \dot{v} = a\cos(u)\,, && v(0) = v_0 && && \text{(5.53c)}\\
& \dot{y} = w\,, && y(0) = y_0\,, && y(t_f) = y_f && \text{(5.53d)}\\
& \dot{w} = a\sin(u)\,, && w(0) = w_0\,. && && \text{(5.53e)}
\end{aligned}$$

The terminal condition is only defined for the position (x, y) and not for the velocity. In analogy to the partial terminal conditions (5.37c), the index set is $\mathcal{I}_f = \{x, y\}$. The Hamiltonian (5.45) of the problem is

$$H(\boldsymbol{x}, u, \boldsymbol{\lambda}) = \lambda_x v + \lambda_v a\cos(u) + \lambda_y w + \lambda_w a\sin(u) \tag{5.54}$$

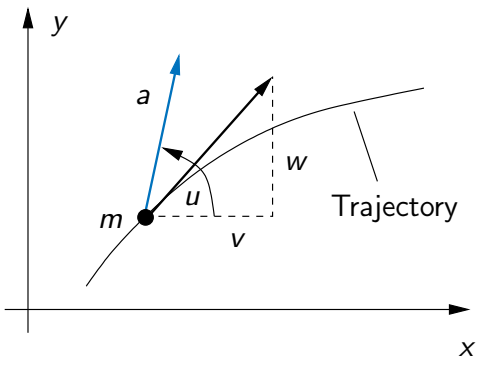


Figure 5.7: Motion of a particle with mass m in the (x, y)-plane.

with $\boldsymbol{x} = [x, v, y, w]^\mathrm{T}$ and the adjoint states $\boldsymbol{\lambda} = [\lambda_x, \lambda_v, \lambda_y, \lambda_w]^\mathrm{T}$. The adjoint system (5.46b) with the terminal conditions (5.47c) reads

$$\dot{\lambda}_x = -\frac{\partial H}{\partial x} = 0 \tag{5.55a}$$

$$\dot{\lambda}_v = -\frac{\partial H}{\partial v} = -\lambda_x\,, \quad \lambda_v(t_f) = 0 \tag{5.55b}$$

$$\dot{\lambda}_y = -\frac{\partial H}{\partial y} = 0 \tag{5.55c}$$

$$\dot{\lambda}_w = -\frac{\partial H}{\partial w} = -\lambda_y\,, \quad \lambda_w(t_f) = 0 \tag{5.55d}$$

with the solution

$$\lambda_x = const.\,, \quad \lambda_v = \lambda_x(t_f - t)\,, \quad \lambda_y = const.\,, \quad \lambda_w = \lambda_y(t_f - t)\,. \tag{5.56}$$

Moreover, the stationarity condition (5.46c) w.r.t. the control must be satisfied

$$\frac{\partial H}{\partial u} = -\lambda_v a \sin(u) + \lambda_w a \cos(u) = 0\,,$$

which can be solved for u

$$\tan(u) = \frac{\lambda_w}{\lambda_v} \overset{(5.56)}{=} \frac{\lambda_y(t_f - t)}{\lambda_x(t_f - t)} = \frac{\lambda_y}{\lambda_x} \quad \Rightarrow \quad u = \arctan\left(\frac{\lambda_y}{\lambda_x}\right) \quad \text{for} \quad -\frac{\pi}{2} < u < \frac{\pi}{2}\,. \tag{5.57}$$

This shows that the optimal control, i.e. the thrust angle, is constant over the whole time interval $[t_0 = 0, t_f]$. As a consequence, the states $x(t)$ and $y(t)$ follow from integrating the dynamics (5.53c), (5.53e) twice and inserting the initial conditions [3]

$$x(t) = g_x(\lambda_x, \lambda_y, t) = x_0 + v_0 t + \tfrac{1}{2} a \cos(u) t^2\,, \quad \cos(u) \overset{(5.57)}{=} \frac{1}{\sqrt{1 + \lambda_y^2/\lambda_x^2}} \tag{5.58a}$$

$$v(t) = g_v(\lambda_x, \lambda_y, t) = v_0 + a \cos(u) t \tag{5.58b}$$

$$y(t) = g_y(\lambda_x, \lambda_y, t) = y_0 + w_0 t + \tfrac{1}{2} a \sin(u) t^2\,, \quad \sin(u) \overset{(5.57)}{=} \frac{\lambda_y}{\lambda_x \sqrt{1 + \lambda_y^2/\lambda_x^2}} \tag{5.58c}$$

$$w(t) = g_w(\lambda_x, \lambda_y, t) = w_0 + a \sin(u) t\,. \tag{5.58d}$$

[3] Remember that $\sin(\arctan(b)) = b/\sqrt{1 + b^2}$, $\cos(\arctan(b)) = 1/\sqrt{1 + b^2}$.

In addition, the transversality condition (5.48) must be satisfied to account for the free end time t_f, whereby we can take advantage of the terminal conditions for the adjoint states $\lambda_v(t_f) = \lambda_w(t_f) = 0$

$$H(\boldsymbol{x}, u, \boldsymbol{\lambda})\Big|_{t=t_f} = -1 \quad \Rightarrow \quad \lambda_x\, g_v(\lambda_x, \lambda_y, t_f) + \lambda_y\, g_w(\lambda_x, \lambda_y, t_f) = -1\,. \tag{5.59}$$

The desired terminal conditions $x(t_f) = x_f$, $y(t_f) = y_f$ for the target point and the end time t_f must be computed from the equations (5.58a), (5.58c), (5.59). This results in a set of (nonlinear) equations for the three unknowns $(\lambda_x, \lambda_y, t_f)$

$$\begin{bmatrix} g_x(\lambda_x, \lambda_y, t_f) \\ g_y(\lambda_x, \lambda_y, t_f) \\ \lambda_x\, g_v(\lambda_x, \lambda_y, t_f) + \lambda_y\, g_w(\lambda_x, \lambda_y, t_f) \end{bmatrix} - \begin{bmatrix} x_f \\ y_f \\ -1 \end{bmatrix} = \boldsymbol{0} \tag{5.60}$$

that must be solved numerically.

A suitable MATLAB function for the solution of the nonlinear equations is fsolve from the Optimization Toolbox. The function fsolve uses the trust region method by default (see Section 3.4.1) to solve a set of equations in residual form $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$ as minimization problem in $\boldsymbol{x}$.

Figure 5.8 lists the MATLAB code for the particle problem as an example for the usage of fsolve to solve (5.60). The desired target point (x_f, y_f) is passed when the function particle(xf,yf) is called. It is assumed that the particle starts at the point $(x_0, y_0) = (0, 0)$ with the velocity $(v_0, w_0) = (0, 1)$ in vertical direction. Figure 5.9 shows the optimal trajectories $x^*(t)$, $y^*(t)$ of the particle in the (x, y)-plane for different target points (x_f, y_f). The arrows illustrate the (constant) direction $u = \arctan(\lambda_y/\lambda_x)$ of the thrust force $F = ma$.

5.3.4 General procedure and singular case

The previous examples illustrate how the optimality conditions (5.46a)-(5.48) can be solved in simple cases using analytical considerations. The general procedure for solving the optimality conditions consists of the following steps:

1. Set-up of the Hamiltonian $H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, t) = l(\boldsymbol{x}, \boldsymbol{u}, t) + \boldsymbol{\lambda}^\mathsf{T} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t)$.
2. Computation of $\boldsymbol{u}$ from (5.46c) such that $\boldsymbol{u}$ can be expressed as function of $(\boldsymbol{x}, \boldsymbol{\lambda}, t)$

   $$\boldsymbol{u} = \boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{\lambda}, t)\,. \tag{5.61}$$

3. Inserting the control function (5.61) into the canonical equations (5.46a)-(5.46b) leads to the boundary value problem

   $$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{\lambda}, t), t)\,, \quad \dot{\boldsymbol{\lambda}} = -H_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{\lambda}, t), \boldsymbol{\lambda}, t)\,, \quad \text{BCs} \tag{5.47}$$

   that is now independent of the control $\boldsymbol{u}$. If the end time t_f is free, then the transversality condition (5.48) extends the boundary conditions (BCs) (5.47).
4. The solution of the BVP leads to the trajectories $\boldsymbol{x}^*(t)$, $\boldsymbol{\lambda}^*(t)$, $t \in [t_0, t_f]$. The optimal control trajectory $\boldsymbol{u}^*(t)$ then follows from (5.61). In most cases, the BVP must be solved numerically (see Section 5.6).

```
function [t,x,y,p] = particle(xf,yf)
% -----------------------------------------------------
% (xf,yf): desired target point
% (t,x,y): trajectory of the particle
% p:       parameter structure

p.a = 1;                                        % parameters
p.x0=0; p.v0=0; p.y0=0; p.w0=1;                 % initial conditions
p.xf=xf; p.yf=yf;                               % terminal conditions (passed from function call)

opt  = optimset('Display','iter');              % options
X0   = [-1,0,1];                                % initial guess for fsolve
Xopt = fsolve(@eqns,X0,opt,p);                  % numerical solution with fsolve
p.lx=Xopt(1); p.ly=Xopt(2); p.tf=Xopt(3);       % solution

t = linspace(0,p.tf,100);                       % trajectories
x = xfct(p.lx,p.ly,t,p);
y = yfct(p.lx,p.ly,t,p);
% -----------------------------------------------------
function res = eqns(X,p)                        % equations in residual form
lx=X(1); ly=X(2); tf=X(3);
res = [ xfct(lx,ly,tf,p) - p.xf;
        yfct(lx,ly,tf,p) - p.yf;
        lx*vfct(lx,ly,tf,p) + ly*wfct(lx,ly,tf,p) + 1 ];
% -----------------------------------------------------
function x = xfct(lx,ly,t,p)                    % functions for x and v
cosu = 1/sqrt(1+(ly/lx)^2);
x = p.x0 + p.v0*t + p.a/2*cosu*t.^2;            % 't.^2' = component-wise evaluation
function v = vfct(lx,ly,t,p)
cosu = 1/sqrt(1+(ly/lx)^2);
v = p.v0 + p.a*cosu*t;
% -----------------------------------------------------
function y = yfct(lx,ly,t,p)                    % functions for y and w
sinu = ly/(lx*sqrt(1+(ly/lx)^2));
y = p.y0 + p.w0*t + p.a/2*sinu*t.^2;            % 't.^2' = component-wise evaluation
function w = wfct(lx,ly,t,p)
sinu = ly/(lx*sqrt(1+(ly/lx)^2));
w = p.w0 + p.a*sinu*t;
```

Figure 5.8: MATLAB code for the particle problem.

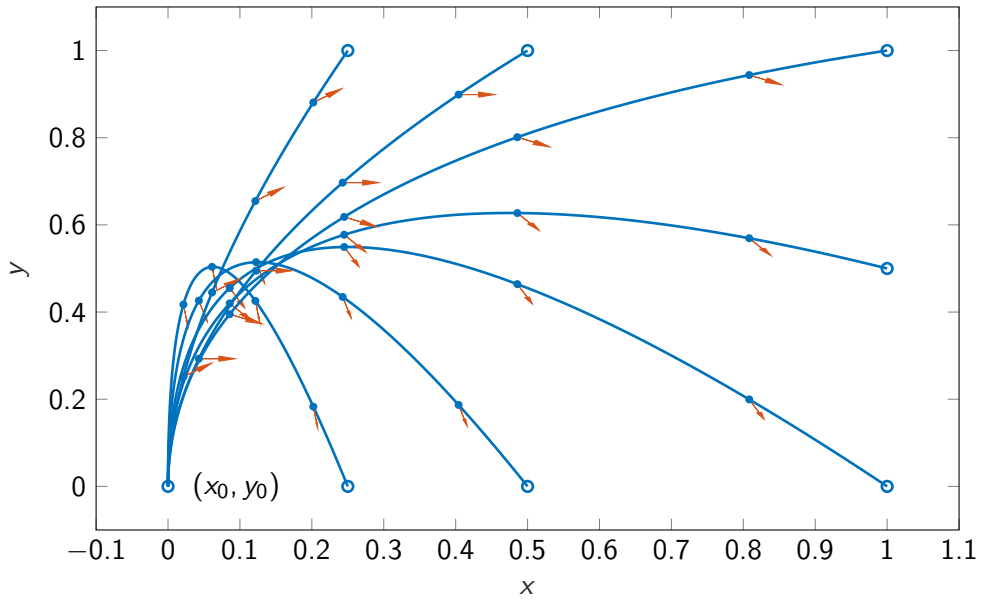


Figure 5.9: Time optimal control of a particle to different target points.

It is important to point out that the optimality conditions (5.46a)-(5.48) are necessary but not sufficient. Hence, the solution of the canonical equations does not automatically imply the optimality of the trajectories $\boldsymbol{x}^*(t)$, $\boldsymbol{u}^*(t)$. From a practical perspective, however, it is often assumed that a (unique) optimal solution exists for a given problem. Moreover, the plausability of the solution can often be verified by physical insight and process knowledge.

If the system function $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$ or the integral cost $l(\boldsymbol{x}, \boldsymbol{u}, t)$ of the cost functional (5.37a) depend nonlinearly on the control $\boldsymbol{u}$, it might be difficult to determine the function (5.61) analytically. In many practical cases, however, the numerical solution of (5.61) can be avoided:

Example 5.4 (Input-affine system & energy optimal cost term) *The computation of the control function (5.61) is straightforward for input-affine systems*

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{f}_0(\boldsymbol{x}) + \sum_{i=1}^{m} \boldsymbol{f}_i(\boldsymbol{x}) u_i \tag{5.62}$$

and an energy optimal term in the cost functional

$$J(\boldsymbol{u}) = V(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} l_0(\boldsymbol{x}) + \tfrac{1}{2} \sum_{i=1}^{m} r_i u_i^2 \, \mathrm{d}t \,, \quad r_i > 0 \,. \tag{5.63}$$

In this case, $\boldsymbol{u} = \boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{\lambda})$ with $\boldsymbol{\psi} = [\psi_1, \ldots, \psi_m]^\mathsf{T}$ can be explicitly computed from the single conditions $\frac{\partial H}{\partial u_i} = r_i u_i + \boldsymbol{\lambda}^\mathsf{T} \boldsymbol{f}_i(\boldsymbol{x}) = 0$, i.e.

$$u_i = \psi_i(\boldsymbol{x}, \boldsymbol{\lambda}) = -\frac{1}{r_i} \boldsymbol{\lambda}^\mathsf{T} \boldsymbol{f}_i(\boldsymbol{x}) \,, \quad i = 1, \ldots, m \,.$$

A special case occurs if one or more elements u_i of the control vector $\boldsymbol{u}$ cannot be determined from the stationarity condition (5.46c). This can easily be demonstrated for an input-affine system (5.62) with scalar control u ($m = 1$). If the integral term of the cost functional (5.37a) is independent of u, i.e. $l = l(\boldsymbol{x}, t)$, the Hamiltonian and its derivative w.r.t. u read

$$H(\boldsymbol{x}, u, \boldsymbol{\lambda}, t) = l(\boldsymbol{x}, t) + \boldsymbol{\lambda}^\mathsf{T} (\boldsymbol{f}_0(\boldsymbol{x}) + \boldsymbol{f}_1(\boldsymbol{x}) u) \,, \quad H_u(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{\lambda}^\mathsf{T} \boldsymbol{f}_1(\boldsymbol{x}) = 0 \,.$$

This case is referred to as *singular problem* because the control u cannot be determined from the stationarity condition. Nevertheless, $H_u(\boldsymbol{x}^*(t), \boldsymbol{\lambda}^*(t)) = 0$ must be satisfied, although it does not provide further information about the optimal control itself. This also implies that its time derivatives $\frac{\mathrm{d}^i}{\mathrm{d}t^i} H_u(\cdot)$ must vanish over the horizon $t \in [t_0, t_f]$. The first time derivative r for which the control u appears explicitly, i.e.

$$\frac{\mathrm{d}^r}{\mathrm{d}t^r} H_u(\boldsymbol{x}^*(t), \boldsymbol{\lambda}^*(t)) = 0 \quad \text{with} \quad \frac{\partial}{\partial u} \left(\frac{\mathrm{d}^r}{\mathrm{d}t^r} H_u(\cdot) \right) \neq 0$$

is used for computing $u^*(t)$. The index r is the *degree of singularity* of the optimal control problem.

It is obvious that the treatment of singular problems can be delicate and is problem-specific. In practice, the singular case can be avoided by adding a *regularization term* $\sum_{i=1}^{m} r_i u_i^2$ to the integral cost function with sufficiently small weights $r_i > 0$. This ensures that u_i does not vanish in the single stationarity conditions $H_{u_i} = 0$, also see Example 5.4.

5.3.5 Interpretation of adjoint variables

In the parameter optimization case, it was shown that the Lagrange multipliers λ_i and μ_i correspond to the sensitivity of the cost function f at the optimum w.r.t. small changes of the corresponding constraints (see Section 4.2.4). A similar interpretation can be derived for the adjoint variables $\boldsymbol{\lambda}^*(t)$ of an optimal control problem. To this end, we consider a fixed end time problem without terminal constraints and terminal cost

$$\min_{\boldsymbol{u}(\cdot)} \quad \int_{t_0}^{t_f} l(\boldsymbol{x}, \boldsymbol{u}, t)\, \mathrm{d}t \tag{5.64a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t)\,, \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0\,. \tag{5.64b}$$

We assume that an optimal and continuous solution $\boldsymbol{x}^*(t)$, $\boldsymbol{u}^*(t)$ exists and that the functions $\boldsymbol{f}$ and l are continuously differentiable.

In the following, we consider a small perturbation $\boldsymbol{\varepsilon}$ of the initial state $\boldsymbol{x}(t_0) = \boldsymbol{x}_0 + \boldsymbol{\varepsilon}$ with the associated optimal trajectories $\boldsymbol{x}^{\boldsymbol{\varepsilon}}(t)$, $\boldsymbol{u}^{\boldsymbol{\varepsilon}}(t)$ that satisfy the system dynamics (5.64b)

$$\dot{\boldsymbol{x}}^{\boldsymbol{\varepsilon}}(t) = \boldsymbol{f}(\boldsymbol{x}^{\boldsymbol{\varepsilon}}(t), \boldsymbol{u}^{\boldsymbol{\varepsilon}}(t), t)\,, \quad \boldsymbol{x}^{\boldsymbol{\varepsilon}}(t_0) = \boldsymbol{x}_0 + \boldsymbol{\varepsilon} \tag{5.65}$$

and define the nominal optimal solution for $\boldsymbol{\varepsilon} = 0$

$$\boldsymbol{x}^*(t) = \boldsymbol{x}^0(t)\,, \quad \boldsymbol{u}^*(t) = \boldsymbol{u}^0(t)\,, \quad t \in [t_0, t_f]\,. \tag{5.66}$$

In the next step, we add the dynamics (5.65) to the cost functional (5.64a)

$$\begin{aligned} V(\boldsymbol{x}^{\boldsymbol{\varepsilon}}(t_0), t_0) &= \int_{t_0}^{t_f} l(\boldsymbol{x}^{\boldsymbol{\varepsilon}}(t), \boldsymbol{u}^{\boldsymbol{\varepsilon}}(t), t)\, \mathrm{d}t \\ &= \int_{t_0}^{t_f} l(\boldsymbol{x}^{\boldsymbol{\varepsilon}}(t), \boldsymbol{u}^{\boldsymbol{\varepsilon}}(t), t) + \boldsymbol{\lambda}^*(t)^\mathsf{T}\left[\boldsymbol{f}(\boldsymbol{x}^{\boldsymbol{\varepsilon}}(t), \boldsymbol{u}^{\boldsymbol{\varepsilon}}(t), t) - \dot{\boldsymbol{x}}^{\boldsymbol{\varepsilon}}(t)\right] \mathrm{d}t\,. \end{aligned}$$

where the adjoint variables $\boldsymbol{\lambda}^*(t)$ correspond to the nominal optimal solution $\boldsymbol{x}^*(t)$, $\boldsymbol{u}^*(t)$.

Under the assumption that $\boldsymbol{x}^{\boldsymbol{\varepsilon}}(t)$ and $\boldsymbol{u}^{\boldsymbol{\varepsilon}}(t)$ are continuously differentiable in $\boldsymbol{\varepsilon}$, we integrate by parts and differentiate afterwards w.r.t. $\boldsymbol{\varepsilon}$, which leads to (omitting arguments)

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\varepsilon}} V(\boldsymbol{x}^{\boldsymbol{\varepsilon}}(t_0), t_0) = -\left[(\boldsymbol{x}_{\boldsymbol{\varepsilon}}^{\boldsymbol{\varepsilon}})^\mathsf{T}\boldsymbol{\lambda}^*\right]_{t_0}^{t_f} + \int_{t_0}^{t_f} (\boldsymbol{x}_{\boldsymbol{\varepsilon}}^{\boldsymbol{\varepsilon}})^\mathsf{T}\left[l_x(\boldsymbol{x}^{\boldsymbol{\varepsilon}}, \boldsymbol{u}^{\boldsymbol{\varepsilon}}, t) + \boldsymbol{f}_x(\boldsymbol{x}^{\boldsymbol{\varepsilon}}, \boldsymbol{u}^{\boldsymbol{\varepsilon}}, t)^\mathsf{T}\boldsymbol{\lambda}^* + \dot{\boldsymbol{\lambda}}^*\right] \\ + (\boldsymbol{u}_{\boldsymbol{\varepsilon}}^{\boldsymbol{\varepsilon}})^\mathsf{T}\left[l_u(\boldsymbol{x}^{\boldsymbol{\varepsilon}}, \boldsymbol{u}^{\boldsymbol{\varepsilon}}, t) + \boldsymbol{f}_u(\boldsymbol{x}^{\boldsymbol{\varepsilon}}, \boldsymbol{u}^{\boldsymbol{\varepsilon}}, t)^\mathsf{T}\boldsymbol{\lambda}^*\right] \mathrm{d}t\,. \end{aligned}$$

Note that the sensitivities $\boldsymbol{x}_{\boldsymbol{\varepsilon}}^{\boldsymbol{\varepsilon}}(t) = \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\varepsilon}}\boldsymbol{x}^{\boldsymbol{\varepsilon}}(t) \in \mathbb{R}^{n\times n}$ and $\boldsymbol{u}_{\boldsymbol{\varepsilon}}^{\boldsymbol{\varepsilon}}(t) = \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\varepsilon}}\boldsymbol{u}^{\boldsymbol{\varepsilon}}(t) \in \mathbb{R}^{m\times n}$ are time-varying matrices. The equation above is a vector equation, where the integral has to be understood component-wise. We recover the nominal optimal solution (5.66) for $\boldsymbol{\varepsilon} = \boldsymbol{0}$, i.e.

$$\begin{aligned} &\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\varepsilon}} V(\boldsymbol{x}^{\boldsymbol{\varepsilon}}(t_0), t_0)\Big|_{\boldsymbol{\varepsilon}=\boldsymbol{0}} = -\boldsymbol{x}_{\boldsymbol{\varepsilon}}^{\boldsymbol{0}}(t_f)^\mathsf{T}\boldsymbol{\lambda}^*(t_f) + \boldsymbol{x}_{\boldsymbol{\varepsilon}}^{\boldsymbol{0}}(t_0)^\mathsf{T}\boldsymbol{\lambda}^*(t_0) \\ &+\int_{t_0}^{t_f} \boldsymbol{x}_{\boldsymbol{\varepsilon}}^{\boldsymbol{0}}(t)^\mathsf{T}\left[l_x(\boldsymbol{x}^*, \boldsymbol{u}^*, t)+\boldsymbol{f}_x(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T}\boldsymbol{\lambda}^*+\dot{\boldsymbol{\lambda}}^*\right]+\boldsymbol{u}_{\boldsymbol{\varepsilon}}^{\boldsymbol{0}}(t)^\mathsf{T}\left[l_u(\boldsymbol{x}^*, \boldsymbol{u}^*, t)+\boldsymbol{f}_u(\boldsymbol{x}^*, \boldsymbol{u}^*, t)^\mathsf{T}\boldsymbol{\lambda}^*\right] \mathrm{d}t\,. \end{aligned}$$

This equation can be drastically simplified because the triplet $(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*)$ satisfies the necessary optimality conditions (5.46b)-(5.46c) with $\boldsymbol{\lambda}^*(t_f) = \boldsymbol{0}$, which leads to

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\varepsilon}} V(\boldsymbol{x}^{\boldsymbol{\varepsilon}}(t_0), t_0)\Big|_{\boldsymbol{\varepsilon}=\boldsymbol{0}} = \boldsymbol{x}_{\boldsymbol{\varepsilon}}^{\boldsymbol{0}}(t_0)^\mathsf{T}\boldsymbol{\lambda}^*(t_0) \quad \Rightarrow \quad V_x(\boldsymbol{x}_0, t_0) = \boldsymbol{\lambda}^*(t_0) \tag{5.67}$$

with $V_{\boldsymbol{x}} = \frac{\partial V}{\partial \boldsymbol{x}}$. In the last relation, we exploited the fact that the sensitivity $\boldsymbol{x}_{\boldsymbol{\xi}}^{\boldsymbol{\varepsilon}}(t_0)$ at the initial time $t = t_0$ is exactly the $(n \times n)$-unit matrix $\boldsymbol{I}$, which is easily seen from (5.65).

The relation (5.67) illustrates that the adjoint variables $\boldsymbol{\lambda}^*(t_0)$ at the initial time $t = t_0$ are a measure for the sensitivity of the value of the cost functional w.r.t. perturbations of the initial state $\boldsymbol{x}_0$, also see Figure 5.5 and 5.6 of Exercise 5.2 and 5.3. This important insight for the initial time point can be generalized to the time interval $[t_0, t_f]$ when applying the *principle of optimality*.

Theorem 5.4 (Principle of optimality) *Suppose that $\boldsymbol{u}^*(t)$, $t \in [t_0, t_f]$ is an optimal control of the problem (5.64) with the associated state trajectory $\boldsymbol{x}^*(t)$. Then, for each subinterval $[t_1, t_f]$ with $t_0 \leq t_1 \leq t_f$, the control $\boldsymbol{u}^*(t)$, $t \in [t_1, t_f]$ is the optimal solution of the problem*

$$\min_{\boldsymbol{u}(\cdot)} \quad \int_{t_1}^{t_f} l(\boldsymbol{x}, \boldsymbol{u}, t)\, \mathrm{d}t \tag{5.68a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t)\,, \quad \boldsymbol{x}(t_1) = \boldsymbol{x}^*(t_1)\,. \tag{5.68b}$$

Thanks to the principle of optimality, the derivation of (5.67) can be transferred to the remainder problem (5.68), i.e. $V_{\boldsymbol{x}}(\boldsymbol{x}^*(t_1), t_1) = \boldsymbol{\lambda}^*(t_1)$. Since the time point t_1 can be freely chosen within the interval $[t_0, t_f]$, we have

$$\boxed{V_{\boldsymbol{x}}(\boldsymbol{x}^*(t), t) = \boldsymbol{\lambda}^*(t)\,, \quad t \in [t_0, t_f]\,.} \tag{5.69}$$

Thus, the adjoint state $\boldsymbol{\lambda}^*(t)$ is a measure of how strongly the cost value would be affected by a small perturbation of the state $\boldsymbol{x}^*(t)$ at time t.

5.3.6 General terminal conditions

The case of partial terminal conditions is the most common one in practice and nicely reflects the duality between the terminal conditions (5.47b) and (5.47c) for the system state $\boldsymbol{x}$ and the adjoint state $\boldsymbol{\lambda}$. However, the derivation of the optimality conditions via the calculus of variations also allows for handling more general terminal conditions of the form

$$\boldsymbol{g}(\boldsymbol{x}(t_f), t_f) = \boldsymbol{0}\,, \quad \boldsymbol{g} : \mathbb{R}^n \times \mathbb{R}_+ \to \mathbb{R}^r\,. \tag{5.70}$$

Similar to the procedure in Section 5.3.1, we add the terminal conditions (5.70) to the extended cost functional (5.38) by introducing further multipliers $\boldsymbol{\nu} \in \mathbb{R}^r$

$$\bar{J} = V(\boldsymbol{x}(t_f)) + \boldsymbol{\nu}^{\mathsf{T}} \boldsymbol{g}(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} l(\boldsymbol{x}(t), \boldsymbol{u}(t), t) + \boldsymbol{\lambda}^{\mathsf{T}} (\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t) - \dot{\boldsymbol{x}})\, \mathrm{d}t\,. \tag{5.71}$$

By considering the first variation of the cost functional and applying partial integration as well as the fundamental lemma of the calculus of variations, it can be shown that the transversality condition (5.47c) for an optimal solution is extended by

$$\boldsymbol{\lambda}^*(t_f) = V_{\boldsymbol{x}}(\boldsymbol{x}^*(t_f), t_f) + \boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{x}^*(t_f), t_f)^{\mathsf{T}} \boldsymbol{\nu}^*\,. \tag{5.72}$$

If the end time t_f is free, the transversality condition (5.48) for the Hamiltonian is modified accordingly

$$H(\boldsymbol{x}^*(t_f^*), \boldsymbol{u}^*(t_f^*), \boldsymbol{\lambda}^*(t_f^*), t_f^*) = -V_t(\boldsymbol{x}^*(t_f^*), t_f^*) - \boldsymbol{g}_t(\boldsymbol{x}^*(t_f^*), t_f^*)^{\mathsf{T}} \boldsymbol{\nu}^*\,. \tag{5.73}$$

In summary, the optimality conditions comprise $2n$ differential equations (5.46a), (5.46b) and $2n + r$ boundary conditions (5.47a), (5.70), (5.72) for the $2n$ states $\boldsymbol{x}^*(t)$, $\boldsymbol{\lambda}^*(t)$ and the r (constant) multipliers $\boldsymbol{\nu}^*$ as well as m algebraic equations (5.46c) for the control $\boldsymbol{u}^*(t)$. The scalar transversality condition (5.73) is added to the equations if the end time t_f^* is free.

5.4 Optimization of linear systems with quadratic cost functional

A large class of optimal control problems concern linear systems with quadratic cost functional

$$\min_{\boldsymbol{u}(\cdot)} \quad J(\boldsymbol{u}) = \frac{1}{2}\boldsymbol{x}(t_f)^\mathsf{T}\boldsymbol{S}\,\boldsymbol{x}(t_f) + \frac{1}{2}\int_{t_0}^{t_f} \boldsymbol{x}(t)^\mathsf{T}\boldsymbol{Q}\,\boldsymbol{x}(t) + \boldsymbol{u}(t)^\mathsf{T}\boldsymbol{R}\,\boldsymbol{u}(t)\;\mathrm{d}t \tag{5.74a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}\,, \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0\,. \tag{5.74b}$$

The terminal state $\boldsymbol{x}(t_f)$ is free and we assume that the weighting matrices $\boldsymbol{S}$, $\boldsymbol{Q}$ are positive semidefinite and $\boldsymbol{R}$ is positive definite. Moreover, the system matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ as well as the weighting matrices $\boldsymbol{Q}$, $\boldsymbol{R}$, $\boldsymbol{S}$ are supposed to be time-invariant although the considerations in the next section directly apply to the time-varying case.

The optimal control problem (5.74) can be interpreted as follows: starting from $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$, bring the system (5.74b) close to the origin $\boldsymbol{x}(t_f) \approx \boldsymbol{0}$ within the time interval $[t_0, t_f]$, whereby the weighting matrices $\boldsymbol{Q}$, $\boldsymbol{R}$, $\boldsymbol{S}$ determine the state and input behavior of the system, also see Exercise 5.2 and 5.3.

5.4.1 Riccati differential equation

In the following, we derive the optimality conditions for the linear-quadratic problem (5.74). With the Hamiltonian

$$H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, t) = \frac{1}{2}\Big(\boldsymbol{x}^\mathsf{T}\boldsymbol{Q}\,\boldsymbol{x} + \boldsymbol{u}^\mathsf{T}\boldsymbol{R}\,\boldsymbol{u}\Big) + \boldsymbol{\lambda}^\mathsf{T}\Big(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}\Big)\,,$$

the optimality conditions (5.46a)-(5.47c) become

$$\dot{\boldsymbol{x}}^* = \quad H_{\boldsymbol{\lambda}} = \boldsymbol{A}\boldsymbol{x}^* + \boldsymbol{B}\boldsymbol{u}^*\,, \qquad \boldsymbol{x}(t_0) = \boldsymbol{x}_0 \tag{5.75a}$$

$$\dot{\boldsymbol{\lambda}}^* = -H_{\boldsymbol{x}} = -\boldsymbol{Q}\boldsymbol{x}^* - \boldsymbol{A}^\mathsf{T}\boldsymbol{\lambda}^*\,, \quad \boldsymbol{\lambda}^*(t_f) = \boldsymbol{S}\boldsymbol{x}^*(t_f) \tag{5.75b}$$

$$\boldsymbol{0} = \quad H_{\boldsymbol{u}} = \boldsymbol{R}\boldsymbol{u}^* + \boldsymbol{B}^\mathsf{T}\boldsymbol{\lambda}^*\,. \tag{5.75c}$$

Solving (5.75c) for the optimal control

$$\boldsymbol{u}^*(t) = -\boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T}\boldsymbol{\lambda}^*(t) \tag{5.76}$$

and inserting into the canonical equations (5.75a), (5.75b) yields the two-point BVP

$$\begin{bmatrix}\dot{\boldsymbol{x}}^* \\ \dot{\boldsymbol{\lambda}}^*\end{bmatrix} = \begin{bmatrix}\boldsymbol{A} & -\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T} \\ -\boldsymbol{Q} & -\boldsymbol{A}^\mathsf{T}\end{bmatrix}\begin{bmatrix}\boldsymbol{x}^* \\ \boldsymbol{\lambda}^*\end{bmatrix}, \quad \begin{matrix}\boldsymbol{x}^*(t_0) = \boldsymbol{x}_0 \\ \boldsymbol{\lambda}^*(t_f) = \boldsymbol{S}\boldsymbol{x}^*(t_f)\,.\end{matrix} \tag{5.77}$$

It can be shown by means of the transition matrix of (5.77) that $\boldsymbol{x}^*(t)$ and $\boldsymbol{\lambda}^*(t)$ are coupled via a time-varying matrix $\boldsymbol{P}(t)$ with the terminal condition $\boldsymbol{P}(t_f) = \boldsymbol{S}$, i.e.

$$\boldsymbol{\lambda}^*(t) = \boldsymbol{P}(t)\boldsymbol{x}^*(t)\,, \quad t \in [t_0, t_f]\,. \tag{5.78}$$

By differentiating (5.78), $\dot{\boldsymbol{\lambda}}^* = \dot{\boldsymbol{P}}\boldsymbol{x}^* + \boldsymbol{P}\dot{\boldsymbol{x}}^*$, and inserting it into the differential equations (5.77), we get

$$\left[\dot{\boldsymbol{P}} + \boldsymbol{P}\boldsymbol{A} + \boldsymbol{A}^\mathsf{T}\boldsymbol{P} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T}\boldsymbol{P} + \boldsymbol{Q}\right]\boldsymbol{x}^*(t) = \boldsymbol{0}\,.$$

In order for this relation to hold for all potential optimal trajectories $\boldsymbol{x}^*(t)$ and all times $t \in [t_0, t_f]$, it is not only sufficient but also necessary that the bracketed term vanishes. This leads to the so-called *Riccati differential equation*

$$\dot{\boldsymbol{P}} = -\boldsymbol{P}\boldsymbol{A} - \boldsymbol{A}^\mathsf{T}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T}\boldsymbol{P} - \boldsymbol{Q}\,, \quad \boldsymbol{P}(t_f) = \boldsymbol{S}\,. \tag{5.79}$$

Note that the Riccati differential equation is nonlinear because $\boldsymbol{P}(t)$ appears quadratically in the term $\boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T}\boldsymbol{P}$. Inserting (5.78) into (5.76) gives

$$\boldsymbol{u}^*(t) = -\boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T}\boldsymbol{P}(t)\boldsymbol{x}^*(t)\,. \tag{5.80}$$

Some important aspects in this context are:

- The Riccati differential equation (5.79) is typically solved backward in time to use the terminal condition $\boldsymbol{P}(t_f) = \boldsymbol{S}$ as initial condition.
- $\boldsymbol{P}(t)$ is symmetric and positive semidefinite. Thus, the Riccati differential equation of order n^2 is equivalent to $n(n+1)/2$ coupled differential equations.
- The derivation is the same for time-varying system matrices $\boldsymbol{A}(t)$, $\boldsymbol{B}(t)$ and/or weighting matrices $\boldsymbol{Q}(t)$, $\boldsymbol{R}(t)$.
- After computing $\boldsymbol{P}(t)$, the optimal trajectories for the state $\boldsymbol{x}^*(t)$ and control $\boldsymbol{u}^*(t)$ follow from integrating the system dynamics (5.75a), i.e. $\dot{\boldsymbol{x}}^* = (\boldsymbol{A} - \boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T}\boldsymbol{P}(t))\boldsymbol{x}^*$ with $\boldsymbol{x}^*(t_0) = \boldsymbol{x}_0$ and eventually evaluating (5.80).

The Riccati differential equation is also important in the context of *optimal feedback control*, because the relation (5.80) corresponds to the optimal feedback law

$$\boldsymbol{u}(t) = -\boldsymbol{K}(t)\boldsymbol{x}(t) \quad \text{with} \quad \boldsymbol{K}(t) = \boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T}\boldsymbol{P}(t)\,, \tag{5.81}$$

where $\boldsymbol{K}(t)$ is considered as time-varying feedback gain matrix.

5.4.2 Example: Swing-up of the pendulum on a cart

The linear-quadratic control design is particularly useful in the context of the *two-degrees-of-freedom control scheme* (2DOF) in Figure 5.10, which is often used in practice for trajectory tracking. The main idea behind the 2DOF structure is to separately design the control performance regarding reference tracking and disturbance rejection. A reference trajectory $\boldsymbol{u}_{ref}(t)$ is injected as *feedforward control* that allows the system to exactly track the reference state trajectory $\boldsymbol{x}_{ref}(t)$ in the nominal case. The feedback controller as the second degree-of-freedom

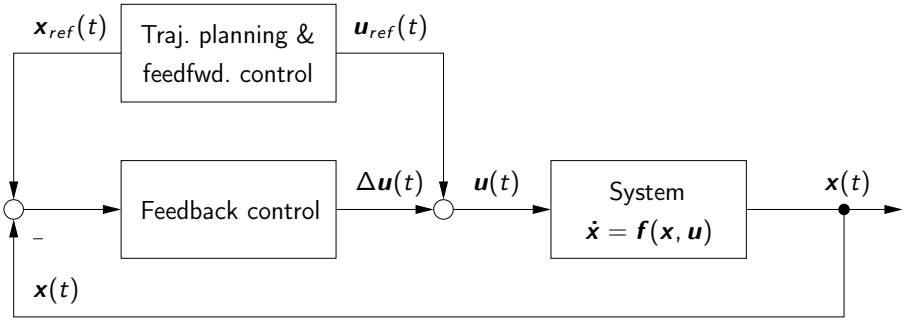


Figure 5.10: Two-degrees-of-freedom control scheme.

stabilizes the system along the reference trajectory $\boldsymbol{x}_{ref}(t)$ and compensates for model inaccuracies and disturbances.

To design the feedback controller, the nonlinear system $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$ can be linearized around the reference trajectories leading to the linear time-varying system

$$\Delta\dot{\boldsymbol{x}} = \boldsymbol{A}(t)\Delta\boldsymbol{x} + \boldsymbol{B}(t)\Delta\boldsymbol{u} \quad \text{with} \quad \boldsymbol{A}(t) = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\Big|_{\substack{\boldsymbol{x}_{ref}(t) \\ \boldsymbol{u}_{ref}(t)}} \quad \text{and} \quad \boldsymbol{B}(t) = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\Big|_{\substack{\boldsymbol{x}_{ref}(t) \\ \boldsymbol{u}_{ref}(t)}} . \tag{5.82}$$

The state $\Delta\boldsymbol{x}(t) = \boldsymbol{x}(t) - \boldsymbol{x}_{ref}(t)$ and control $\Delta\boldsymbol{u}(t) = \boldsymbol{u}(t) - \boldsymbol{u}_{ref}(t)$ of the linearized system denote the deviations from the reference trajectories. The Riccati differential equation can be used to derive an optimal feedback law of the form (5.81) to stabilize the nonlinear system around the reference trajectory $\boldsymbol{x}_{ref}(t)$.

The concept is illustrated for the swing-up of a pendulum on a cart that is depicted in Figure 5.11. A simplified nonlinear model for the cart position y and the angle ϕ of the pendulum is[4]

$$\ddot{y} = u\,, \quad \ddot{\phi} = \frac{3}{2l}\Big(g\sin(\phi) - \cos(\phi)u\Big) \tag{5.83}$$

with the length l of the pendulum and the acceleration due to gravity g. The input u is the acceleration $\ddot{y}$ of the cart.

The control task is the swing-up of the pendulum from the lower to the upper (inverted) equilibrium, i.e.

$$\left.\begin{aligned} y(0) &= 0\,, & \dot{y}(0) &= 0 \\ \phi(0) &= -\pi\,, & \dot{\phi}(0) &= 0 \end{aligned}\right\} \quad \Rightarrow \quad \left\{\begin{aligned} y(t_f) &= 0\,, & \dot{y}(t_f) &= 0 \\ \phi(t_f) &= 0\,, & \dot{\phi}(t_f) &= 0\,. \end{aligned}\right. \tag{5.84}$$

A suitable feedforward control trajectory for the swing-up of a pendulum with length $l = 1\,\text{m}$ in the time $t_f = 2\,\text{s}$ is

$$u_{ref}(t) = \sum_{i=1}^{5} c_i\, t^i\,, \quad t \in [0, t_f = 2\ \text{s}] \tag{5.85}$$

with the coefficients $\boldsymbol{c} = [-135.128, 625.696, -888.598, 497.538, -96.3861]^\mathrm{T}$ (neglecting SI units). This trajectory was computed with the methodology [10] which is not further detailed here.

[4] The model can be derived with the Lagrange formalism and neglecting friction for the sake of simplicity. The moment of inertia of the pendulum is $J = \frac{1}{12}ml^2$ with the mass m. The pendulum dynamics (5.83) are independent of m because the mass m of the pendulum is cancelled out in the equations.

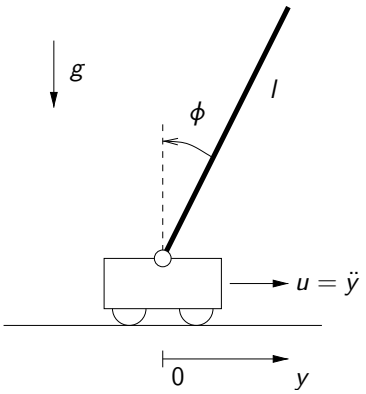


Figure 5.11: Pendulum on a cart.

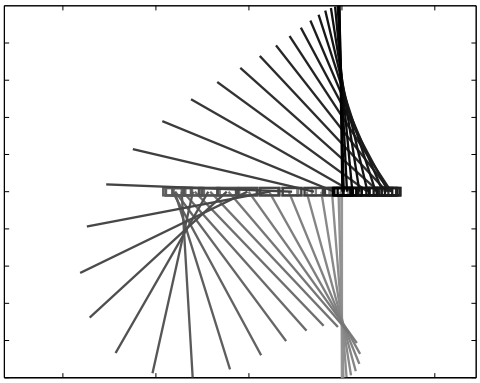


Figure 5.12: Snapshots of the swing-up.

Figure 5.12 illustrates the swing-up motion of the pendulum. The corresponding state trajectory $\boldsymbol{x}_{ref}(t)$ with $\boldsymbol{x} = [y, \dot{y}, \phi, \dot{\phi}]^\mathsf{T}$ is shown in Figure 5.13 (- - -) and is computed by numerically integrating the dynamics (5.83).

Due to potential model inaccuracies and the instability of the inverted pendulum position, a feedback controller is essential to stabilize the pendulum during the swing-up maneuver. To this end, the nonlinear model (5.83) is linearized around the reference trajectories $\boldsymbol{x}_{ref}(t)$ and $\boldsymbol{u}_{ref}(t)$ leading to the linear time-varying system (5.82)

$$\Delta\dot{\boldsymbol{x}} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{3}{2l}[g\cos\phi_{ref}(t) + \sin\phi_{ref}(t)u_{ref}(t)] & 0 \end{bmatrix}}_{=\boldsymbol{A}(t)} \Delta\boldsymbol{x} + \underbrace{\begin{bmatrix} 0 \\ 1 \\ 0 \\ -\frac{3}{2l}\cos\phi_{ref}(t) \end{bmatrix}}_{=\boldsymbol{b}(t)} \Delta u \quad (5.86)$$

with $\boldsymbol{x}(t) = \boldsymbol{x}_{ref}(t) + \Delta\boldsymbol{x}(t)$ and $u(t) = u_{ref}(t) + \Delta u(t)$. The Riccati control law (5.81) is computed for the cost functional (5.74a)

$$J(\Delta u) = \frac{1}{2}\Delta\boldsymbol{x}(t_f)^\mathsf{T}\boldsymbol{S}\,\Delta\boldsymbol{x}(t_f)^\mathsf{T} + \frac{1}{2}\int_0^{t_f} \Delta\boldsymbol{x}(t)^\mathsf{T}\boldsymbol{Q}\Delta\boldsymbol{x}(t) + R\Delta u(t)^2\,\mathrm{d}t\,, \quad t_f = 2\,\mathrm{s} \quad (5.87)$$

with the diagonal matrices $\boldsymbol{S} = \boldsymbol{Q} = \mathrm{diag}(1,1,1,1)$. The weight for the control is chosen as $R = 0.01$ and therefore relatively small compared to the states, which corresponds to a fast control action.

The Riccati differential equation (5.79) must be integrated numerically to obtain the matrix $\boldsymbol{P}(t)$ on the time interval $t \in [0, t_f]$. The optimal feedback law for the swing-up of the pendulum follows from (5.81)

$$\Delta u(t) = -\boldsymbol{k}(t)^\mathsf{T}\Delta\boldsymbol{x}(t) = \frac{1}{R}\,\boldsymbol{b}(t)^\mathsf{T}\boldsymbol{P}(t)\Big(\boldsymbol{x}_{ref}(t) - \boldsymbol{x}(t)\Big)\,. \quad (5.88)$$

For simplicity, we assume that the full state vector $\boldsymbol{x} = [y, \dot{y}, \phi, \dot{\phi}]^\mathsf{T}$ is available and no state observer is necessary. The 2DOF control structure in Figure 5.10 shows that the control u that is applied to the pendulum consists of the combination of the feedforward and feedback part

$$u(t) = u_{ref}(t) + \Delta u(t)\,. \quad (5.89)$$

The simulated trajectories for the swing-up of the pendulum in Figure 5.13 show the open-loop case (only feedforward action without feedback) and the closed-loop case (feedforward and feedback) for a disturbance of the initial downward position. The open-loop trajectories illustrate that the pendulum fails to follow the reference trajectories and falls down before the upper unstable position is reached. In closed-loop, the pendulum is stabilized along the swing-up trajectories despite the initial disturbance. The plot of the control signal shows that the feedback correction $\Delta u(t)$ requires approximately 1.2 s to stabilize the initial disturbance, before the control signal is dominated by the feedforward part $u_{ref}(t)$.

Another interesting aspect is that the feedback gains $\boldsymbol{k}(t)$ in Figure 5.13 change sign which is due to the fact the pendulum is transferred from the lower stable position to the upper unstable one. The change of sign is an indicator that the linear time-varying system (5.86) temporarily looses controllability during the swing-up. For an experimental validation of the swing-up, it would be recommended to reduce the feedback gains $\boldsymbol{k}(t)$ to zero during these critical time intervals and to steer through them open-loop with the feedforward control $u_{ref}(t)$.

Figure 5.14 lists the MATLAB code for solving the Riccati differential equation (5.79), the computation of the feedback gains (5.88), and the simulation of the swing-up with the perturbed initial state (5.84). For simplicity, the Riccati differential equation (5.79) is numerically integrated for all elements of the matrix $\boldsymbol{P}(t)$ although $\boldsymbol{P}(t)$ is symmetric as pointed out before.

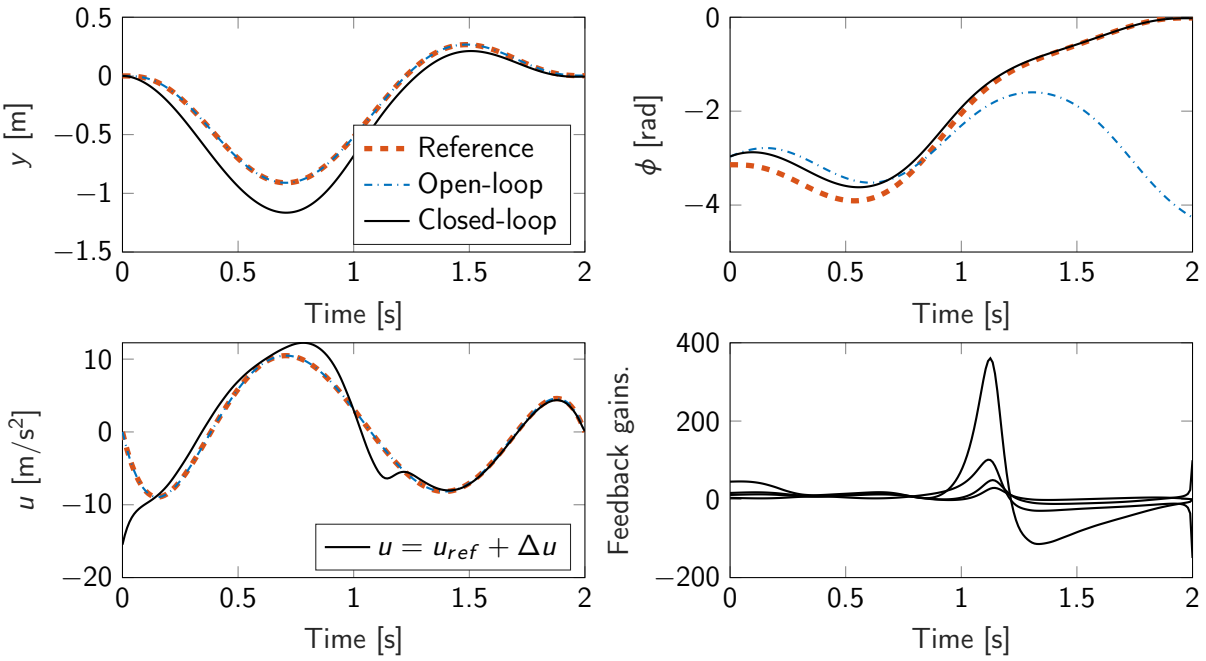


Figure 5.13: Simulation results for the pendulum swing-up.

5.4.3 Algebraic Riccati equation

The computation of the optimal feedback (5.81) is particularly simple if the time interval $[t_0, t_f]$ of the cost functional (5.74a) is extended to infinity with $t_f \to \infty$. In this case, the terminal cost $\boldsymbol{x}(t_f)^{\mathsf{T}} \boldsymbol{S}\, \boldsymbol{x}(t_f)$ can be discarded since $\lim_{t\to\infty} \boldsymbol{x}(t) = \boldsymbol{0}$ has to hold anyway or the integral in (5.74a) would not be finite. We therefore set $\boldsymbol{S} = \boldsymbol{0}$.

```
function vec = pendulum
% ------------------------------------------------------------
g = 9.81; l = 1;                                                % parameters
p.a1 = 3*g/(2*l); p.a2 = 3/(2*l); p.n=4; p.steps = 200;

p.tf = 2;                                                       % reference trajectories
vec.t = linspace(0,p.tf,p.steps);                               % ----------------------
c = [-135.128, 625.696, -888.598, 497.538, -96.3861];           % feedforward trajectory
vec.uref = c(1)*vec.t+c(2)*vec.t.^2+c(3)*vec.t.^3+c(4)*vec.t.^4+c(5)*vec.t.^5;
[tsim,xsim] = ode45(@pendulum_sim,vec.t,[0;0;-pi;0],[],vec,p, 0);     % num. int. for state traj.
vec.xref = xsim';

p.Q = diag([1,1,1,1]); p.R = 0.01;  p.S = diag([1,1,1,1]);      % weighting matrices
S = reshape(p.S,p.n^2,1);                                       % matrix -> vector
[tsim,Psim] = ode45(@riccati_ode,vec.t(end:-1:1),S,[],vec,p);   % Riccati: backward integration
vec.P = Psim(end:-1:1,:)';                                      % -----------------------------

for i=1:length(vec.t),                                          % time-varying feedback gain
  [A,B] = matrices(vec.t(i),vec,p);                             % --------------------------
  P = reshape(vec.P(:,i),p.n,p.n);
  vec.k(:,i) = inv(p.R)*B'*P;
end
                                                                % simulation with initial error
X0 = [0;0;-pi+10*pi/180;2];                                     % -----------------------------
[tsim,x_ol] = ode45(@pendulum_sim,vec.t,X0,[],vec,p, 0);        % open-loop
[tsim,x_cl] = ode45(@pendulum_sim,vec.t,X0,[],vec,p, 1);        % closed-loop
vec.x_ol = x_ol'; vec.x_cl = x_cl';

for i=1:p.steps,                                                % computation of feedback part
  vec.du(i) = vec.k(:,i)'*(vec.xref(:,i)-vec.x_cl(:,i));
end

function F = pendulum_sim(t,x,vec,p,control)
% ------------------------------------------------------------
y=x(1:2); phi=x(3:4);
uref = interp1(vec.t,vec.uref,t);                               % feedforward control
if control==0,                                                  % simulation open-loop
  u = uref;
else                                                            % simulation closed-loop
  xref = interp1(vec.t,vec.xref',t)';
  k  = interp1(vec.t,vec.k',t);
  u  = uref + k*(xref-x);
end
F = [ y(2); u; phi(2); p.a1*sin(phi(1))-p.a2*cos(phi(1))*u ];   % nonlin. pendulum model

function Pp = riccati_ode(t,Pvec,vec,p)
% ------------------------------------------------------------
[A,B] = matrices(t,vec,p);                                      % time-varying matrices A and B
P     = reshape(Pvec,p.n,p.n);                                  % vector -> matrix
Pp    = -P*A - A'*P + P*B*inv(p.R)*B'*P - p.Q;                  % Riccati diff. eqn.
Pp    = reshape(Pp,p.n^2,1);                                    % matrix -> vector

function [A,B] = matrices(t,vec,p)
% ------------------------------------------------------------
phiref = interp1(vec.t,vec.xref(3:4,:)',t)';                    % current reference values
uref   = interp1(vec.t,vec.uref,t);
A = [0,1,0,0; 0,0,0,0; 0,0,0,1;                                 % time-varying matrices A and B
     0,0,p.a1*cos(phiref(1))+p.a2*sin(phiref(1))*uref,0];
B = [0; 1; 0; -p.a2*cos(phiref(1))];
```

Figure 5.14: MATLAB code to compute the optimal feedback law (5.88) and for simulating the pendulum swing-up.

It is also assumed that

- the matrices $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{Q}$, and $\boldsymbol{R}$ in (5.74) are time-invariant,
- $\boldsymbol{Q}$ is positive semidefinite and $\boldsymbol{R}$ is positive definite,
- the system (5.74b) is completely controllable,
- the pair $[\boldsymbol{A}, \boldsymbol{C}]$ is completely observable, where the matrix $\boldsymbol{C}$ is defined by the decomposition [5] $\boldsymbol{C}^\mathsf{T}\boldsymbol{C} = \boldsymbol{Q}$ (e.g. via a Cholesky decomposition).

Under these assumptions, the solution of the Riccati differential equation (5.79) converges in backward time against a stationary positive definite solution, see e.g. [14, 15]. One option to compute the stationary solution $\boldsymbol{P}$ is to integrate (5.79) backward in time starting from $\boldsymbol{P}(t_f) = \boldsymbol{0}$ for a sufficiently large t_f until the variations of $\boldsymbol{P}(t)$ lie below a certain tolerance.

A more elegant method for computing $\boldsymbol{P}$ is the solution of the *algebraic Riccati equation*

$$\boldsymbol{P}\boldsymbol{A} + \boldsymbol{A}^\mathsf{T}\boldsymbol{P} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T}\boldsymbol{P} + \boldsymbol{Q} = \boldsymbol{0} \tag{5.90}$$

that directly follows from the stationary consideration of (5.79). The Riccati equation is nonlinear, which implies that multiple solutions can exist. Under the given assumptions, however, there is only *one positive definite* solution $\boldsymbol{P}$. There exist various methods for solving the algebraic Riccati equation (5.90), e.g. those due to *Kalman* or *Englar*, that are not detailed in the following.

The principle of optimality (Theorem 5.4) can be used to show that the time-invariant feedback law

$$\boldsymbol{u}(t) = -\bar{\boldsymbol{K}}\boldsymbol{x}(t) \quad \text{with} \quad \bar{\boldsymbol{K}} = \boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T}\bar{\boldsymbol{P}}$$

is the optimal solution to the problem (5.74) on the infinite horizon $[t_0, \infty)$. This important result is summarized in the following theorem.

Theorem 5.5 (LQ control for linear time-invariant systems) *Consider the completely controllable linear time-invariant system*

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}\,, \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0$$

and the cost functional

$$J(\boldsymbol{u}) = \frac{1}{2}\int_{t_0}^{\infty} \boldsymbol{x}(t)^\mathsf{T}\boldsymbol{Q}\,\boldsymbol{x}(t) + \boldsymbol{u}(t)^\mathsf{T}\boldsymbol{R}\,\boldsymbol{u}(t)\,\mathrm{d}t \tag{5.91}$$

with $\boldsymbol{Q}$ being positive semidefinite and $\boldsymbol{R}$ positive definite. Moreover, suppose that the pair $[\boldsymbol{A}, \boldsymbol{C}]$ with $\boldsymbol{Q} = \boldsymbol{C}^\mathsf{T}\boldsymbol{C}$ is completely observable. Then, the minimum of (5.91) is attained by the optimal feedback law

$$\boldsymbol{u}(t) = -\bar{\boldsymbol{K}}\boldsymbol{x}(t) \quad \textit{with} \quad \bar{\boldsymbol{K}} = \boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T}\bar{\boldsymbol{P}}\,, \tag{5.92}$$

where $\boldsymbol{P}$ is the unique positive definite solution of the algebraic Riccati equation (5.90). Moreover, all eigenvalues of the matrix $(\boldsymbol{A} - \boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T}\boldsymbol{P})$ have negative real part such that the closed-loop system

$$\dot{\boldsymbol{x}} = \left(\boldsymbol{A} - \boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T}\boldsymbol{P}\right)\boldsymbol{x}\,, \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0$$

is asymptotically stable.

[5] An illustrative explanation of the last two assumptions can be found in [15].

The optimal feedback law (5.92) is often called *Riccati controller*, *LQ controller* or simply *LQR* ("linear quadratic regulator") and is often used in practice. In contrast to the concept of pole placement or eigenvalue assignment, the optimal controller is designed by choosing the weighting matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$, usually in diagonal form, which allows a straightforward interpretation for the single states and control variables. In MATLAB, the optimal feedback gain $\boldsymbol{K} = \boldsymbol{R}^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{P}$ can be computed with the functions lqr or lqrd/dlqr in the discrete-time case.

5.5 Optimal control problems with input constraints

For the previous considerations and the calculus of variations in general, we assumed that the optimal control problem (5.37) is unconstrained. In the following, we extend these results to the case of input constraints

$$\boldsymbol{u}(t) \in U \subseteq \mathbb{R}^m \quad \forall t \in [t_0, t_f]\,. \tag{5.93}$$

Input constraints often occur in practice and can be incorporated into the structure of the optimality conditions by means of *Pontryagin's maximum principle*. State constraints are significantly harder to account for and are therefore not considered at this stage.

5.5.1 Pontryagin's maximum principle

The maximum principle was postulated by *L.S. Pontryagin* in 1956 and was proven by him and his colleagues *Boltjanski*, *Gamkrelidse*, and *Mischenko* in the following years for more and more general system classes.[6] Similar to the calculus of variations, the maximum principle is typically formulated in terms of the Hamiltonian function and the canonical equations (5.46). However, the stationary condition $H_{\boldsymbol{u}} = \boldsymbol{0}$ is not valid anymore due to the constraints (5.93).

To illustrate this aspect, Figure 5.15 plots exemplary profiles of the Hamiltonian $H(\boldsymbol{x}^*, u, \boldsymbol{\lambda}^*, t)$ for a scalar control variable u and some fixed (optimal) trajectories $\boldsymbol{x}^*(t)$, $\boldsymbol{\lambda}^*(t)$ evaluated at a specific time point t. In the unconstrained case a), the conditions (5.46c) and (5.49), i.e. $H_u = 0$ and positive semidefiniteness of H_{uu}, hold for a minimum of the Hamiltonian at every time point t. However, if u is constrained to an interval $U = [u^-, u^+]$, the minimum can lie on the boundary u^- or u^+, where $H_u = 0$ is in general not satisfied as the cases b) and c) in Figure 5.15 illustrate.

The discussion shows that it is reasonable to replace the stationarity condition (5.46c) for the Hamiltonian by its minimization

$$\boxed{H(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*, t) = \min_{\boldsymbol{u} \in U} H(\boldsymbol{x}^*, \boldsymbol{u}, \boldsymbol{\lambda}^*, t)\,.}$$

This condition forms the basis of Pontryagin's maximum principle that is summarized in the following theorem for time-invariant problems.

[6] The name "maximum principle" is due to the fact that it was originally formulated for maximization problems. In this light, we should rather speak of minimum principle as it is also done in parts of the literature. However, to credit its importance in dynamic optimization and the contributions of *L.S. Pontryagin* in general, we stay with the original name "maximum principle".

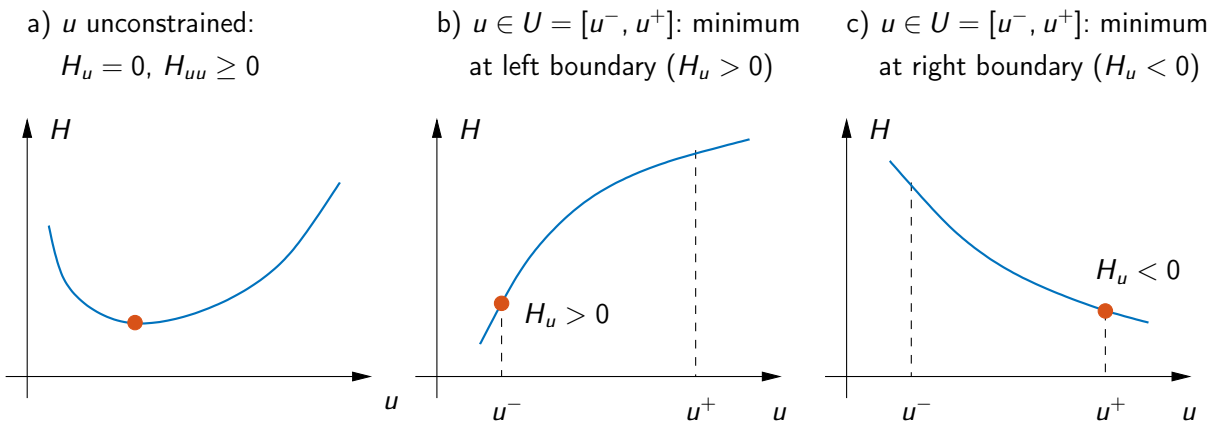


Figure 5.15: Minimum of the Hamiltonian in the unconstrained and constrained case.

Theorem 5.6 (Maximum principle for time-invariant systems) *Consider the optimal control problem*

$$\min_{\boldsymbol{u}(\cdot)} \quad J(\boldsymbol{u}) = \int_{t_0}^{t_f} l(\boldsymbol{x}(t), \boldsymbol{u}(t))\,\mathrm{d}t \tag{5.94a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \tag{5.94b}$$

$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0\,, \quad \boldsymbol{x}(t_f) = \boldsymbol{x}_f \tag{5.94c}$$

$$\boldsymbol{u}(t) \in U\,, \quad \forall t \in [t_0, t_f] \tag{5.94d}$$

with the free end time t_f. Suppose that the functions l and $\boldsymbol{f}$ are continuous in $(\boldsymbol{x}, \boldsymbol{u})$ and continuously differentiable in $\boldsymbol{x}$. If the problem (5.94) has an optimal solution $\boldsymbol{x}^*(t)$, $\boldsymbol{u}^*(t), t \in [t_0, t_f^*]$, then there exists a non-negative constant $\lambda_0 \geq 0$ and a non-vanishing (piecewise continuously differentiable) time function $\boldsymbol{\lambda}^* \in C_n^1[t_0, t_f^*]$ with $\boldsymbol{\lambda}^* = [\lambda_1^*, \ldots, \lambda_n^*]^\mathrm{T}$ such that

a) $\boldsymbol{x}^*(t)$ and $\boldsymbol{\lambda}^*(t)$, $t \in [t_0, t_f^*]$ satisfy the canonical equations

$$\dot{\boldsymbol{x}}^* = \boldsymbol{f}(\boldsymbol{x}^*, \boldsymbol{u}^*)\,, \quad \dot{\boldsymbol{\lambda}}^* = -H_{\boldsymbol{x}}(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*) \tag{5.95}$$

with the boundary conditions (5.94c) and the Hamiltonian

$$H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, \lambda_0) = \lambda_0\, l(\boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{\lambda}^\mathrm{T} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})\,. \tag{5.96}$$

b) $H(\boldsymbol{x}^*(t), \boldsymbol{u}, \boldsymbol{\lambda}^*(t), \lambda_0^*)$ as a function of $\boldsymbol{u}$ has a global minimum at the point $\boldsymbol{u} = \boldsymbol{u}^*(t) \in U$ for all $t \in [t_0, t_f^*]$, i.e.

$$H(\boldsymbol{x}^*(t), \boldsymbol{u}^*(t), \boldsymbol{\lambda}^*(t), \lambda_0^*) = \min_{\boldsymbol{u} \in U} H(\boldsymbol{x}^*(t), \boldsymbol{u}, \boldsymbol{\lambda}^*(t), \lambda_0^*) \quad \forall t \in [t_0, t_f^*]\,. \tag{5.97}$$

c) $H(\boldsymbol{x}^*(t), \boldsymbol{u}^*(t), \boldsymbol{\lambda}^*(t), \lambda_0^*)$ is constant for all $t \in [t_0, t_f^*]$. If the end time t_f^* is free, then the transversality condition

$$H(\boldsymbol{x}^*(t_f^*), \boldsymbol{u}^*(t_f^*), \boldsymbol{\lambda}^*(t_f^*), \lambda_0^*) = 0 \tag{5.98}$$

is additionally satisfied.

In analogy to the unconstrained case, the maximum principle can be extended to problems with partial or general terminal conditions by adapting the boundary conditions in (5.94c) by (5.47b), (5.47c) or (5.70), (5.72), respectively.

In the time-varying case, i.e. $l = l(\boldsymbol{x}, \boldsymbol{u}, t)$ and $\boldsymbol{f} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t)$, the Hamiltonian is no longer constant along the optimal solution (statement in point c) of Theorem 5.6) but satisfies the differential equation (5.50)

$$\frac{\mathrm{d}}{\mathrm{d}t} H = \lambda_0^* \, l_t(\boldsymbol{x}^*(t), \boldsymbol{u}^*(t), t) + \boldsymbol{\lambda}^*(t)^\mathsf{T} \boldsymbol{f}_t(\boldsymbol{x}^*(t), \boldsymbol{u}^*(t), t) \quad \forall t \in [t_0, t_f^*]$$

with $l_t = \frac{\partial l}{\partial t}$ and $\boldsymbol{f}_t = \frac{\partial \boldsymbol{f}}{\partial t}$.

The additional parameter λ_0 in the Hamiltonian (5.96) accounts for *abnormal problems* for which $\lambda_0^* = 0$ holds. This case occurs, for instance, if the optimal solution is independent of the integrated cost function l in (5.94a). An abnormal problem is often ill-posed in the sense that its solution does not affect the value of the cost functional. For well-posed problems, λ_0^* is typically set to $\lambda_0^* = 1$ such that the same Hamiltonian results as in the unconstrained case.

Example 5.5 (Abnormal case) *We consider the problem*

$$\begin{aligned} \min_{u(\cdot)} \quad & \int_0^1 l(x, u) \mathrm{d}t \\ \text{s.t.} \quad & \dot{x} = u\,, \quad x(0) = 0\,, \quad x(1) = 1 \\ & u(t) \in [0, 1]\,. \end{aligned}$$

There exists only one control $u^*(t) = 1$ that steers the state $x^*(t) = t$ from $x^*(0) = 0$ to $x^*(1) = 1$. This shows that the optimal solution is independent of the choice of the integral cost $l(x, u)$.

5.5.2 General procedure

The application of the maximum principle is similar to the unconstrained case in Section 5.3.4. [7]

1. Set-up of the Hamiltonian with $\lambda_0 = 1$: $H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}) = l(\boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{\lambda}^\mathsf{T} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$.
2. The solution of the minimization problem $\min_{\boldsymbol{u} \in U} H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda})$ for each potential combination $(\boldsymbol{x}, \boldsymbol{\lambda})$ defines the optimal control function

$$\boldsymbol{u} = \boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{\lambda}) = \arg\min_{\boldsymbol{u} \in U} H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}) \tag{5.99}$$

   in dependence of $(\boldsymbol{x}, \boldsymbol{\lambda})$. The construction of (5.99) is illustrated in Section 5.5.3 for selected cases.
3. Inserting the function (5.99) into the canonical equations (5.95) leads to the boundary value problem

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{\lambda}))\,, \quad \dot{\boldsymbol{\lambda}} = -H_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{\lambda}), \boldsymbol{\lambda})\,, \quad \text{BCs (5.94}c\text{)}.$$

[7] In analogy to Theorem 5.6, we consider the time-invariant case without loss of generality.

In case of partial or general nonlinear terminal conditions, $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$ and $\boldsymbol{x}(t_f) = \boldsymbol{x}_f$ in (5.94c) are replaced by (5.47b), (5.47c) or (5.70), (5.72), respectively. If the end time is free, then (5.98) extends the boundary conditions (BCs).

4. The (numerical) solution of the boundary value problem consists of $\boldsymbol{x}^*(t)$, $\boldsymbol{\lambda}^*(t)$. The optimal control trajectory $\boldsymbol{u}^*(t)$ is obtained afterwards from evaluating (5.99).

Similar to the unconstrained optimality conditions, the maximum principle only defines necessary conditions for an optimal solution. For certain problem classes, however, the maximum principle is also sufficient, in particular for the time-optimal control of time-invariant linear systems.

5.5.3 Minimization of the Hamiltonian

The solution of the minimization problem (5.97) is illustrated for selected relevant cases. It is assumed that the nonlinear system (5.94b) is given in the input-affine form

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{f}_0(\boldsymbol{x}) + \sum_{i=1}^{m} \boldsymbol{f}_i(\boldsymbol{x}) u_i \,. \tag{5.100}$$

Moreover, we consider the common case that all components u_i of the control vector $\boldsymbol{u} = [u_1, \dots, u_m]^\mathrm{T}$ are box-constrained, i.e.

$$\boldsymbol{u} \in U = [\boldsymbol{u}^-, \boldsymbol{u}^+] \quad \text{or} \quad u_i \in [u_i^-, u_i^+] \,, \quad i = 1, \dots, m \,, \tag{5.101}$$

and derive the optimal control function (5.99) component-wise with $\boldsymbol{\psi} = [\psi_1, \dots, \psi_m]^\mathrm{T}$.

Cost functional with consumption term

We consider the particular form of the cost functional (5.94a)

$$J(\boldsymbol{u}) = \int_{t_0}^{t_f} l_0(\boldsymbol{x}) + \sum_{i=1}^{m} r_i |u_i| \, \mathrm{d}t \,, \quad r_i > 0 \,. \tag{5.102}$$

If the state-dependent term $l_0(\boldsymbol{x})$ is zero, the term $\sum_{i=1}^{m} r_i |u_i|$ corresponds to the *control surface* over time or the *consumption* of the optimal control problem. For $l_0(\boldsymbol{x}) \neq 0$, a trade-off is achieved between both terms depending on the coefficients $r_i > 0$.

The Hamiltonian for the problem is defined as

$$H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}) = l_0(\boldsymbol{x}) + \sum_{i=1}^{m} r_i |u_i| + \boldsymbol{\lambda}^\mathrm{T} \left(\boldsymbol{f}_0(\boldsymbol{x}) + \sum_{i=1}^{m} \boldsymbol{f}_i(\boldsymbol{x}) u_i \right) .$$

We can omit the term $l_0(\boldsymbol{x}) + \boldsymbol{\lambda}^\mathrm{T} \boldsymbol{f}_0(\boldsymbol{x})$ in the minimization problem $\min_{\boldsymbol{u} \in U} H$ as it is independent of the control $\boldsymbol{u}$.[8] In addition, the elements u_i of the control vector $\boldsymbol{u} = [u_1, \dots, u_m]^\mathrm{T}$ enter linearly in H which allows one to consider the minimization of H separately for each u_i, i.e.

$$\min_{u_i \in [u_i^-, u_i^+]} H_i(u_i) = r_i |u_i| + q_i(\boldsymbol{x}, \boldsymbol{\lambda}) u_i \quad \text{with} \quad q_i(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{\lambda}^\mathrm{T} \boldsymbol{f}_i(\boldsymbol{x}) \,. \tag{5.103}$$

[8] It is important to note that for the Hamiltonian minimization, $H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda})$ is understood as a function of $\boldsymbol{u}$ with $\boldsymbol{x}$, $\boldsymbol{\lambda}$ being fixed (more accurately: for the *optimal* $\boldsymbol{x}^*(t)$, $\boldsymbol{\lambda}^*(t)$ at each time point t) as formulated in the maximum principle (Theorem 5.6).

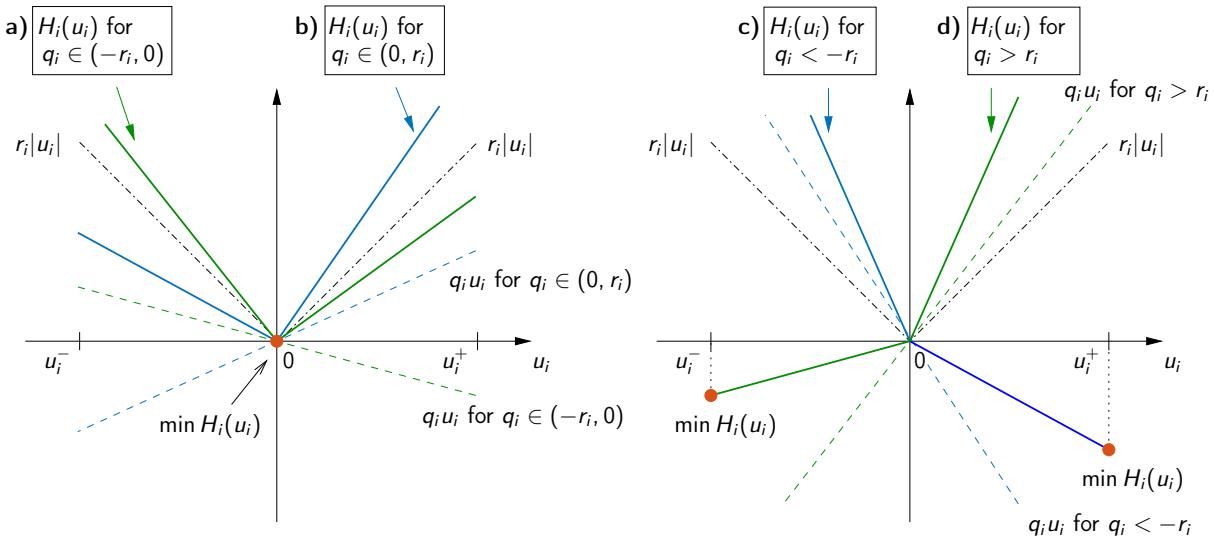


Figure 5.16: Illustration of (5.103) for the consumption optimal case.

The prefactor $q_i(\boldsymbol{x}, \boldsymbol{\lambda})$ plays an important role for solving the problem. Figure 5.16 illustrates the different cases a) to d) that are captured in the case-dependent and component-wise definition of the optimal control function (5.99)

$$u_i = \psi_i(\boldsymbol{x}, \boldsymbol{\lambda}) = \begin{cases} u_i^- & \text{if } \; q_i(\boldsymbol{x}, \boldsymbol{\lambda}) \geq r_i \\ 0 & \text{if } \; q_i(\boldsymbol{x}, \boldsymbol{\lambda}) \in (-r_i, r_i)\,, \\ u_i^+ & \text{if } \; q_i(\boldsymbol{x}, \boldsymbol{\lambda}) \leq -r_i \end{cases} \quad i = 1, \ldots, m\,. \tag{5.104}$$

Note that the control is piecewise continuous and not uniquely defined at the switching points $q_i(\boldsymbol{x}, \boldsymbol{\lambda}) = \pm r_i$. In practice, however, this is typically not an issue as the switching points are usually crossed at isolated time points.

Cost functional with energy term

The partial consideration of energy optimality (in the sense of the energy of the control signal $\boldsymbol{u}(t)$) in the cost functional (5.94a) leads to the formulation

$$J(\boldsymbol{u}) = \int_{t_0}^{t_f} l_0(\boldsymbol{x}) + \tfrac{1}{2} \sum_{i=1}^{m} r_i u_i^2 \, \mathrm{d}t\,, \quad r_i > 0\,. \tag{5.105}$$

Similar to the previous case, the minimization of the Hamiltonian can be handled element-wise by defining the functions

$$\min_{u_i \in [u_i^-, u_i^+]} H_i(u_i) = \tfrac{1}{2} r_i u_i^2 + q_i(\boldsymbol{x}, \boldsymbol{\lambda}) u_i \quad \text{with} \quad q_i(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{\lambda}^\mathsf{T} \boldsymbol{f}_i(\boldsymbol{x})\,. \tag{5.106}$$

Thanks to the quadratic term with $r_i > 0$, the function $H_i(u_i)$ always has an unconstrained minimum at the point

$$u_i^0 = -\frac{1}{r_i} q_i(\boldsymbol{x}, \boldsymbol{\lambda})\,,$$

which directly follows from the condition $\frac{\mathrm{d}H_i}{\mathrm{d}u_i} = r_i u_i + q_i(\boldsymbol{x}, \boldsymbol{\lambda}) = 0$. If u_i^0 lies inside the admissible interval $[u_i^-, u_i^+]$, then the optimal solution of (5.106) is defined by $u_i^* = u_i^0$. If

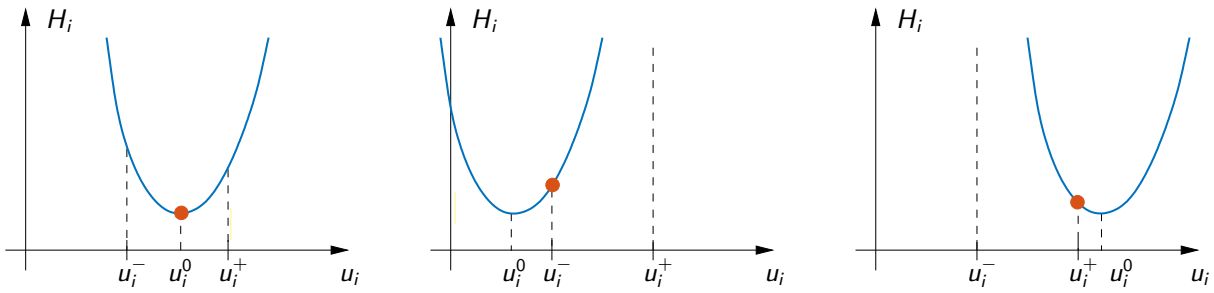


Figure 5.17: Illustration of (5.106) for the energy optimal case.

u_i^0 lies outside the interval $[u_i^-, u_i^+]$, the minimum of $H_i(u_i)$ lies either on the left or right boundary u_i^- or u_i^+, because $H_i(u_i)$ is strictly monotonically increasing (decreasing) for $u_i^0 < u_i^-$ ($u_i^0 > u_i^+$), see Figure 5.17. In summary, the optimal control function (5.99) is defined component-wise by

$$u_i = \psi_i(\boldsymbol{x}, \boldsymbol{\lambda}) = \begin{cases} u_i^- & \text{if } \; u_i^0 \leq u_i^- \\ u_i^0 & \text{if } \; u_i^0 \in (u_i^-, u_i^+)\,, \\ u_i^+ & \text{if } \; u_i^0 \geq u_i^+ \end{cases} \qquad i = 1, \ldots, m\,. \tag{5.107}$$

Note that u_i is continuous in contrast to the previous consumption optimal case.

Time optimal cost functional

For time optimal problems, the cost functional (5.94a) is defined by

$$J(\boldsymbol{u}) = \int_{t_0}^{t_f} 1 \, \mathrm{d}t = t_f - t_0 \tag{5.108}$$

with the respective Hamiltonian

$$H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}) = 1 + \boldsymbol{\lambda}^\mathsf{T} \left(\boldsymbol{f}_0(\boldsymbol{x}) + \sum_{i=1}^{m} \boldsymbol{f}_i(\boldsymbol{x}) u_i \right).$$

Again, the minimization problem (5.97) is treated component-wise

$$\min_{u_i \in [u_i^-, u_i^+]} H_i(u_i) = q_i(\boldsymbol{x}, \boldsymbol{\lambda}) u_i \quad \text{with} \quad q_i(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{\lambda}^\mathsf{T} \boldsymbol{f}_i(\boldsymbol{x})\,. \tag{5.109}$$

The problem is linear in u_i and the solution therefore depends on the sign of $q_i(\boldsymbol{x}, \boldsymbol{\lambda})$

$$u_i = \psi_i(\boldsymbol{x}, \boldsymbol{\lambda}) = \begin{cases} u_i^- & \text{if } \; q_i(\boldsymbol{x}, \boldsymbol{\lambda}) > 0 \\ u_i^+ & \text{if } \; q_i(\boldsymbol{x}, \boldsymbol{\lambda}) < 0 \end{cases}, \qquad i = 1, \ldots, m\,. \tag{5.110}$$

A critical case occurs if $q_i(\boldsymbol{x}(t), \boldsymbol{\lambda}(t))$ is zero on a subinterval $I_s \subseteq [t_0, t_f]$. In this case, the Hamiltonian H is independent of u_i and H is trivially minimal for every value of u_i. This *singular problem* was already described in Section 5.3.4 in the unconstrained case. In practice, the singularity issue is often avoided by adding a *regularization term*

$$J(\boldsymbol{u}) = \int_{t_0}^{t_f} 1 + \tfrac{1}{2} \sum_{i=1}^{m} r_i u_i^2 \, \mathrm{d}t$$

with sufficiently small coefficients $r_i > 0$. This formulation corresponds to the case (5.105) such that $u_i = \psi_i(\boldsymbol{x}, \boldsymbol{\lambda})$ can be computed with (5.107).

5.5.4 Example: Double integrator

A popular example to illustrate the maximum principle is the time optimal transition of a double integrating system to the origin $\boldsymbol{x}(t_f) = \mathbf{0}$. The optimal control problem with the state $\boldsymbol{x} = [x_1, x_2]^\mathrm{T}$ is formulated as

$$\min_{u(\cdot)} \quad \int_{t_0=0}^{t_f} 1 \,\mathrm{d}t = t_f \tag{5.111a}$$

$$\text{s.t.} \quad \dot{x}_1 = x_2, \quad \dot{x}_2 = u \tag{5.111b}$$

$$\boldsymbol{x}(0) = \boldsymbol{x}_0\,, \quad \boldsymbol{x}(t_f) = \mathbf{0} \tag{5.111c}$$

$$|u(t)| \leq 1 \quad \forall t \in [0, t_f]\,. \tag{5.111d}$$

With the Hamiltonian $H(\boldsymbol{x}, u, \boldsymbol{\lambda}) = 1 + \lambda_1 x_2 + \lambda_2 u$, the adjoint dynamics for $\boldsymbol{\lambda} = [\lambda_1, \lambda_2]^\mathrm{T}$ read

$$\dot{\lambda}_1 = 0 \qquad \Rightarrow \quad \lambda_1(t) = c_1 \tag{5.112a}$$

$$\dot{\lambda}_2 = -\lambda_1 \qquad \lambda_2(t) = -c_1 t + c_2 \tag{5.112b}$$

with the integration constants c_1 and c_2. The condition (5.97) for the Hamiltonian leads to the optimal control function (5.110)

$$u(t) = \begin{cases} +1 & \text{if } \lambda_2 < 0 \\ -1 & \text{if } \lambda_2 > 0\,. \end{cases} \tag{5.113}$$

The singular case for $\lambda_2(t) = 0$ on a non-vanishing subinterval I_s cannot occur due to the fact that $\lambda_2(t)$ represents a straight line, see (5.112b), and therefore $\lambda_2(t) = 0$ can only hold for an isolated time point t_s (switching between the limits ± 1) or for the whole time interval $[0, t_f]$. This, however, would contradict the transversality condition (5.98) for the free end time t_f

$$H(\boldsymbol{x}, u, \boldsymbol{\lambda})\Big|_{t=t_f} = 1 + \lambda_2(t_f)\, u(t_f) = 0\,.$$

In fact, it can be shown that the time-optimal control of linear time-invariant systems is always non-singular, see e.g. [15, 8].

Since $\lambda_2(t)$ is a straight line, there exists at most one switching event for the control in the time interval $[t_0, t_f]$ leading to the four possible control sequences $\{+1\}$, $\{-1\}$, $\{+1, -1\}$, $\{-1, +1\}$ for an optimal solution. This also implies that the trajectories in the (x_1, x_2)-plane are parabolas, which is addressed in the following exercise.

Exercise 5.4 *Show that the solution of the double integrator dynamics (5.111b) for $u(t) = \pm 1 = const.$ satisfies the parabola equation*

$$x_1 = \frac{x_2^2}{2u} + c\,, \quad u = \pm 1\,, \quad c \in \mathbb{R}\,.$$

The optimization goal is to reach the origin $\boldsymbol{x}(t_f) = \mathbf{0}$ as fast as possible from an arbitrary initial state $\boldsymbol{x}(0) = \boldsymbol{x}_0$. Figure 5.18 illustrates that the origin is only reachable via the parabola $x_1 = x_2^2/2$ for $u = 1$ or $x_1 = -x_2^2/2$ for $u = -1$, respectively. Both cases can be combined in the *switching curve*

$$S(x_2) = -\tfrac{1}{2} x_2 |x_2|\,.$$

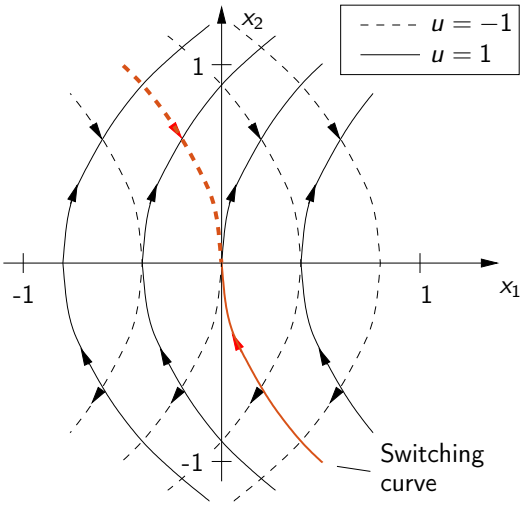


Figure 5.18: Trajectories of the double integrator in the (x_1, x_2)-plane.

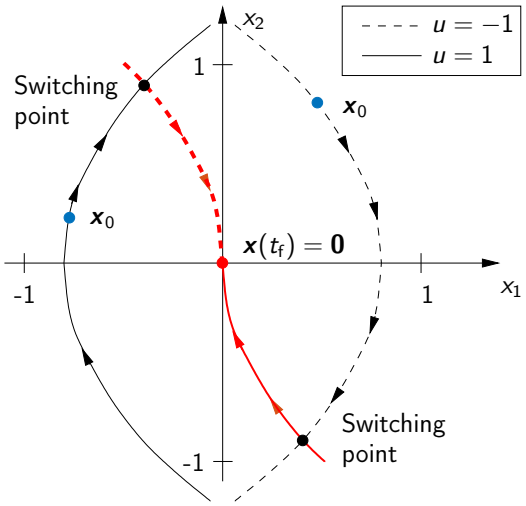


Figure 5.19: Optimal switching for different initial states $\boldsymbol{x}_0$.

In view of the fact that only four switching sequences $\{+1\}$, $\{-1\}$, $\{+1, -1\}$, $\{-1, +1\}$ are possible for u, there exists only one way to reach the origin $\boldsymbol{x}(t_f) = \mathbf{0}$ as fast as possible from $\boldsymbol{x}(0) = \boldsymbol{x}_0$.

- If $x_{1,0} = S(x_{2,0})$, no switching is required and $\boldsymbol{x}(t_f) = \mathbf{0}$ is directly reached via the switching curve with $u(t) = 1$ or $u(t) = -1$, see Figure 5.18.
- If $\boldsymbol{x}_0$ lies to the left or the right of the switching curve, i.e. $x_{1,0} < S(x_{2,0})$ or $x_{1,0} > S(x_{2,0})$, exactly one switching event is necessary to reach the curve $S(x_2)$ and follow it to the origin, see Figure 5.19.

These considerations lead to the optimal control law

$$u(t) = \begin{cases} +1 & \text{if } x_1 < S(x_2) \\ +1 & \text{if } x_1 = S(x_2) \text{ and } x_1 > 0 \\ -1 & \text{if } x_1 > S(x_2) \\ -1 & \text{if } x_1 = S(x_2) \text{ and } x_1 < 0\,. \end{cases} \tag{5.114}$$

Figure 5.20 shows the time optimal trajectories of the double integrator for various initial states $\boldsymbol{x}_0 = [x_{1,0}, x_{2,0}]^\mathrm{T}$.

Exercise 5.5 *Verify that the optimal switching point t_s and the end time t_f are case-dependently given by*

$$t_s = \begin{cases} x_{2,0} + \sqrt{\frac{1}{2}x_{2,0}^2 + x_{1,0}} & \text{if } x_{1,0} > S(x_{2,0}) \\ -x_{2,0} + \sqrt{\frac{1}{2}x_{2,0}^2 - x_{1,0}} & \text{if } x_{1,0} < S(x_{2,0}) \end{cases} \tag{5.115}$$

$$t_f = \begin{cases} x_{2,0} + \sqrt{2x_{2,0}^2 + 4x_{1,0}} & \text{if } x_{1,0} > S(x_{2,0}) \\ -x_{2,0} + \sqrt{2x_{2,0}^2 - 4x_{1,0}} & \text{if } x_{1,0} < S(x_{2,0}) \\ |x_{2,0}| & \text{if } x_{1,0} = S(x_{2,0})\,. \end{cases} \tag{5.116}$$

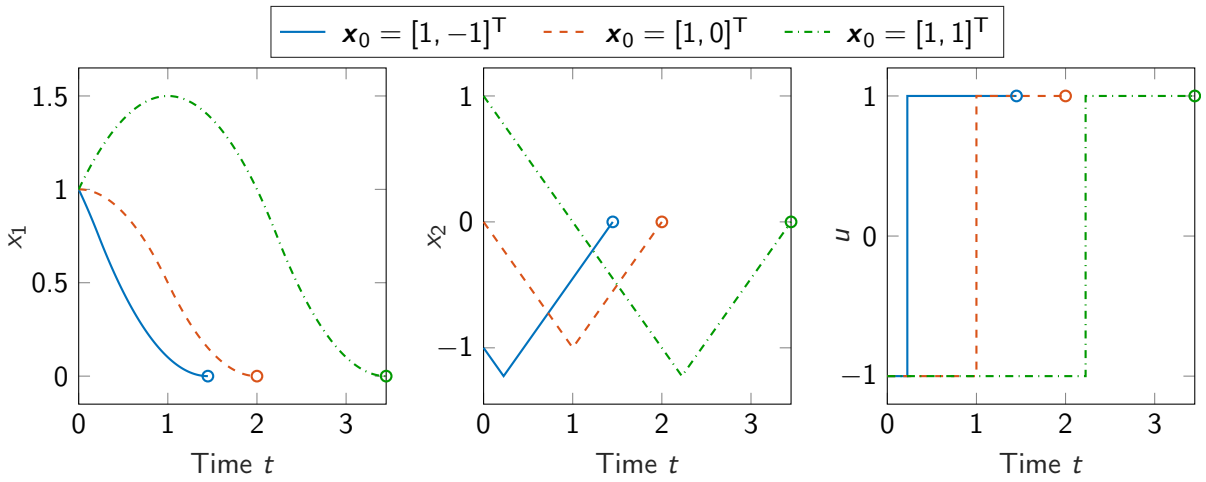


Figure 5.20: Time-optimal trajectories of the double integrator.

A compact way to express the case-dependent definition of the control law (5.114) is

$$u(t) = -\text{sign}\left(x_1 + \tfrac{1}{2}x_2|x_2|\right), \quad 0 = \text{sign}(0). \tag{5.117}$$

The sign function does not include the special case $x_1 = S(x_2)$ in (5.114), which means that the state $\boldsymbol{x}$ will always slightly cross the curve $S(x_2)$ before the switching occurs. This is not an issue from the numerical perspective because after the switching event the state $\boldsymbol{x}$ follows a parallel trajectory in close vicinity of the switching curve $S(x_2)$.

Exercise 5.6 *Implement the system (5.111b) with the optimal control law (5.117) in* MATLAB/SIMULINK *and verify the results in Figure 5.20 for different initial states. Use t_f from (5.116) as the simulation time in* MATLAB/SIMULINK.

5.5.5 State-dependent input constraints

The maximum principle (Theorem 5.6) can be extended to a more general class of input constraints

$$\boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \leq \boldsymbol{0} \quad \forall t \in [t_0, t_f] \tag{5.118}$$

with the state/input-dependent constraint function $\boldsymbol{h} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^q$. Note that the admissible set

$$U(\boldsymbol{x}) = \{\boldsymbol{u} \in \mathbb{R}^m : \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u}) \leq \boldsymbol{0}\} \tag{5.119}$$

now depends on the state $\boldsymbol{x}$, i.e. the control must satisfy $\boldsymbol{u}(t) \in U(\boldsymbol{x}(t))\ \forall t \in [t_0, t_f]$ with the associated state trajectory $\boldsymbol{x}(t)$. We have to assume that all elements of $\boldsymbol{h} = [h_1, \ldots, h_q]^T$ explicitly depend on $\boldsymbol{u}$ and that the constraint qualification [9]

$$\text{rank}\left[\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{u}}, \ \text{diag}(\boldsymbol{h})\right] = q \tag{5.120}$$

is satisfied. Similar to the procedure in the parameter optimization case (Chapter 4), the inequality constraints (5.118) are added to the Hamiltonian (5.96) with (time-varying) Lagrange

[9] The constraint qualification (5.120) implies that the gradients of all constraints $h_i(\boldsymbol{x}, \boldsymbol{u}) \leq 0$ that are active at point $(\boldsymbol{x}, \boldsymbol{u})$ are linearly independent.

multipliers $\boldsymbol{\mu} = [\mu_1, \dots, \mu_q]^\mathsf{T}$

$$H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, \lambda_0, \boldsymbol{\mu}) = \lambda_0\, l(\boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{\lambda}^\mathsf{T} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{\mu}^\mathsf{T} \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u})\,. \tag{5.121}$$

The conditions (5.95)-(5.98) of the maximum principle are adapted accordingly for the modified Hamiltonian (5.121) and the (state-dependent) admissible set (5.119). In addition, the *complementary and sign conditions*

$$\mu_i^*(t) h_i(\boldsymbol{x}^*(t), \boldsymbol{u}^*(t)) = 0\,, \quad \mu_i^*(t) \geq 0 \quad \forall t \in [t_0, t_f] \tag{5.122}$$

must be satisfied, similar to the KKT conditions (4.29d), (4.29e) in the parameter optimization case.

A significantly more complex case are pure state constraints

$$\boldsymbol{x}(t) \in X \subseteq \mathbb{R}^n \quad \text{or} \quad \boldsymbol{h}(\boldsymbol{x}(t)) \leq 0 \quad \forall t \in [t_0, t_f]$$

with $\boldsymbol{h} : \mathbb{R}^n \to \mathbb{R}^q$, which require a detailed look at *constrained and unconstrained arcs*, i.e. subintervals of $[t_0, t_f]$ where the constraints are active or inactive. Moreover, there are various ways of incorporating state constraints within the Hamiltonian function leading to different formulations of the optimality conditions. More information on this topic can be found in the survey [11] and the textbooks [5, 15].

5.6 Numerical solution of optimality conditions (indirect methods)

In most cases, dynamic optimization problems or optimal control problems must be solved by numerical methods. So-called *indirect methods* are based on the solution of the optimality conditions (5.46)-(5.48) or (5.95)-(5.98) in the input-constrained case. The most common methods for solving these kinds of two-point boundary value problems are *discretization*, *shooting* or the *gradient method* that are illustrated in the following.

5.6.1 Discretization method

The general optimality conditions (5.46)-(5.48) or (5.95)-(5.98) in the input-constrained case form a two-point boundary value problem for the states $(\boldsymbol{x}, \boldsymbol{\lambda})$ and the control $\boldsymbol{u}$. In the following, we assume that the optimal control can be expressed as the function $\boldsymbol{u} = \boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{\lambda})$ of the states $\boldsymbol{x}$ and $\boldsymbol{\lambda}$, which renders the canonical equations independent of $\boldsymbol{u}$ (see Section 5.3.4 and 5.5.2)

$$\left.\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{\lambda}), t) \\ \dot{\boldsymbol{\lambda}} &= -H_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{\lambda}), \boldsymbol{\lambda}, t) \end{aligned}\right\} \quad \dot{\bar{\boldsymbol{x}}} = \boldsymbol{F}(\bar{\boldsymbol{x}}, t)\,, \quad \bar{\boldsymbol{x}} = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{\lambda} \end{bmatrix}. \tag{5.123}$$

In what follows, we consider a fixed end time t_f and the initial and partial terminal conditions

$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0\,, \quad \boldsymbol{G}(\bar{\boldsymbol{x}}(t_f)) = \begin{bmatrix} \left[x_i(t_f) - x_{f,i}\right]_{i \in \mathcal{I}_f} \\ \left[\lambda_i(t_f) - V_{x_i}(\boldsymbol{x}(t_f), t_f)\right]_{i \notin \mathcal{I}_f} \end{bmatrix} = \boldsymbol{0} \tag{5.124}$$

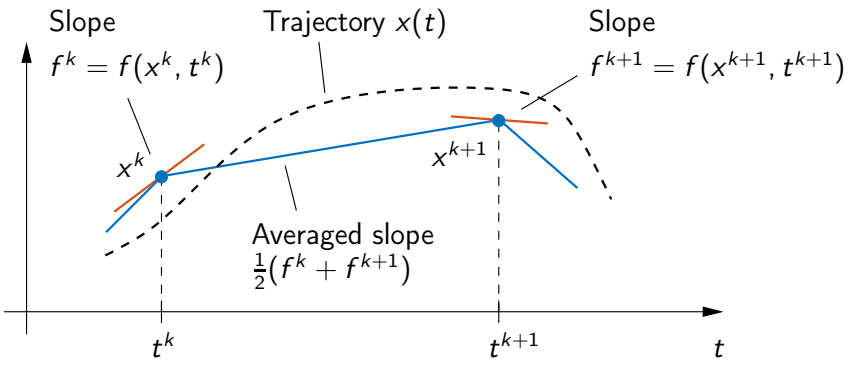


Figure 5.21: Illustration of the trapezoidal scheme.

with $\boldsymbol{G} : \mathbb{R}^{2n} \to \mathbb{R}^n$ such that $2n$ boundary conditions belong to the $2n$ differential equations. The handling of general terminal conditions and of a free end time t_f is adressed in Section 5.6.3.

A popular method for solving two-point boundary value problems is the discretization of the time interval $[t_0, t_f]$

$$t_0 = t^0 < t^1 < \cdots < t^N = t_f \,,$$

in order to approximate the solution at the $N+1$ discretized time points t^k, i.e.

$$\bar{\boldsymbol{x}}^k \approx \bar{\boldsymbol{x}}(t^k) \,, \quad k = 0, 1, \ldots, N \,. \tag{5.125}$$

A well known discretization formula is the *trapezoidal scheme*

$$\frac{\bar{\boldsymbol{x}}^{k+1} - \bar{\boldsymbol{x}}^k}{t^{k+1} - t^k} = \frac{1}{2}\Big[\boldsymbol{F}(\bar{\boldsymbol{x}}^k, t^k) + \boldsymbol{F}(\bar{\boldsymbol{x}}^{k+1}, t^{k+1})\Big] \,, \quad k = 0, 1, \ldots, N-1 \tag{5.126a}$$

that is illustrated in Figure 5.21. The discretizations (5.126) together with the $2n$ boundary conditions (5.124)

$$\boldsymbol{x}^0 = \boldsymbol{x}_0 \,, \quad \boldsymbol{G}(\bar{\boldsymbol{x}}^N) = \boldsymbol{0} \tag{5.126b}$$

form a nonlinear set of equations (5.126) of order $2n(N+1)$ for the $2n(N+1)$ variables (5.125).

A related method for solving two-point BVPs is *collocation* that uses basis functions (e.g. polynomials) to approximate the state $\bar{\boldsymbol{x}}(t)$ on each interval $[t^k, t^{k+1}]$. The consistency with the differential equations (5.123) is enforced at additional *collocation points* within the intervals $[t^k, t^{k+1}]$. Continuity over the intervals $[t^k, t^{k+1}]$ is ensured by further consistency constraints at the interval bounds t^k. Overall, this leads to a set of nonlinear equations for the coefficients of the basis functions. Note that the collocation method is equivalent to certain discretization formulas depending on the choice of basis functions and collocation points.

Some characteristic properties of discretization methods are listed in the following:

- Discretization methods are often numerically more robust than shooting (Section 5.6.2) due to the simultaneous consideration of the discretized dynamics and boundary conditions in (5.126) and the possibility to use implicit integration schemes such as the trapezoidal scheme.
- A good initial guess of the adjoint variables is important and often decides upon whether and how fast the method converges. This is particularly true for full terminal constraints

$\boldsymbol{x}(t_f) = \boldsymbol{x}_f$ with the consequence that the adjoint dynamics are not supported by any boundary conditions, see (5.123).

- The discretization method requires a high implementation effort. However, the Jacobian of the set of equations (5.126) is sparsely occupied and is block-structured, which can be effectively exploited by *sparse numerics*.
- The number of discretization points $N+1$ has a direct impact on the solution. Larger values of N enhance the accuracy but increase the overall dimension of the equations to be solved.
- The trapezoidal scheme is a second-order method, which means that the numerical solution $\bar{\boldsymbol{x}}^k \approx \bar{\boldsymbol{x}}(t^k)$ corresponds to the Taylor expansion of the exact solution up to the quadratic term. An alternative are e.g. higher-order Runge-Kutta schemes.

5.6.2 Shooting method

The discretization method constructs a high-dimensional set of equations for the differential equations (5.123) and boundary conditions (5.124). In contrast to this, the *shooting method* considers the initial value problem

$$\dot{\bar{\boldsymbol{x}}} = \boldsymbol{F}(\bar{\boldsymbol{x}}, t)\,, \quad \bar{\boldsymbol{x}}(t_0) = \begin{bmatrix} \boldsymbol{x}_0 \\ \boldsymbol{\lambda}_0 \end{bmatrix} \tag{5.127}$$

with the unknown initial values $\boldsymbol{\lambda}_0$ of the adjoint states $\boldsymbol{\lambda}$. The solution of (5.127) can formally be written is dependence of $\boldsymbol{\lambda}_0$

$$\bar{\boldsymbol{x}}(t; \boldsymbol{\lambda}_0) = \begin{bmatrix} \boldsymbol{x}_0 \\ \boldsymbol{\lambda}_0 \end{bmatrix} + \int_{t_0}^{t} \boldsymbol{F}(\bar{\boldsymbol{x}}(\tau; \boldsymbol{\lambda}_0), \tau)\, \mathrm{d}\tau\,. \tag{5.128}$$

The satisfaction of the terminal conditions (5.124) leads to the nonlinear equations

$$\boldsymbol{G}(\bar{\boldsymbol{x}}(t_f; \boldsymbol{\lambda}_0)) = \boldsymbol{0} \tag{5.129}$$

of order n for the initial adjoint values $\boldsymbol{\lambda}_0 \in \mathbb{R}^n$ of the same order.

The nonlinear equations (5.129) are in general solved iteratively, e.g. by means of Newton's method. To this end, the initial value problem (5.127) is numerically integrated to obtain the trajectory (5.128), which is also the reason for the name *shooting*.

Example 5.6 *To illustrate the shooting method, we consider the boundary value problem*

$$\dot{x}_1 = x_2\,, \quad \dot{x}_2 = f(x_1, x_2)\,, \quad x_1(t_0) = 0\,, \quad x_1(t_f) = 0\,.$$

If the problem is written as initial value problem, the solution $x_1(t; x_{2,0})$ depends on the initial value $x_2(t_0) = x_{2,0}$ of the second state, where the value $x_{2,0}$ corresponds to the initial slope of the trajectory $x_1(t; x_{2,0})$ that must be chosen such that the terminal condition $x_1(t_f; x_{2,0}) = 0$ is satisfied, see Figure 5.22.

Some important properties of the shooting method that are worth mentioning are:

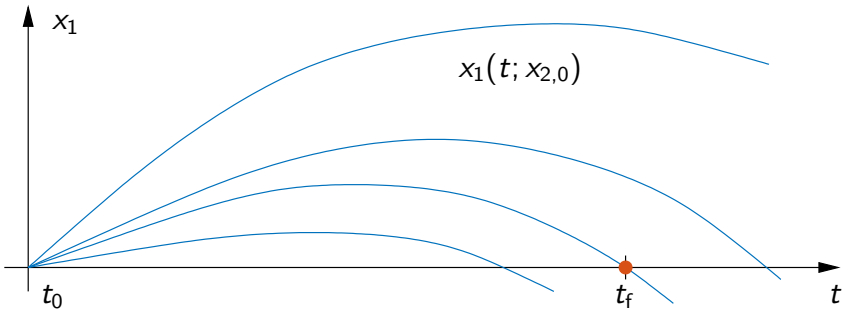


Figure 5.22: Illustration of the shooting method (Example 5.6).

- The implementation effort is low (at least compared to the discretization or the gradient method in Section 5.6.4) due to the sequential solution of the initial value problem (5.127) and the terminal constraints (5.129).
- Good initial values of $\boldsymbol{\lambda}_0$ are important as the shooting method is very sensitive w.r.t. $\boldsymbol{\lambda}_0$. This is also closely related to the next point.
- The integration of the initial value problem (5.127) can be numerically critical for a larger time interval $[t_0, t_f]$. This is due to the fact that the canonical equations contain unstable modes. [10]
- The numerical sensitivity issues can be resolved or at least mitigated by *multiple shooting*, where the interval $[t_0, t_f]$ is further divided in subintervals [1, 16].

5.6.3 Handling of general terminal constraints and free end time

In the previous sections, a fixed end time and partial terminal conditions were assumed. In case of general terminal constraints (5.70), the function $\boldsymbol{G}$ in (5.124) is replaced by

$$\boldsymbol{G}(\bar{\boldsymbol{x}}(t_f), \boldsymbol{\nu}) = \begin{bmatrix} \boldsymbol{g}(\boldsymbol{x}(t_f), t_f) \\ \boldsymbol{\lambda}(t_f) - V_{\boldsymbol{x}}(\boldsymbol{x}(t_f), t_f) - \boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{x}(t_f), t_f)^{\mathsf{T}} \boldsymbol{\nu} \end{bmatrix} = \boldsymbol{0} \tag{5.130}$$

with $\boldsymbol{G} : \mathbb{R}^{2n} \times \mathbb{R}^r \to \mathbb{R}^{n+r}$. The Lagrange multipliers $\boldsymbol{\nu} \in \mathbb{R}^r$ are additional parameters that account for the terminal conditions in the function $\boldsymbol{G}$.

If the end time t_f is unspecified, a time transformation

$$t = t_0 + \delta\tau \quad \text{with} \quad \tau \in [0, 1] \tag{5.131}$$

can be applied to transform the canonical equations to the normalized time interval $[0, 1]$ of the new time coordinate τ. The parameter δ is the stretching factor that determines the end time $t_f = t_0 + \delta$ (corresponding to $\tau = 1$). With the new states

$$\tilde{\boldsymbol{x}}(\tau) := \begin{bmatrix} \boldsymbol{x}(t_0 + \delta\tau) \\ \boldsymbol{\lambda}(t_0 + \delta\tau) \end{bmatrix}, \quad \tau \in [0, 1]$$

and the notation $(\cdot)' = \frac{\mathrm{d}}{\mathrm{d}\tau}(\cdot)$, the canonical equations (5.123) become

$$\tilde{\boldsymbol{x}}' = \delta \boldsymbol{F}(\tilde{\boldsymbol{x}}, t_0 + \delta\tau), \quad \tau \in (0, 1). \tag{5.132}$$

[10] In the linear case, the eigenvalues of the canonical equations, see (5.77), are symmetric to the imaginary axis.

In addition, the initial and terminal conditions in (5.124) or (5.130), respectively, must be expressed in terms of the new time coordinate $\tau \in [0, 1]$ and are extended by the transversality condition (5.48) or (5.73).

The time stretch factor δ and/or multipliers $\boldsymbol{\nu}$ are further unknown variables for the discretization or shooting method with the same number of additional equations in the boundary conditions $\boldsymbol{G}(\cdot) = \boldsymbol{0}$. The following table summarizes the different cases:

End time t_f	Terminal cond.	Add. parameters $\delta \in \mathbb{R}$	$\boldsymbol{\nu} \in \mathbb{R}^r$	Number of initial/terminal cond. $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$	$\boldsymbol{G}(\cdot) = \boldsymbol{0}$
fixed	partial	-	-	n	n
fixed	general	-	✓	n	$n + r$
free	partial	✓	-	n	$n + 1$
free	general	✓	✓	n	$n + r + 1$

5.6.4 Gradient method

Another well known method for the solution of optimal control problems is the *gradient method* that follows the same spirit as the gradient method in the context of parameter optimization (see Section 3.3.2). The gradient method is mainly used for optimal control problems with fixed end time and free terminal state

$$\min_{\boldsymbol{u}(\cdot)} \quad J(\boldsymbol{u}) = V(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} l(\boldsymbol{x}(t), \boldsymbol{u}(t), t)\,\mathrm{d}t \tag{5.133a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t), \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0\,. \tag{5.133b}$$

The essential question in this regard is how the gradient, i.e. the direction of steepest descent for the functional $J(\boldsymbol{u})$ can be computed.

Theorem 5.7 (Gâteaux derivative of $J(\boldsymbol{u})$) *Suppose that the functional $J(\boldsymbol{u})$ is Gâteaux differentiable at point $\boldsymbol{u} \in \mathcal{C}_m^0[t_0, t_f]$ and let $\boldsymbol{x}, \boldsymbol{\lambda} \in \mathcal{C}_n^1[t_0, t_f]$ be the solution of the canonical equations*

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t)\,, \qquad \boldsymbol{x}(t_0) = \boldsymbol{x}_0 \tag{5.134a}$$

$$\dot{\boldsymbol{\lambda}} = -H_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, t)\,, \quad \boldsymbol{\lambda}(t_f) = V_{\boldsymbol{x}}(\boldsymbol{x}(t_f))\,. \tag{5.134b}$$

Then, the Gâteaux derivative in direction of $\boldsymbol{v} \in \mathcal{C}_m^0[t_0, t_f]$ is defined by

$$\delta J(\boldsymbol{u}; \boldsymbol{v}) = \int_{t_0}^{t_f} H_{\boldsymbol{u}}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t), t)^{\mathsf{T}} \boldsymbol{v}(t)\,\mathrm{d}t\,. \tag{5.135}$$

Exercise 5.7 (optional) *Prove Theorem 5.7 by considering the control trajectory $\boldsymbol{w}^\varepsilon(t) = \boldsymbol{u}(t) + \varepsilon \boldsymbol{v}(t)$, $t \in [t_0, t_f]$ with the associated state trajectory $\boldsymbol{y}^\varepsilon(t)$ and $\boldsymbol{x}(t) = \boldsymbol{y}^0(t)$ for $\varepsilon = 0$. Derive $\delta J(\boldsymbol{u}, \boldsymbol{v}) = \frac{\mathrm{d}J(\boldsymbol{w}^\varepsilon)}{\mathrm{d}\varepsilon}\Big|_{\varepsilon=0}$ by augmenting $J(\boldsymbol{w}^\varepsilon)$ with the integrand $\boldsymbol{\lambda}(t)^{\mathsf{T}}(\boldsymbol{f}(\boldsymbol{y}^\varepsilon(t), \boldsymbol{w}^\varepsilon(t), t) - \dot{\boldsymbol{y}}^\varepsilon(t)) = 0$.*

Initialization:	
$j \leftarrow 0$	Iteration counter
ε	Convergence tolerance
$\boldsymbol{u}^0(t)$	Initial control trajectory
$\boldsymbol{x}^0(t)$	Integration of (5.134a) in forward time
repeat	
$\boldsymbol{\lambda}^j(t)$	Integration of (5.134b) in backward time
$\boldsymbol{g}^j(t) \leftarrow$ (5.136)	Computation of gradient
$\alpha^j \leftarrow \arg\min_{\alpha>0} J(\boldsymbol{u}^j - \alpha \boldsymbol{g}^j)$	Line search (often approximate solution)
$\boldsymbol{u}^{j+1}(t) = \boldsymbol{u}^j(t) - \alpha^j \boldsymbol{g}^j(t)$	New control trajectory with step size α^j
$\boldsymbol{x}^{j+1}(t)$	Integration of (5.134a) in forward time
$j \leftarrow j+1$	
until $\lvert J(\boldsymbol{u}^j) - J(\boldsymbol{u}^{j-1})\rvert \leq \varepsilon$	Convergence criterion (or similar one)

Table 5.1: Algorithm of the gradient method.

If we define the gradient as the partial derivative of the Hamiltonian w.r.t. $\boldsymbol{u}$

$$\boldsymbol{g}(t) = H_{\boldsymbol{u}}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t), t)\,, \quad t \in [t_0, t_f]\,, \tag{5.136}$$

then the Gâteaux derivative (5.135) becomes $\delta J(\boldsymbol{u}, -\boldsymbol{g}) < 0$ and $-\boldsymbol{g}$ is the steepest descent direction at the point $\boldsymbol{u} \in C_m^0[t_0, t_f]$. Given the current control trajectory $\boldsymbol{u}^j(t)$, $t \in [t_0, t_f]$ in iteration j, the gradient method exploits the decoupling of the boundary conditions for $\boldsymbol{x}(t_0)$ and $\boldsymbol{\lambda}(t_f)$ and sequentially solves the system and adjoint dynamics (5.134) in forward and backward time to compute $\boldsymbol{x}^j(t)$ and $\boldsymbol{\lambda}^j(t)$. Afterwards, the negative gradient $-\boldsymbol{g}^j(t)$ is used to achieve a decrease in the cost functional (5.133a). The basic algorithm of the gradient method is summarized in Table 5.1.

The underlying line search problem is usually solved approximately, in its simplest form based on the Armijo condition (see Section 3.3.1). Some variants of the gradient method use a constant step size α leading to the control update $\boldsymbol{u}^{j+1}(t) = \boldsymbol{u}^j(t) - \alpha \boldsymbol{g}^j(t)$ in all iterations. In this case, however, α must be chosen sufficiently small which can lead to a slow convergence speed. A good trade-off between accuracy and computation effort can be achieved with an explicit or adaptive line search strategy. In particular, an explicit step size can be computed from minimizing the difference between two control updates $\boldsymbol{u}^{j+1}(t)$ and $\boldsymbol{u}^j(t)$. An alternative is to approximately solve the line search problem on an interval $[\alpha^-, \alpha^+]$ that is dynamically adapted over the iterations. More details on these strategies can be found in [7].

Some important properties of the gradient method are listed in the following. More information can e.g. be found in [17, 5, 15].

- Input constraints $\boldsymbol{u} \in U$ can be considered via *projection* onto the admissible set U.
- An advantage of the gradient method in particular in case of general nonlinear problems is that the control $\boldsymbol{u}$ is not explicitly computed from $H_{\boldsymbol{u}} = \boldsymbol{0}$ or $\min_{\boldsymbol{u}\in U} H$ (see Sections 5.3.4 and 5.5.2).

- The gradient method is numerically more robust than shooting due to the fact that the adjoint system (5.134b) is not integrated in the unstable forward direction. A further advantage over shooting and also discretization methods is that no initial guess of the adjoint trajectory is required.
- The gradient method typically leads to a fast cost decrease in the first iterations but tends to converge slowly in the vicinity of the optimal solution.
- An alternative are *conjugate gradient methods* or *second-order gradient methods* with better convergence speed towards the optimal solution.
- The gradient method can be extended to handle partial or general terminal conditions and a free end time. This, however, impacts the robustness and convergence properties.

5.7 Numerical solution by discretization (direct methods)

Direct methods parametrize the control trajectory $\boldsymbol{u}(t)$, $t \in [t_0, t_f]$ in order to reduce the optimal control problem to a parameter optimization problem that can be solved with the methods from unconstrained and constrained optimization (see Chapter 3 and 4). The following table lists some advantages of direct/indirect methods.

Direct methods:

- Derivation of canonical equations is not necessary
- Handling of state constraints is easier
- Usually larger convergence domain
- Powerful algorithms available

Indirect methods:

- Give insight into the structure of the optimal solution
- Highly accurate solution (important e.g. in aerospace applications)
- Adjoint variables can be used for sensitivity analysis and feedback control ("*neighbouring extremal control*")

Direct methods rely on *partial* or *full discretization*, which is detailed in the following. To this end, we consider a general constrained optimal control problem of the form

$$\min_{\boldsymbol{u}(\cdot)} \quad J(\boldsymbol{u}) = V(\boldsymbol{x}(t_f)) + \int_{t_0}^{t_f} l(\boldsymbol{x}(t), \boldsymbol{u}(t), t)\,\mathrm{d}t \tag{5.137a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t)\,, \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0 \tag{5.137b}$$

$$\boldsymbol{g}(\boldsymbol{x}(t_f)) = \boldsymbol{0} \tag{5.137c}$$

$$\boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \leq \boldsymbol{0} \quad \forall t \in [t_0, t_f]\,. \tag{5.137d}$$

The end time t_f is supposed to be fixed for simplicity. A free end time t_f can be considered by the time transformation (5.131) in order to transform the problem to a new time coordinate $\tau \in [0, 1]$.

5.7.1 Partial discretization

Direct solution methods discretize the time interval $[t_0, t_f]$

$$t_0 = t^0 < t^1 < \cdots < t^N = t_f \tag{5.138}$$

and parametrize the control $\boldsymbol{u}$ on the single N subintervals $[t^k, t^{k+1}]$. In the simplest case, $\boldsymbol{u}$ is chosen constant on each subinterval

$$\boldsymbol{u}(t) = \boldsymbol{\psi}(t, \hat{\boldsymbol{u}}) = \boldsymbol{u}^k\,, \quad t \in [t^k, t^{k+1}) \quad \text{with} \quad \hat{\boldsymbol{u}} = \begin{bmatrix} \boldsymbol{u}^0 \\ \vdots \\ \boldsymbol{u}^{N-1} \end{bmatrix} \in \mathbb{R}^{mN}\,, \tag{5.139}$$

which results in a piecewise constant control trajectory $\boldsymbol{u}(t) = \boldsymbol{\psi}(t, \hat{\boldsymbol{u}})$ in dependence of the mN parameters $\hat{\boldsymbol{u}}$. An alternative is, for instance, a piecewise linear parametrization for $\boldsymbol{\psi}(t, \hat{\boldsymbol{u}})$, see Figure 5.23.

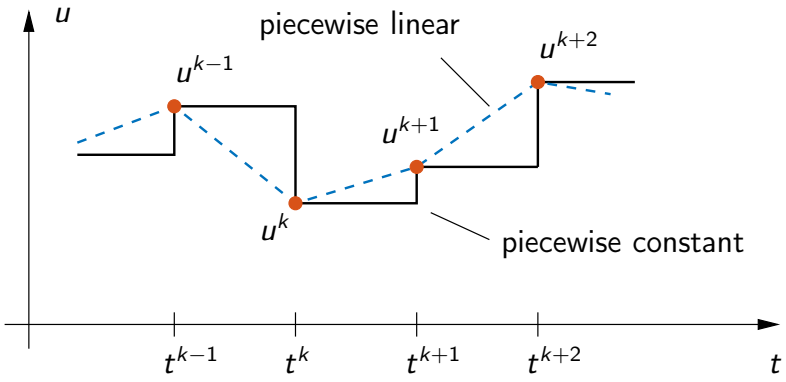


Figure 5.23: Piecewise constant and piecewise linear parametrization of the control ($m = 1$).

The parametrization of the control trajectory $\boldsymbol{u}(t) = \boldsymbol{\psi}(t, \hat{\boldsymbol{u}})$ transforms the optimal control problem (5.137) to a (finite-dimensional) parameter optimization problem

$$\min_{\hat{\boldsymbol{u}} \in \mathbb{R}^{mN}} \; \hat{J}(\hat{\boldsymbol{u}}) = V(\boldsymbol{x}(t_f)) + \sum_{k=0}^{N-1} \int_{t^k}^{t^{k+1}} l(\boldsymbol{x}(t), \boldsymbol{\psi}(t, \hat{\boldsymbol{u}}), t)\, \mathrm{d}t \tag{5.140a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\psi}(t, \hat{\boldsymbol{u}}), t)\,, \quad t \in (t^k, t^{k+1})\,, \qquad k = 0, 1, \ldots, N-1 \tag{5.140b}$$

$$\boldsymbol{x}(t^0) = \boldsymbol{x}_0\,, \quad \boldsymbol{x}(t^k) = \lim_{t \nearrow t^k} \boldsymbol{x}(t)\,, \qquad k = 0, 1, \ldots, N-1 \tag{5.140c}$$

$$\boldsymbol{g}(\boldsymbol{x}(t^N)) = \boldsymbol{0} \tag{5.140d}$$

$$\boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{\psi}(t, \hat{\boldsymbol{u}}), t) \leq \boldsymbol{0} \quad \forall t \in [t^k, t^{k+1}]\,, \quad k = 0, 1, \ldots, N-1\,. \tag{5.140e}$$

The solution of the nonlinear optimization problem (5.140) w.r.t. the parameters $\hat{\boldsymbol{u}}$ requires to numerically solve the dynamics (5.140b) on the single subintervals. The conditions (5.140c) ensure the continuity of the state $\boldsymbol{x}(t)$ at the interval boundary points t^k.

A problem of the formulation (5.140) is that formally the inequality constraints (5.140e) have to be satisfied on each interval $[t^k, t^{k+1}]$ and not only at the boundary points t^k. A practical approach in this regard is to replace (5.140e) by pointwise inequality constraints

$$\boldsymbol{h}(\boldsymbol{x}(t^{k,j}), \boldsymbol{\psi}(t^{k,j}, \hat{\boldsymbol{u}}), t^{k,j}) \leq \boldsymbol{0}$$

at a number of inner points $t^{k,j} \in [t^k, t^{k+1}]$. The problem (5.140) can then be solved with standard methods from nonlinear optimization, e.g. sequential quadratic programming (SQP).

Some characteristic properties of direct methods based on partial discretization are:

- If the conditions (5.140c) are implemented as initial conditions and not as equality constraints within the optimization method, the state trajectory $\boldsymbol{x}(t)$, $t \in [t_0, t_f]$ satisfies the differential equation (5.140b) (to the accuracy of the numerical integration scheme) during the single iterations of the optimization method.
- The computation of the gradient of the cost and constraints in (5.140), e.g. via sensitivity differential equations, is computationally complex.
- The partial discretization approach is a *direct sequential method* due to the underlying integration of the system equations (5.140b) during the optimization iterations.

5.7.2 Full discretization

The full discretization approach discretizes the control trajectory as well as the differential equations (5.137b) at the discretization points (5.138). In particular, if we use the trapezoidal scheme (5.126), see Section 5.6.1, and use a piecewise constant discretization for the control $\boldsymbol{u}$, we get the equations

$$\frac{\boldsymbol{x}^{k+1} - \boldsymbol{x}^k}{t^{k+1} - t^k} = \frac{1}{2}\Big[\boldsymbol{f}(\boldsymbol{x}^k, \boldsymbol{u}^k, t^k) + \boldsymbol{f}(\boldsymbol{x}^{k+1}, \boldsymbol{u}^{k+1}, t^{k+1})\Big], \quad k = 0, 1, \ldots, N-1\,.$$

The full discretization of the optimal control problem (5.137) thus yields the nonlinear optimization problem

$$\min_{\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}} \quad \hat{J}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}) = V(\boldsymbol{x}^N) + \sum_{k=0}^{N-1} \frac{t^{k+1} - t^k}{2}\Big[l(\boldsymbol{x}^k, \boldsymbol{u}^k, t^k) + l(\boldsymbol{x}^{k+1}, \boldsymbol{u}^{k+1}, t^{k+1})\Big] \tag{5.141a}$$

$$\text{s.t.} \quad \frac{\boldsymbol{x}^{k+1} - \boldsymbol{x}^k}{t^{k+1} - t^k} = \frac{1}{2}\Big[\boldsymbol{f}(\boldsymbol{x}^k, \boldsymbol{u}^k, t^k) + \boldsymbol{f}(\boldsymbol{x}^{k+1}, \boldsymbol{u}^{k+1}, t^{k+1})\Big], \quad k = 0, 1, \ldots, N-1 \tag{5.141b}$$

$$\boldsymbol{x}^0 = \boldsymbol{x}_0\,, \quad \boldsymbol{g}(\boldsymbol{x}^N) = \boldsymbol{0} \tag{5.141c}$$

$$\boldsymbol{h}(\boldsymbol{x}^k, \boldsymbol{u}^k, t^k) \leq \boldsymbol{0} \quad k = 0, 1, \ldots, N\,. \tag{5.141d}$$

The discretized system equations (5.141b) as well as the initial and terminal conditions (5.141c) are treated as equality constraints. The optimization variables consist of the discretized input and state variables

$$\hat{\boldsymbol{u}} = \begin{bmatrix} \boldsymbol{u}^0 \\ \vdots \\ \boldsymbol{u}^N \end{bmatrix} \in \mathbb{R}^{m(N+1)}, \quad \hat{\boldsymbol{x}} = \begin{bmatrix} \boldsymbol{x}^0 \\ \vdots \\ \boldsymbol{x}^N \end{bmatrix} \in \mathbb{R}^{n(N+1)} \tag{5.142}$$

of dimension $(m+n)(N+1)$. In contrast, the partial discretization of the control led to the optimization vector $\hat{\boldsymbol{u}}$ of dimension mN. On the other hand, the differential equations (5.137b) do not need to be integrated separately as they are handled as (discretized) equality constraints.

Some properties of the full discretization approach are:

- The optimization variables $\hat{\boldsymbol{x}}$ satisfy the discretized system equations (5.141b) only after convergence to the optimal solution $\hat{\boldsymbol{x}}^*$.
- The computations of the gradients of the cost function and constraints in (5.141) is simpler than in case of partial discretization. Moreover, the matrices are sparsely occupied and block-structured, which can be effectively exploited by sparse numerics.
- The inequality constraints (5.141d) are already given in discretized form in contrast to (5.140e) in the partial discretization case.
- The accuracy of the fully discretized optimal solution is directly related to the number of discretization points $N+1$. The choice of N not only influences the resolution of the control trajectory as in case of partial discretization but also how accurately $\hat{\boldsymbol{x}}$ approximates the actual solution of the differential equations (5.137b).
- The full discretization approach is a *direct simultaneous method* because the solution of the dynamics and the actual optimization is done at the same time.

5.8 Software overview

Besides the option to discretize an optimal control problem manually and to solve it as nonlinear optimization problem, there exist various software packages to solve dynamic optimization problems that mainly rely on direct partial or full discretization and often link to external solvers (e.g. IPOPT or qpOASES). A selection of open source and commercial tools are listed in the following.

- ACADO (open source licence)
  `http://acado.github.io/`
- CasADi (open source licence)
  `https://web.casadi.org`
- GRAMPC (open source licence)
  `https://sourceforge.net/projects/grampc`
- HQP (open source licence)
  `http://hqp.sourceforge.net`
- ICLOCS2 (open source, but external solver required)
  `http://www.ee.ic.ac.uk/iclocs`
- TOMLAB/PROPT (commercial)
  `http://tomdyn.com`

5.9 References

[1] U.M. Ascher, R.M.M. Mattheij, and R.D. Russell. *Numerical solution of boundary value problems of ordinary differential equations*. Prentice Hall, 1988.

[2] M. Athans and P.L. Falb. *Optimal Control*. New York: McGraw-Hill, 1966.

[3] R. Bittner, A. Driescher, and E.D. Gilles. "Drift dynamics modeling for automatic track-keeping of inland vessels". In: *10th Saint Petersburg International Conference on Integrated Navigation Systems*. St. Petersburg (Rußland), 2003, pp. 218–227.

[4] B. van Brunt. *The Calculus of Variations*. New York: Springer, 2004.

[5] A.E. Bryson and Y.-C. Ho. *Applied Optimal Control*. New York: John Wiley & Sons, 1975.

[6] B. Chachuat. *Nonlinear and Dynamic Optimization: From Theory to Practice*. `https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.362.5256&rep=rep1&type=pdf`, 2007.

[7] T. Englert, A. Völz, F. Mesmer, S. Rhein, and K. Graichen. "A software framework for embedded nonlinear model predictive control using a gradient-based augmented Lagrangian approach (GRAMPC)". In: *Optimization and Engineering* 20.3 (2019), pp. 769–809. DOI: 10.1007/s11081-018-9417-2.

[8] O. Föllinger. *Optimale Steuerung und Regelung*. München: Oldenbourg, 1994.

[9] R. Fourer, D.M. Gay, and B.W. Kernighan. *AMPL: A modeling language for mathematical programming*. Duxbury, 2002.

[10] K. Graichen. *Feedforward Control Design for Finite-Time Transition Problems of Nonlinear Systems with Input and Output Constraints.* Dissertation, Universität Stuttgart. `http://dx.doi.org/10.18419/opus-4093`. Aachen: Shaker-Verlag, 2006.

[11] R.F. Hartl, S.P. Sethi, and R.G. Vickson. "A survey of the maximum principles for optimal control problems with state constraints". In: *SIAM Review* 37 (1995), pp. 181–218.

[12] M.I. Kamien and N.L. Schwartz. *Dynamic Optimization: The Calculus of Variations and Optimal Control in Economics and Management*. 2nd edition. Amsterdam: North-Holland, Amsterdam, 1991.

[13] J. Kierzenka and L.F. Shampine. "A BVP solver based on residual control and the MATLAB PSE". In: *ACM Transactions on Mathematical Software* 27 (2001), pp. 299–316.

[14] H. Kwakernaak and R. Sivan. *Linear Optimal Control Systems*. New York: Wiley-Interscience, 1972.

[15] M. Papageorgiou, M. Leibold, and M. Buss. *Optimierung*. Berlin: Springer, 2012.

[16] H.J. Pesch. "Optimal and nearly optimal guidance by multiple shooting". In: *Proc. Intern. Symp. "Mécanique Spatiale"*. Ed. by Centre d'Études Spatiales (CNES). Erhältlich unter `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.54.8387`. Toulouse, 6.-10.11.1989: Cepadues Editions, 1990, pp. 761–771.

[17] E. Polak. "An historical survey of computational methods in optimal control". In: *SIAM Review* 15 (1973), pp. 553–584.

[18] L.F. Shampine, J. Kierzenka, and M.W. Reichelt. *Solving boundary value problems for ordinary differential equations in* MATLAB *with bvp4c*. `https://classes.engineering.wustl.edu/che512/bvp_paper.pdf`. 2003.

6 Model predictive control (MPC)

Model predictive control relies on the solution of a dynamic optimization problem on a moving horizon and is particularly suitable for controlling nonlinear multiple input systems subject to constraints. The chapter gives an introduction to MPC and different MPC formulations. Since MPC typically requires a high computational effort, we also focus on aspects of a real-time implementation. Further insights and detailed discussions on MPC can be found in the literature, e.g. [20, 2, 8, 5, 15].

6.1 Basic concept behind MPC

This section introduces the working principle of MPC for controlling nonlinear systems of the form

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)), \quad \boldsymbol{x}(0) = \boldsymbol{x}_0 \tag{6.1}$$

with the state $\boldsymbol{x} \in \mathbb{R}^n$ and control input $\boldsymbol{u} \in \mathbb{R}^m$ that are subject to the constraints

$$\boldsymbol{u}(t) \in U, \quad \boldsymbol{x}(t) \in X, \quad t \geq 0. \tag{6.2}$$

Without loss of generality, we consider the control task of stabilizing the origin

$$\boldsymbol{0} = \boldsymbol{f}(\boldsymbol{0}, \boldsymbol{0}). \tag{6.3}$$

Note that any other equilibrium $(\boldsymbol{x}_e, \boldsymbol{u}_e)$ with $\boldsymbol{0} = \boldsymbol{f}(\boldsymbol{x}_e, \boldsymbol{u}_e)$ can be mapped to the origin by considering the simple state/input transformation $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x}_e$ and $\tilde{\boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{u}_e$. Although we focus on setpoint stabilization, MPC can be used for various control tasks such as path following MPC, economic MPC, stochastic MPC, robust MPC or even as moving horizon estimator (MHE) for state observation. We refer to the references above for more details.

6.1.1 Dynamic optimization on moving horizon

The goal of MPC is to act as a control law for a given system based on the solution of an optimal control problem

$$\begin{aligned}
\min_{\bar{\boldsymbol{u}}(\cdot)} \quad & J_T(\boldsymbol{x}_k, \bar{\boldsymbol{u}}) = V(\bar{\boldsymbol{x}}(T)) + \int_0^T l(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau))\, \mathrm{d}\tau && \text{(6.4a)} \\
\text{s.t.} \quad & \dot{\bar{\boldsymbol{x}}}(\tau) = \boldsymbol{f}(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau)), \quad \bar{\boldsymbol{x}}(0) = \boldsymbol{x}_k = \boldsymbol{x}(t_k) && \text{(6.4b)} \\
& \bar{\boldsymbol{x}}(\tau) \in X, \quad \bar{\boldsymbol{u}}(\tau) \in U, \quad \tau \in [0, T] && \text{(6.4c)} \\
& \bar{\boldsymbol{x}}(T) \in S && \text{(6.4d)}
\end{aligned}$$

with the current system state $\boldsymbol{x}_k$ at the time point t_k. The prediction horizon T defines the time interval $[0, T]$ on which the system trajectories are predicted. The bar indicates

MPC-internal variables with the prediction time coordinate $\tau \in [0, T]$ in contrast to the real system (6.1) with time $t \geq 0$. We also consider the following standing assumptions:

Standing assumptions:

A.1 The system function $\boldsymbol{f}$ is continuously differentiable in its arguments and there exists a bounded solution of the system (6.1) for all $\boldsymbol{x}_0 \in X$ and all controls $\boldsymbol{u}(t) \in U$, $t \in [0, T]$.

A.2 The constraint sets X and U with $\mathbf{0} \in X \subseteq \mathbb{R}^n$ and $\mathbf{0} \in U \subset \mathbb{R}^m$ are convex and U is additionally compact.

A.3 The integral cost term $l(\boldsymbol{x}, \boldsymbol{u})$ and the (optional) terminal cost function $V(\boldsymbol{x})$ in the cost functional (6.4a) are continuously differentiable in their arguments. Moreover, they are positive definite with $l(\mathbf{0}, \mathbf{0}) = V(\mathbf{0}) = 0$ and satisfy the quadratic bounds

$$\begin{aligned} m_l(||\boldsymbol{x}||^2 + ||\boldsymbol{u}||^2) \leq \; & l(\boldsymbol{x}, \boldsymbol{u}) \; \leq M_l(||\boldsymbol{x}||^2 + ||\boldsymbol{u}||^2) \\ m_V||\boldsymbol{x}||^2 \leq \; & V(\boldsymbol{x}) \; \leq M_V||\boldsymbol{x}||^2 \end{aligned} \tag{6.5}$$

for some constants $m_l, M_l > 0$ and $m_V, M_V > 0$.

A common choice of $l(\boldsymbol{x}, \boldsymbol{u})$ and $V(\boldsymbol{x})$ are quadratic functions

$$V(\boldsymbol{x}) = \boldsymbol{x}^\mathsf{T} \boldsymbol{P} \boldsymbol{x}\,, \quad l(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{x}^\mathsf{T} \boldsymbol{Q} \boldsymbol{x} + \boldsymbol{u}^\mathsf{T} \boldsymbol{R} \boldsymbol{u} \tag{6.6}$$

with positive definite matrices $\boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{R}$. Besides the input and state constraints (6.4c), some MPC formulations impose an additional terminal constraint (6.4d) with $\mathbf{0} \in S \subset \mathbb{R}^n$ or $S_\beta = \mathbf{0}$ (Section 6.2).

Under the assumption that the optimal control problem (6.4) is feasible [1], the optimal solution and the optimal cost functional are denoted by

$$\bar{\boldsymbol{u}}_T^*(\tau; \boldsymbol{x}_k)\,, \quad \bar{\boldsymbol{x}}_T^*(\tau; \boldsymbol{x}_k)\,, \quad \tau \in [0, T] \tag{6.7}$$

and

$$J_T^*(\boldsymbol{x}_k) := J_T(\boldsymbol{x}_k, \bar{\boldsymbol{u}}_k^*)\,.$$

It is typically assumed in the literature that the optimal solution of (6.4) is determined in each time step t_k. In the current sampling interval $t \in [t_k, t_k + \Delta t)$ with the *sampling time* Δt, the first part of the optimal control trajectory $\bar{\boldsymbol{u}}_T^*(\tau; \boldsymbol{x}_k)$ is used as input

$$\boldsymbol{u}(t_k + \tau) = \bar{\boldsymbol{u}}_T^*(\tau; \boldsymbol{x}_k) =: \boldsymbol{\kappa}_T(\bar{\boldsymbol{x}}_T^*(\tau; \boldsymbol{x}_k); \boldsymbol{x}_k)\,, \quad \tau \in [0, \Delta t) \tag{6.8}$$

for the system (6.1), which can be interpreted as an *optimal control law* with $\boldsymbol{\kappa}_T(\mathbf{0}; \boldsymbol{x}_k) = \mathbf{0}$. In the next sampling step $t_{k+1} = t_k + \Delta t$, the problem (6.4) is re-solved with the new initial state $\boldsymbol{x}_{k+1}$. Figure 6.1 illustrates the moving horizon concept of MPC.

If no model unaccuracies or disturbances occur, the system follows the predicted state trajectory to the next state $\boldsymbol{x}_{k+1} = \boldsymbol{x}_T^*(\Delta t; \boldsymbol{x}_k)$. The closed-loop trajectory in the nominal case therefore reads

$$\begin{aligned} \boldsymbol{x}(t) &= \boldsymbol{x}(t_k + \tau) = \bar{\boldsymbol{x}}_T^*(\tau; \boldsymbol{x}_k)\,, \\ \boldsymbol{u}(t) &= \boldsymbol{u}(t_k + \tau) = \bar{\boldsymbol{u}}_T^*(\tau; \boldsymbol{x}_k)\,, \quad \tau \in [0, \Delta t)\,, \quad k \in \mathbb{N}_0^+\,. \end{aligned} \tag{6.9}$$

[1] Existence results for solutions of certain classes of optimal control problems can e.g. be found in [18, 1].

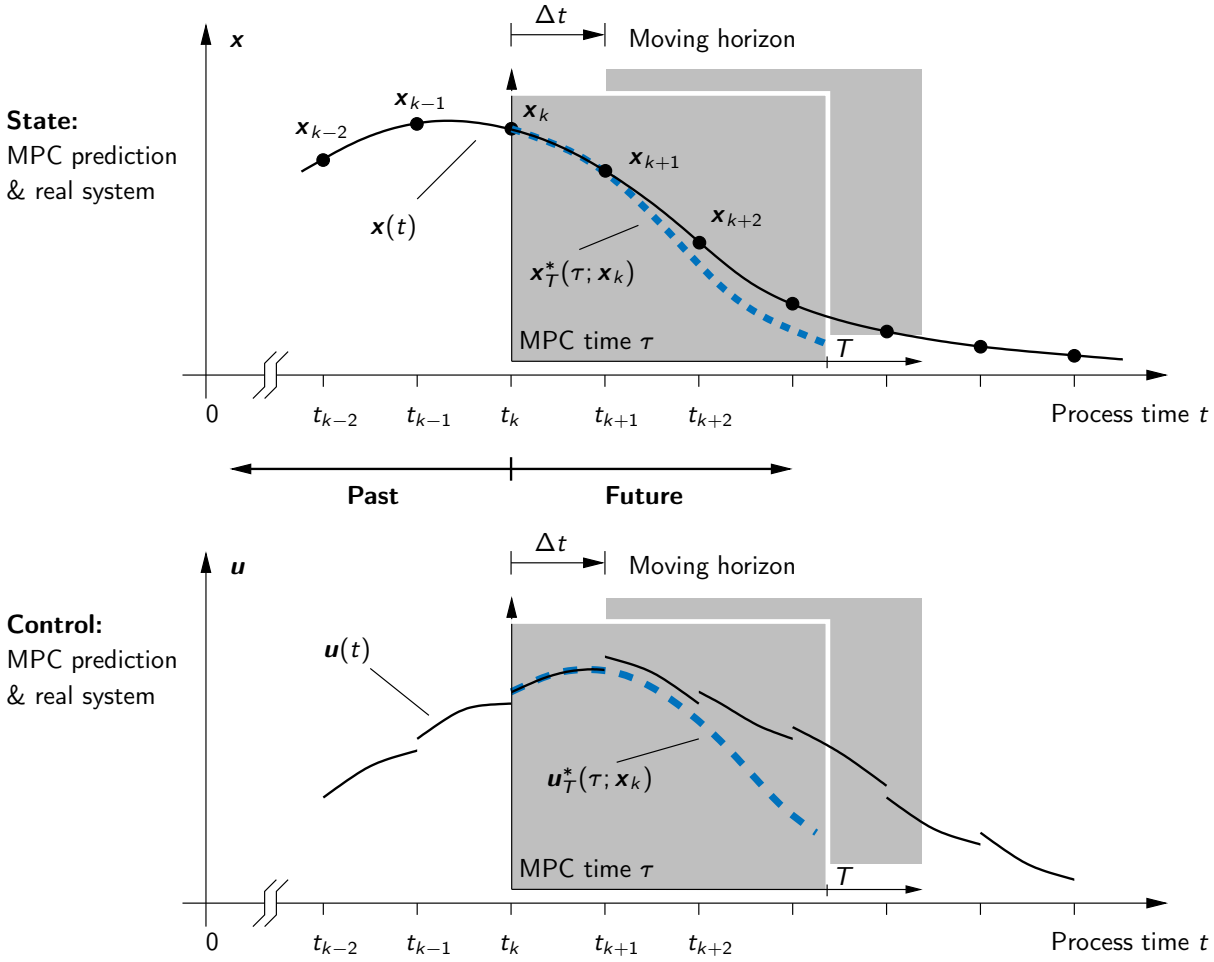


Figure 6.1: Moving horizon principle of MPC.

6.1.2 Continuous-time vs. discrete-time MPC

For highly dynamical systems with short sampling times, the piecewise continuous control input (6.8) is of rather theoretical nature. Instead, a constant input is often applied to the system on the current sampling interval, e.g. by setting $\boldsymbol{u}(t_k + \tau) = \bar{\boldsymbol{u}}_T^*(0; \boldsymbol{x}_k) = const.$ for $\tau \in [0, \Delta t)$, or by directly using a piecewise constant parametrization of the control trajectory for the solution of (6.4), see Figure 6.2 as well as Section 5.7.1.

If the control is kept constant over a sampling interval, i.e. $\boldsymbol{u}(t) = \boldsymbol{u}_k$ for $t \in [t_k, t_k + \Delta t)$, we formally obtain the state trajectory $\boldsymbol{x}(t) = \boldsymbol{x}(t; \boldsymbol{x}_k, \boldsymbol{u}_k)$ and, in particular, the state $\boldsymbol{x}_{k+1} = \boldsymbol{x}(t_{k+1}; \boldsymbol{x}_k, \boldsymbol{u}_k)$ at the next sampling instant $t_{k+1} = t_k + \Delta t$. Thus, the continuous-time system (6.1) can formally be written as a nonlinear discrete-time system

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}_\mathrm{d}(\boldsymbol{x}_k, \boldsymbol{u}_k)\,. \tag{6.10}$$

In fact, the majority of the MPC literature considers a discrete-time version of (6.4)

$$\min_{\hat{\boldsymbol{u}}} \quad J_N(\boldsymbol{x}_k, \bar{\boldsymbol{u}}) = V(\bar{\boldsymbol{x}}_N) + \sum_{j=0}^{N-1} l(\bar{\boldsymbol{x}}_j, \bar{\boldsymbol{u}}_j) \tag{6.11a}$$

$$\text{s.t.} \quad \bar{\boldsymbol{x}}_{j+1} = \boldsymbol{f}_\mathrm{d}(\bar{\boldsymbol{x}}_j, \bar{\boldsymbol{u}}_j)\,, \quad \bar{\boldsymbol{x}}_0 = \boldsymbol{x}_k = \boldsymbol{x}(t_k) \tag{6.11b}$$

$$\bar{\boldsymbol{x}}_j \in X\,, \quad \bar{\boldsymbol{u}}_j \in U\,, \quad j = 0, 1, \ldots, N-1 \tag{6.11c}$$

$$\bar{\boldsymbol{x}}_N \in S \tag{6.11d}$$

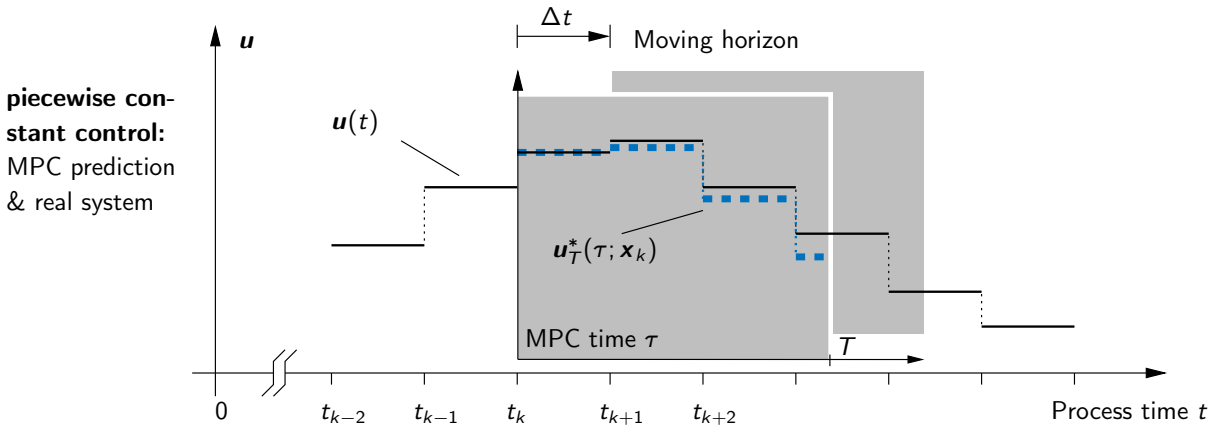


Figure 6.2: MPC with piecewise constant control.

with N prediction steps and the input parametrization $\hat{\boldsymbol{u}} = [\bar{\boldsymbol{u}}_0, \dots, \bar{\boldsymbol{u}}_{N-1}]^\mathsf{T}$. The discrete-time approach has the advantage that it is mathematically easier to analyze and that weaker assumptions are required compared to the continuous-time case.

On the other hand, the discrete-time formulation (6.10) for a nonlinear continuous-time system (6.1) can in general only be derived approximately. In addition, the system theoretic properties of the original system (6.1) are usually better observed in continuous-time, which is the reason that we focus on the continuous-time MPC formulation from Section 6.1.1.

6.2 MPC formulations

Different MPC formulations exist as variants of the general optimal control problem (6.4). These different formulations are the result of the historical development of MPC but are also motivated by practical considerations such as computational efficiency for real-time implementations. The formulations therefore differ in their importance in theory and practice. For instance, terminal constraints (6.4d) are useful to prove stability in closed-loop. On the other hand, it is difficult to guarantee the existence of a solution for terminal-constrained problems and the numerical effort for its solution is significantly higher than in case of a free terminal state. This section gives an overview of different MPC formulations and their stability properties in closed-loop.

6.2.1 Infinite horizon

If we consider an infinitely long prediction horizon $T \to \infty$, the optimal control problem (6.4) can be written without terminal conditions (6.4d) and without terminal cost in (6.4a) because the optimal solution must satisfy $\lim_{t\to\infty} \bar{\boldsymbol{x}}^*_\infty(t; \boldsymbol{x}_k) = \boldsymbol{0}$ for the cost to be finite. This leads to the MPC formulation

$$\min_{\bar{\boldsymbol{u}}(\cdot)} \quad J_\infty(\boldsymbol{x}_k, \bar{\boldsymbol{u}}) = \int_0^\infty l(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau))\, \mathrm{d}\tau \tag{6.12a}$$

$$\text{s.t.} \quad \dot{\bar{\boldsymbol{x}}}(\tau) = \boldsymbol{f}(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau)), \quad \bar{\boldsymbol{x}}(0) = \boldsymbol{x}_k = \boldsymbol{x}(t_k) \tag{6.12b}$$

$$\bar{\boldsymbol{x}}(\tau) \in X, \quad \bar{\boldsymbol{u}}(\tau) \in U, \quad \tau \in [0, \infty) \tag{6.12c}$$

and the following stability result:

Theorem 6.1 (MPC with infinite horizon) *Suppose that the set $\Gamma \subseteq X \subseteq \mathbb{R}^n$ of initial states $\boldsymbol{x}_k$ for which (6.12) has a bounded solution with $J_\infty^*(\boldsymbol{x}_k) < \infty$ is non-empty. Then, the cost satisfies*

$$J_\infty^*(\bar{\boldsymbol{x}}_\infty^*(\Delta t; \boldsymbol{x}_k)) = J_\infty^*(\boldsymbol{x}_k) - \int_0^{\Delta t} l(\bar{\boldsymbol{x}}_\infty^*(\tau; \boldsymbol{x}_k), \bar{\boldsymbol{u}}_\infty^*(\tau; \boldsymbol{x}_k))\,\mathrm{d}\tau \quad \forall\, \boldsymbol{x}_k \in \Gamma \tag{6.13}$$

and the origin of the system (6.1) in closed-loop is asymptotically stable in the sense of $\lim_{t\to\infty} ||\boldsymbol{x}(t)|| = 0$ with Γ as the domain of attraction.

Proof: The equation (6.13) directly follows from Bellman's principle of optimality. If (6.12) is feasible for $t = t_0$ ($\boldsymbol{x}_0 \in \Gamma$), then the infinite horizon guarantees that (6.12) is also feasible for the next point $\boldsymbol{x}_{k+1} = \bar{\boldsymbol{x}}_\infty^*(\Delta t; \boldsymbol{x}_k)$, i.e. $\boldsymbol{x}_{k+1} \in \Gamma$. By the definition of the set Γ, the closed-loop trajectory $\boldsymbol{x}(t)$ in (6.9) is bounded. This also holds for the time derivative $\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t))$ with $\boldsymbol{u}(t) \in U$ since U is compact and $\boldsymbol{f}$ is continuous in $\boldsymbol{x}$ and $\boldsymbol{u}$. Thus, $\boldsymbol{x}(t)$ is *uniformly continuous*. Moreover, with inductive reasoning the cost satisfies

$$J_\infty^*(\boldsymbol{x}(t)) = J_\infty^*(\boldsymbol{x}_0) - \int_0^t l(\boldsymbol{x}(\tau), \boldsymbol{u}(\tau))\,\mathrm{d}\tau \overset{(6.5)}{\leq} J_\infty^*(\boldsymbol{x}_0) - \int_0^t m_l ||\boldsymbol{x}(\tau)||^2\,\mathrm{d}\tau \tag{6.14}$$

along the closed-loop trajectory $\boldsymbol{x}(t)$. Since $J_\infty^*(\boldsymbol{x}_0)$, $\boldsymbol{x}_0 \in \Gamma$ is bounded and $J_\infty^*(\boldsymbol{x}(t)) \geq 0$, the integral in (6.14) is bounded as well. We now require *Barbalat's lemma* to proceed:

Lemma 6.1 (Barbalat's Lemma) *Let $\phi : \mathbb{R} \to \mathbb{R}$ be a uniformly continuous function on the interval $[0, \infty)$. If the integral*

$$\lim_{t\to\infty} \int_0^t \phi(\tau)\,\mathrm{d}\tau$$

exists and is finite, then $\lim_{t\to\infty} \phi(t) = 0$.

Obviously, the assumptions of Lemma 6.1 are satisfied with $\phi(\tau) = ||\boldsymbol{x}(\tau)||^2$, which implies that $||\boldsymbol{x}(t)|| \to 0$ for $t \to \infty$. ■

Note that $\lim_{t\to\infty} ||\boldsymbol{x}(t)|| = 0$ does not automatically imply asymptotic stability in the sense of *Lyapunov* (see e.g. the course "Nonlinear Control Systems"). This can be proved if in addition continuity of J_∞^* is assumed or shown.

Moreover, if J_∞^* is continuously differentiable on Γ and if the levels sets of J_∞^* are compact, then for $\Delta t \to 0$ the cost satisfies

$$\frac{\mathrm{d}}{\mathrm{d}t} J_\infty^*(\boldsymbol{x}(t)) = \frac{\partial J_\infty^*(\boldsymbol{x}(t))}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)) = -l(\boldsymbol{x}(t), \boldsymbol{u}(t)) \overset{(6.5)}{\leq} -m_l ||\boldsymbol{x}(t)||^2 \leq 0 \tag{6.15}$$

along the closed-loop trajectory (6.9). This shows that each compact subset

$$\Gamma_\alpha = \{x \in \Gamma : J_\infty^*(\boldsymbol{x}) \leq \alpha\} \tag{6.16}$$

of the domain of attraction Γ is *positive invariant*. In fact, this property holds not only for the infinitesimal case ($\Delta t \to 0$) but also for the incremental one ($\Delta t > 0$). The optimal cost $J_\infty^*(\boldsymbol{x})$ then represents a *Lyapunov function*, which implies that the origin is asymptotically stable.

6.2.2 Finite horizon and terminal condition

The numerical solution of the problem (6.12) on an infinite horizon can only be solved approximately and requires a high computational effort. Over the history of MPC, several formulations have emerged that use a finite horizon $T < \infty$ in connection with additional conditions to maintain stability in closed-loop. The first variant concerns an MPC formulation with terminal condition:

$$\min_{\bar{\boldsymbol{u}}(\cdot)} \quad J_T(\boldsymbol{x}_k, \bar{\boldsymbol{u}}) = \int_0^T l(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau))\,\mathrm{d}\tau \tag{6.17a}$$

$$\text{s.t.} \quad \dot{\bar{\boldsymbol{x}}}(\tau) = \boldsymbol{f}(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau))\,, \quad \bar{\boldsymbol{x}}(0) = \boldsymbol{x}_k = \boldsymbol{x}(t_k) \tag{6.17b}$$

$$\bar{\boldsymbol{x}}(\tau) \in X\,, \quad \bar{\boldsymbol{u}}(\tau) \in U\,, \quad \tau \in [0, T] \tag{6.17c}$$

$$\bar{\boldsymbol{x}}(T) = \boldsymbol{0}\,. \tag{6.17d}$$

The formulation (6.17) demands that the predicted state trajectory $\bar{\boldsymbol{x}}_T^*(\tau; \boldsymbol{x}_k)$ exactly reaches the origin (i.e. the desired setpoint) at the end of the prediction horizon T. We have the following stability results for this MPC formulation with terminal condition.

Theorem 6.2 (MPC with terminal condition) *Suppose that the set $\Gamma \subseteq X \subseteq \mathbb{R}^n$ of initial states $\boldsymbol{x}_k$ for which (6.17) has a bounded solution with $J_T^*(\boldsymbol{x}_k) < \infty$ is non-empty. Then, the cost satisfies*

$$J_T^*(\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k)) \leq J_T^*(\boldsymbol{x}_k) - \int_0^{\Delta t} l(\bar{\boldsymbol{x}}_T^*(\tau; \boldsymbol{x}_k), \bar{\boldsymbol{u}}_T^*(\tau; \boldsymbol{x}_k))\,\mathrm{d}\tau \quad \forall\, \boldsymbol{x}_k \in \Gamma \tag{6.18}$$

and the origin of the system (6.1) in closed-loop is asymptotically stable in the sense of $\lim_{t\to\infty} ||\boldsymbol{x}(t)|| = 0$ with Γ as the domain of attraction.

Proof: Since $\bar{\boldsymbol{x}}_T^*(T; \boldsymbol{x}_k) = \boldsymbol{0}$ holds for all $\boldsymbol{x}_k \in \Gamma$, the trajectory

$$\hat{\boldsymbol{x}}(\tau) = \begin{cases} \bar{\boldsymbol{x}}_T^*(\tau + \Delta t; \boldsymbol{x}_k) & \tau \in [0, T - \Delta t) \\ \boldsymbol{0} & \tau \in [T - \Delta t, T] \end{cases}, \quad \hat{\boldsymbol{u}}(\tau) = \begin{cases} \bar{\boldsymbol{u}}_T^*(\tau + \Delta t; \boldsymbol{x}_k) & \tau \in [0, T - \Delta t) \\ \boldsymbol{0} & \tau \in [T - \Delta t, T] \end{cases}$$

is feasible for the problem (6.17) in MPC step $k+1$, i.e. for $\boldsymbol{x}_{k+1} = \bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k)$. The trajectories $\hat{\boldsymbol{x}}(\tau)$ and $\hat{\boldsymbol{u}}(\tau)$ can be used to compute an upper bound on the cost in step $k+1$

$$\begin{aligned} J_T^*(\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k)) &\leq \int_0^T l(\hat{\boldsymbol{x}}(\tau), \hat{\boldsymbol{u}}(\tau))\,\mathrm{d}\tau \\ &= \int_{\Delta t}^T l(\bar{\boldsymbol{x}}_T^*(\tau; \boldsymbol{x}_k), \bar{\boldsymbol{u}}_T^*(\tau; \boldsymbol{x}_k))\,\mathrm{d}t + \overbrace{\int_0^{\Delta t} l(\boldsymbol{0}, \boldsymbol{0})\,\mathrm{d}\tau}^{=0} \\ &= J_T^*(\boldsymbol{x}_k) - \int_0^{\Delta t} l(\bar{\boldsymbol{x}}_T^*(\tau; \boldsymbol{x}_k), \bar{\boldsymbol{u}}_T^*(\tau; \boldsymbol{x}_k))\,\mathrm{d}t\,, \end{aligned}$$

which corresponds to (6.18). The proof of stability ($\lim_{t\to\infty} ||\boldsymbol{x}(t)|| = 0$) then follows the lines of the proof of Theorem 6.1. ■

The practical implementation of the MPC formulation with terminal conditions (6.17d) has several deficiencies. On the one hand, the set Γ on which the problem (6.17) has a (bounded) solution can be small if the prediction horizon T is chosen too short. The problem might even be unsolvable in case of disturbances. On the other hand, the (numerically) exact satisfaction of a terminal condition requires a high numerical effort.

6.2.3 Finite horizon and terminal set

One option to relax the terminal condition $\bar{\boldsymbol{x}}(T) = \mathbf{0}$ in (6.17) is to replace it with a *terminal set* $S_\beta \subseteq X$ that contains the origin ($\mathbf{0} \in S_\beta$). This directly leads to the MPC formulation with terminal set

$$\begin{aligned}
\min_{\bar{\boldsymbol{u}}(\cdot)} \quad & J_T(\boldsymbol{x}_k, \bar{\boldsymbol{u}}) = V(\bar{\boldsymbol{x}}(T)) + \int_0^T l(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau))\,\mathrm{d}\tau && \text{(6.19a)}\\
\text{s.t.} \quad & \dot{\bar{\boldsymbol{x}}}(\tau) = \boldsymbol{f}(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau))\,, \quad \bar{\boldsymbol{x}}(0) = \boldsymbol{x}_k = \boldsymbol{x}(t_k) && \text{(6.19b)}\\
& \bar{\boldsymbol{x}}(\tau) \in X\,, \quad \bar{\boldsymbol{u}}(\tau) \in U\,, \quad \tau \in [0, T] && \text{(6.19c)}\\
& \bar{\boldsymbol{x}}(T) \in S_\beta\,. && \text{(6.19d)}
\end{aligned}$$

However, we require an additional design condition for the terminal cost function in (6.19a) to compensate for the relaxation of the terminal condition in the last formulation (6.17).

Theorem 6.3 (MPC with terminal set) *Suppose that the set $\Gamma \subseteq X \subseteq \mathbb{R}^n$ of initial states $\boldsymbol{x}_k$ for which (6.19) has a bounded solution with $J_T^*(\boldsymbol{x}_k) < \infty$ is non-empty. Moreover, there exists a compact non-empty set $S_\beta = \{\boldsymbol{x} \in \mathbb{R}^n : V(\boldsymbol{x}) \leq \beta\} \subseteq \Gamma$ with $\beta > 0$ and a (local) control law $\boldsymbol{q}(\boldsymbol{x}) \in U$ for all $\boldsymbol{x} \in S_\beta$ such that*

$$\frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{q}(\boldsymbol{x})) + l(\boldsymbol{x}, \boldsymbol{q}(\boldsymbol{x})) \leq \mathbf{0} \quad \forall\, \boldsymbol{x} \in S_\beta\,. \tag{6.20}$$

Then, the cost satisfies

$$J_T^*(\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k)) \leq J_T^*(\boldsymbol{x}_k) - \int_0^{\Delta t} l(\bar{\boldsymbol{x}}_T^*(\tau; \boldsymbol{x}_k), \bar{\boldsymbol{u}}_T^*(\tau; \boldsymbol{x}_k))\,\mathrm{d}\tau \quad \forall\, \boldsymbol{x}_k \in \Gamma \tag{6.21}$$

and the origin of the system (6.1) in closed-loop is asymptotically stable in the sense of $\lim_{t\to\infty} ||\boldsymbol{x}(t)|| = 0$ with Γ as the domain of attraction.

Proof: A solution of (6.3) satisfies $\bar{\boldsymbol{x}}_T^*(T; \boldsymbol{x}_k) \in S_\beta$ for all $\boldsymbol{x}_k \in \Gamma$. To derive an estimate of the cost in MPC step $k+1$, we consider the auxiliary feasible trajectory

$$\hat{\boldsymbol{x}}(\tau) = \begin{cases} \bar{\boldsymbol{x}}_T^*(\tau + \Delta t; \boldsymbol{x}_k), & \tau \in [0, T - \Delta t) \\ \bar{\boldsymbol{x}}^q(\tau - T + \Delta t), & \tau \in [T - \Delta t, T] \end{cases}, \quad \hat{\boldsymbol{u}}(\tau) = \begin{cases} \bar{\boldsymbol{u}}_T^*(\tau + \Delta t; \boldsymbol{x}_k), & \tau \in [0, T - \Delta t) \\ \bar{\boldsymbol{u}}^q(\tau - T + \Delta t), & \tau \in [T - \Delta t, T]\,, \end{cases}$$

where $\bar{\boldsymbol{x}}^q(\tau)$ with $\bar{\boldsymbol{x}}^q(0) = \bar{\boldsymbol{x}}_T^*(T; \boldsymbol{x}_k)$ is the trajectory that results from applying the local control law $\bar{\boldsymbol{u}}^q(\tau) = \boldsymbol{q}(\bar{\boldsymbol{x}}^q(\tau))$. Note that $\bar{\boldsymbol{x}}^q(\tau) \in S_\beta$ for all $\tau \geq 0$, i.e. S_β is positive invariant due to the definition of S_β and the inequality (6.20) that can be expressed in the form

$$\frac{\mathrm{d}}{\mathrm{d}\tau} V(\bar{\boldsymbol{x}}^q(\tau)) \leq -l(\bar{\boldsymbol{x}}^q(\tau), \bar{\boldsymbol{u}}^q(\tau))\,. \tag{6.22}$$

Obviously, the trajectory $\hat{\boldsymbol{x}}(\tau)$, $\hat{\boldsymbol{u}}(\tau)$, $\tau \in [0, T]$ is feasible for the problem (6.19) in step $k+1$, i.e. for $\boldsymbol{x}_{k+1} = \bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k)$, which demonstrates the problem's solvability in step $k+1$. We

can now derive an upper bound on the cost

$$\begin{aligned} J_T^*(\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k)) &\leq \int_0^T l(\hat{\boldsymbol{x}}(\tau), \hat{\boldsymbol{u}}(\tau))\,\mathrm{d}\tau + V(\hat{\boldsymbol{x}}(T)) \\ &= J_T^*(\boldsymbol{x}_k) - \int_0^{\Delta t} l(\bar{\boldsymbol{x}}_T^*(\tau; \boldsymbol{x}_k), \bar{\boldsymbol{u}}_T^*(\tau; \boldsymbol{x}_k))\,\mathrm{d}\tau \\ &\quad + \underbrace{V(\bar{\boldsymbol{x}}^q(\Delta t)) - V(\bar{\boldsymbol{x}}^q(0)) + \int_0^{\Delta t} l(\bar{\boldsymbol{x}}^q(\tau), \bar{\boldsymbol{u}}^q(\tau))\,\mathrm{d}\tau}_{\leq 0}\,. \end{aligned} \tag{6.23}$$

The integration of the inequality (6.22) shows that the bracketed term in (6.23) is non-positive, which proves (6.21). The proof of stability in the sense of $\lim_{t\to\infty} ||\boldsymbol{x}(t)|| = 0$ corresponds to the proof of Theorem 6.1. ■

The additional requirement in Theorem 6.3 concerns the existence of a (local) control law $\boldsymbol{q}(\boldsymbol{x})$ such that the terminal cost $V(\boldsymbol{x})$ satisfies the inequality (6.20). In nonlinear control theory, (6.20) represents a (strengthened) *control Lyapunov inequality* which implies that the terminal cost $V(\boldsymbol{x})$ represents a *control Lyapunov function* (CLF) and $\boldsymbol{q}(\boldsymbol{x})$ a CLF control law.

Exercise 6.1 (CLF in the linear unconstrained case) *Consider the linear unconstrained MPC formulation*

$$\min_{\bar{\boldsymbol{u}}(\cdot)} \quad J_T(\boldsymbol{x}_k, \bar{\boldsymbol{u}}) = \bar{\boldsymbol{x}}^\mathsf{T}(T)\boldsymbol{P}\bar{\boldsymbol{x}}(T) + \int_0^T \bar{\boldsymbol{x}}^\mathsf{T}(\tau)\boldsymbol{Q}\bar{\boldsymbol{x}}(\tau) + \bar{\boldsymbol{u}}^\mathsf{T}(\tau)\boldsymbol{R}\bar{\boldsymbol{u}}(\tau),\mathrm{d}\tau \tag{6.24a}$$

$$\text{s.t.} \quad \dot{\bar{\boldsymbol{x}}}(\tau) = \boldsymbol{A}\bar{\boldsymbol{x}}(\tau) + \boldsymbol{B}\bar{\boldsymbol{u}}(\tau)\,, \quad \bar{\boldsymbol{x}}(0) = \boldsymbol{x}_k = \boldsymbol{x}(t_k) \tag{6.24b}$$

with $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{u} \in \mathbb{R}^m$, and the symmetric positive definite matrices $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{R} \in \mathbb{R}^{m\times m}$. Suppose that the system (6.24b) is stabilizable and the matrix $\boldsymbol{P}$ is the solution of the algebraic Riccati equation

$$\boldsymbol{P}\boldsymbol{A} + \boldsymbol{A}^\mathsf{T}\boldsymbol{P} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T}\boldsymbol{P} + \boldsymbol{Q} = \boldsymbol{0}\,. \tag{6.25}$$

Show that the quadratic terminal cost and the control law

$$\boldsymbol{q}(\boldsymbol{x}) = -\boldsymbol{K}\boldsymbol{x} \quad \textit{with} \quad \boldsymbol{K} = \boldsymbol{R}^{-1}\boldsymbol{B}^\mathsf{T}\boldsymbol{P} \tag{6.26}$$

together satisfy the CLF inequality (6.20) on $S_\beta = \mathbb{R}^n$, which implies that the terminal set S_β can be omitted.

Obviously, the CLF inequality (6.20) can be globally satisfied in the linear unconstrained MPC case (6.24) without having to account for the terminal set S_β.

Exercise 6.2 (CLF in the nonlinear unconstrained case) *Consider the nonlinear MPC formulation*

$$\min_{\bar{\boldsymbol{u}}(\cdot)} \quad J_T(\boldsymbol{x}_k, \bar{\boldsymbol{u}}) = \bar{\boldsymbol{x}}^\mathsf{T}(T)\boldsymbol{P}\bar{\boldsymbol{x}}(T) + \int_0^T \gamma\,\bar{\boldsymbol{x}}^\mathsf{T}(\tau)\boldsymbol{Q}\bar{\boldsymbol{x}}(\tau) + \bar{\boldsymbol{u}}^\mathsf{T}(\tau)\boldsymbol{R}\bar{\boldsymbol{u}}(\tau)\,\mathrm{d}\tau \tag{6.27a}$$

$$\text{s.t.} \quad \dot{\bar{\boldsymbol{x}}}(\tau) = \boldsymbol{f}(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau))\,, \quad \bar{\boldsymbol{x}}(0) = \boldsymbol{x}_k = \boldsymbol{x}(t_k) \tag{6.27b}$$

$$\bar{\boldsymbol{x}}(T) \in S_\beta\,. \tag{6.27c}$$

with $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{u} \in \mathbb{R}^m$, the positive definite matrices $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{R} \in \mathbb{R}^{m\times m}$ and some constant $\gamma \in (0, 1)$. Suppose that the linearization of the system (6.27b) around the origin

is stabilizable and let $\boldsymbol{P}$ be the solution of the algebraic Riccati equation (6.25). Show that for each $\gamma \in (0,1)$ there exists a terminal set $S_\beta \ni \mathbf{0}$ on which the CLF inequality (6.20) is satisfied with the local control law (6.26).

In the nonlinear case (6.27), the additional parameter $\gamma \in (0,1)$ is required to weaken the state-dependent integral part compared to the terminal cost to guarantee that a non-empty terminal set S_β exists.

In Exercise 6.1 and 6.2, the terminal cost function $V(\boldsymbol{x})$ is computed with the Riccati equation (6.25). There exist further approaches for constructing $V(\boldsymbol{x})$ such that the CLF inequality (6.20) holds. An algorithm for computing $V(\boldsymbol{x})$ and the terminal set S_β can e.g. be found in [4].

6.2.4 Finite horizon and terminal cost

The substitution of the terminal condition $\bar{\boldsymbol{x}}(T) = \mathbf{0}$ by the terminal set $\bar{\boldsymbol{x}}(T) \in S_\beta$ in the previous section already relaxes the computational aspects for solving the MPC problem. For an actual implementation, it would be even better to also drop the terminal set constraint $\bar{\boldsymbol{x}}(T) \in S_\beta$. This leads to the following MPC formulation with terminal cost and free terminal state

$$\min_{\bar{\boldsymbol{u}}(\cdot)} \quad J_T(\boldsymbol{x}_k, \bar{\boldsymbol{u}}) = V(\bar{\boldsymbol{x}}(T)) + \int_0^T l(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau))\,\mathrm{d}\tau \tag{6.28a}$$

$$\text{s.t.} \quad \dot{\bar{\boldsymbol{x}}}(\tau) = \boldsymbol{f}(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau))\,, \quad \bar{\boldsymbol{x}}(0) = \boldsymbol{x}_k = \boldsymbol{x}(t_k) \tag{6.28b}$$

$$\bar{\boldsymbol{x}}(\tau) \in X\,, \quad \bar{\boldsymbol{u}}(\tau) \in U\,, \quad \tau \in [0, T]\,. \tag{6.28c}$$

The absence of a terminal set S_β leads to the question under which conditions the end point of the optimal trajectory $\bar{\boldsymbol{x}}_T^*(T; \boldsymbol{x}_k)$ reaches S_β on its own such that the CLF inequality (6.20) holds. The proof of the following lemma [14] is an adaptation of the discrete-time version in [19] and is omitted here for simplicity.

Lemma 6.2 (Reachability of the terminal set) *Suppose that the set $\Gamma \subseteq X \subseteq \mathbb{R}^n$ of initial states $\boldsymbol{x}_k$ for which (6.28) has a bounded solution with $J_T^*(\boldsymbol{x}_k) < \infty$ is non-empty. Moreover, there exists a compact non-empty set $S_\beta = \{\boldsymbol{x} \in \mathbb{R}^n : V(\boldsymbol{x}) \leq \beta\} \subseteq \Gamma$ with $\beta > 0$ and a (local) control law $\boldsymbol{q}(\boldsymbol{x}) \in U$ for all $\boldsymbol{x} \in S_\beta$ such that*

$$\frac{\partial V}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{q}(\boldsymbol{x})) + l(\boldsymbol{x}, \boldsymbol{q}(\boldsymbol{x})) \leq \mathbf{0} \quad \forall\, \boldsymbol{x} \in S_\beta\,. \tag{6.29}$$

and consider the compact set

$$\Gamma_\alpha = \{\boldsymbol{x} \in \Gamma : J_T^*(\boldsymbol{x}) \leq \alpha\}\,, \quad \alpha = \beta\left(1 + \frac{m_l}{M_V} T\right) \tag{6.30}$$

with $m_l, M_V > 0$ from (6.5). Then, $\bar{\boldsymbol{x}}_T^*(T; \boldsymbol{x}_k) \in S_\beta$ for all $\boldsymbol{x}_k \in \Gamma_\alpha$ and $S_\beta \subseteq \Gamma_\alpha$.

This shows the interesting fact that the terminat set is automatically reached for all $\boldsymbol{x}_k \in \Gamma_\alpha$. The following theorem now states the stability properties.

Theorem 6.4 (MPC with terminal cost) *Let the assumptions of Lemma 6.2 hold. Then, the cost satisfies*

$$J_T^*(\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k)) \leq J_T^*(\boldsymbol{x}_k) - \int_0^{\Delta t} l(\bar{\boldsymbol{x}}_T^*(\tau; \boldsymbol{x}_k), \bar{\boldsymbol{u}}_T^*(\tau; \boldsymbol{x}_k))\, \mathrm{d}\tau \quad \forall\, \boldsymbol{x}_k \in \Gamma_\alpha \tag{6.31}$$

and the origin of the system (6.1) in closed-loop is asymptotically stable in the sense of $\lim_{t\to\infty} ||\boldsymbol{x}(t)|| = 0$ *with* Γ_α *lying in the domain of attraction.*

Proof: Lemma 6.2 states that $\bar{\boldsymbol{x}}_T^*(T; \boldsymbol{x}_k) \in S_\beta$ for all $\boldsymbol{x}_k \in \Gamma_\alpha$. Thus, the derivation of the inequality (6.31) as well as the feasibility of (6.28) in step $k+1$ follows the proof of Theorem 6.3. Moreover, $\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k) \in \Gamma_\alpha$ in view of (6.31) and the definition of Γ_α, which ensures that (6.31) also holds in the next step $k+1$. The proof of stability ($\lim_{t\to\infty} ||\boldsymbol{x}(t)|| = 0$) in closed-loop corresponds again to the proof of Theorem 6.1. ■

The definition of the domain of attraction in (6.30) shows that Γ_α can be enlarged by choosing a longer prediction horizon T, see Figure 6.3. This intuitive property is also observed in practice.

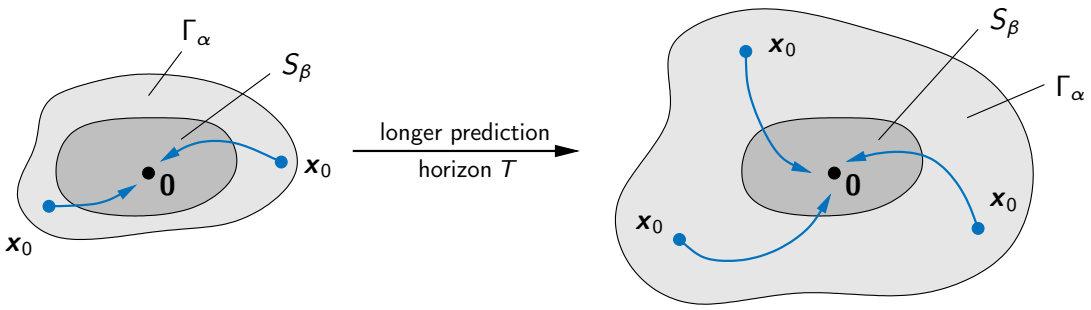


Figure 6.3: Domain of attraction for MPC with terminal cost.

Example 6.1 *We consider the nonlinear system [4]*

$$\begin{aligned} \dot{x}_1 &= x_2 + (\mu + (1-\mu)x_1)u\,, & x_1(0) &= x_{10} \\ \dot{x}_2 &= x_1 + (\mu - 4(1-\mu)x_2)u\,, & x_2(0) &= x_{20} \end{aligned} \tag{6.32}$$

with the state $\boldsymbol{x} = [x_1, x_2]^\mathsf{T}$ *and the constrained control input* $u \in U = [-2, 2]$. *The parameter* $\mu \in (0, 1)$ *is set to* $\mu = 0.5$. *The origin is an unstable equilibrium of the system (6.32) and the linearization around the origin is stabilizable. The integral term in the cost functional (6.28a) is defined by*

$$l(\boldsymbol{x}, u) = 0.5x_1^2 + 0.5x_2^2 + u^2\,. \tag{6.33}$$

The CLF inequality (6.29) is satisfied on the set $S_\beta = \{\boldsymbol{x} \in \mathbb{R}^2 : V(\boldsymbol{x}) \leq \beta = 0.7\}$ *with the terminal cost*

$$V(\boldsymbol{x}) = \boldsymbol{x}^\mathsf{T} \begin{bmatrix} 16.5926 & 11.5926 \\ 11.5926 & 16.5926 \end{bmatrix} \boldsymbol{x} \tag{6.34}$$

and the linear control law $\boldsymbol{q}(\boldsymbol{x}) = [2.118, 2.118]\boldsymbol{x}$. *The left side of Figure 6.4 shows* S_β *as well as the set* Γ_α *for different prediction horizons* T. *For all initial values* $\boldsymbol{x}_k \in \Gamma_\alpha$, *the end point* $\bar{\boldsymbol{x}}_T^*(T; \boldsymbol{x}_k)$ *lies in the terminal region* S_β. *The single sets* Γ_α *are computed numerically and*

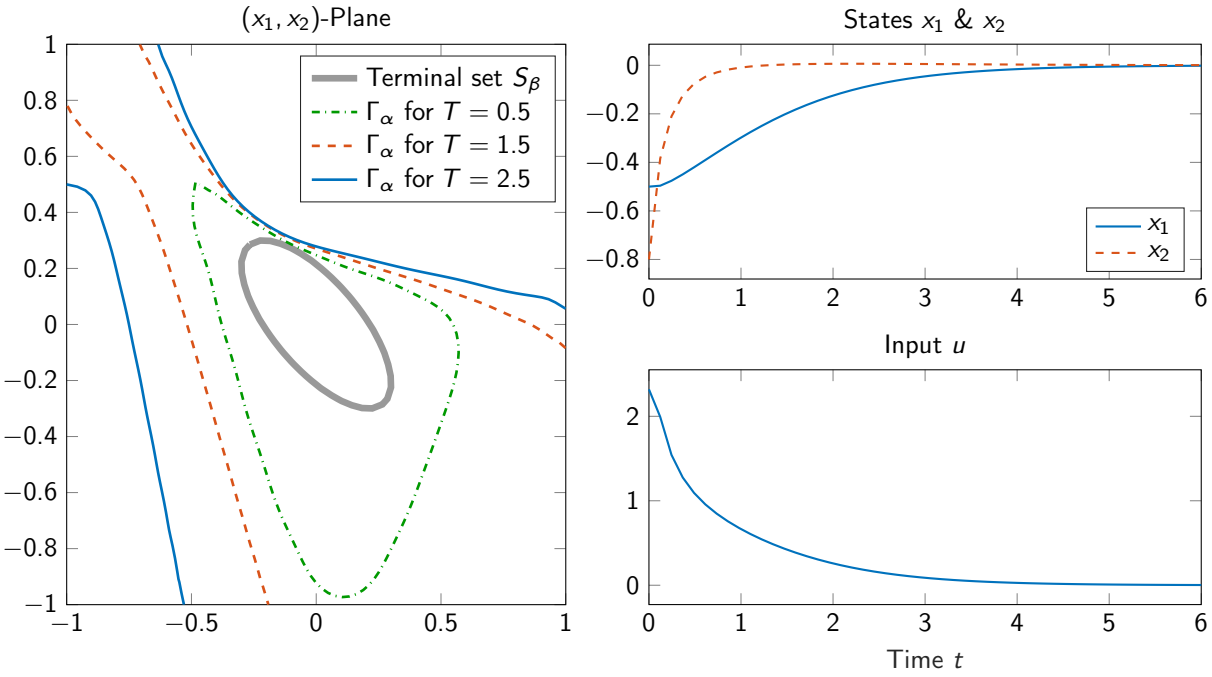


Figure 6.4: Terminal set S_β with $\beta = 0.7$ and domain of attraction Γ_α for different horizon lengths T (Example 6.1).

therefore are a better estimate of the domain of attraction than the conservative definition (6.30). As already pointed out in Lemma 6.2, the shape of S_β and Γ_α reveals that $S_\beta \subseteq \Gamma_\alpha$ holds for all T and that the domain of attraction Γ_α can be enlarged by increasing the prediction horizon. In addition, the right side of Figure 6.4 shows the simulated trajectories in closed-loop for the initial point $\boldsymbol{x}_0 = [-0.5, -0.8]^\mathsf{T}$ and $T = 1.5$.

6.2.5 Sufficiently long horizon

Even if the previous MPC formulation does not require a terminal set constraint, the proof of stability nevertheless requires the terminal cost $V(\boldsymbol{x})$ to satisfy the CLF inequality (6.29). Depending on the horizon length T, this leads to a dominance within the cost functional and thus "falsifies" the qualitative control behavior that is usually tuned by the user via the weights in the integral term. This effect can e.g. by seen in Example 6.1.

On the other hand, we know from Section 6.2.1 that an infinite horizon guarantees stability without terminal cost or terminal constraint. It is therefore reasonable to ask whether a "sufficiently long" horizon can ensure asymptotic stability without having to impose the terminal cost $V(\boldsymbol{x})$ and the CLF inequality (6.29). To answer this question, we consider the following MPC formulation:

$$\min_{\bar{\boldsymbol{u}}(\cdot)} \quad J_T(\boldsymbol{x}_k, \bar{\boldsymbol{u}}) = \int_0^T l(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau))\, \mathrm{d}\tau \tag{6.35a}$$

$$\text{s.t.} \quad \dot{\bar{\boldsymbol{x}}}(\tau) = \boldsymbol{f}(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau))\,, \quad \bar{\boldsymbol{x}}(0) = \boldsymbol{x}_k = \boldsymbol{x}(t_k) \tag{6.35b}$$

$$\bar{\boldsymbol{u}}(\tau) \in U\,, \quad \tau \in [0, T]\,. \tag{6.35c}$$

The investigation of this MPC formulation without terminal cost and terminal constraint is mathematically more complex than the previous cases and therefore requires stronger assump-

tions. This is also the reason for omitting the state constraints $\bar{\boldsymbol{x}}(\tau) \in X$. Moreover, we require the definition of *(local) Lipschitz continuity*.

Definition 6.1 (Local Lipschitz continuity) *A function f is* locally Lipschitz continuous *in $\boldsymbol{x}$ if for every point $\boldsymbol{x}_0$ there exists a neighborhood $B_r(\boldsymbol{x}_0) = \{\boldsymbol{x} \in \mathbb{R}^n : ||\boldsymbol{x} - \boldsymbol{x}_0|| \leq r\}$ with radius $r > 0$ and a* Lipschitz cosnstant *$L > 0$ such that*

$$|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)| \leq L||\boldsymbol{x}_1 - \boldsymbol{x}_2|| \tag{6.36}$$

for all $\boldsymbol{x}_1$, $\boldsymbol{x}_2 \in B_r(\boldsymbol{x}_0)$.

The Lipschitz condition (6.36) can be extended to vector-valued functions $\boldsymbol{f}$ with $||\boldsymbol{f}(\boldsymbol{x}_1) - \boldsymbol{f}(\boldsymbol{x}_2)|| \leq L||\boldsymbol{x}_1 - \boldsymbol{x}_2||$ and using a suitable vector norm. The following theorem states that under certain conditions a sufficiently long horizon T can be found such that the closed-loop system is stabilized.

Theorem 6.5 (MPC with sufficiently long horizon) *Suppose that the problem (6.35) is feasible for all $\boldsymbol{x}_k \in \mathbb{R}^n$ and $T \in [\Delta t, \infty)$. Moreover, assume that*

1. *the linearization of (6.1) around the origin is stabilizable,*
2. *$J_T^*(\boldsymbol{x})$ is continuously differentiable in $\boldsymbol{x}$ and for all $T \in [\Delta t, \infty)$,*
3. *the optimal control law in (6.8) is locally Lipschitz continuous.*

Then, for every compact set $\Gamma_\alpha = \{\boldsymbol{x} \in \mathbb{R}^n : J_\infty^*(\boldsymbol{x}) \leq \alpha\}$ with $\alpha < \infty$ and for each $\delta \in (0, 1)$, there exists some $T_\delta < \infty$ such that for all $T \geq T_\delta$ the cost satisfies

$$J_T^*(\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k)) \leq J_T^*(\boldsymbol{x}_k) - \delta \int_0^{\Delta t} l(\bar{\boldsymbol{x}}_T^*(\tau; \boldsymbol{x}_k), \bar{\boldsymbol{u}}_T^*(\tau; \boldsymbol{x}_k))\, \mathrm{d}t \quad \forall\, \boldsymbol{x}_k \in \Gamma_\alpha \tag{6.37}$$

and the origin of the system (6.1) in closed-loop is exponentially stable with Γ_α lying in the domain of attraction.

Proof: Bellman's principle of optimality allows us to split the optimal cost functional $J_T^*(\boldsymbol{x}_k)$ at point $\boldsymbol{x}_k$ into two parts

$$J_T^*(\boldsymbol{x}_k) = \int_0^{\Delta t} l(\bar{\boldsymbol{x}}_T^*(\tau; \boldsymbol{x}_k), \bar{\boldsymbol{u}}_T^*(\tau; x_k))\, \mathrm{d}\tau + J_{T-\Delta t}^*(\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k))\,. \tag{6.38}$$

Moreover, the inequality

$$J_{T_1}^*(\boldsymbol{x}_k) \leq \int_0^{T_1} l(\bar{\boldsymbol{x}}_{T_2}^*(\tau; \boldsymbol{x}_k), \bar{\boldsymbol{u}}_{T_2}^*(\tau; \boldsymbol{x}_k))\, \mathrm{d}\tau \leq J_{T_2}^*(\boldsymbol{x}_k)$$

shows that $J_{T_1}^*(\boldsymbol{x}_k) \leq J_{T_2}^*(\boldsymbol{x}_k)$ for all $T_1 \leq T_2$. Negating equation (6.38), adding $J_T^*(\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k))$, and using $J_T^*(\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k)) \leq J_\infty^*(\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k))$, we get

$$\begin{aligned} &J_T^*(\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k)) - J_T^*(\boldsymbol{x}_k) \\ &\quad \leq J_\infty^*(\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k)) - J_{T-\Delta t}^*(\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k)) - \int_0^{\Delta t} l(\bar{\boldsymbol{x}}_T^*(\tau; \boldsymbol{x}_k), \bar{\boldsymbol{u}}_T^*(\tau; \boldsymbol{x}_k))\, \mathrm{d}\tau\,. \end{aligned} \tag{6.39}$$

For further investigations, we define the function $\psi_T : X_\alpha \to \mathbb{R}$

$$\psi_T(\boldsymbol{x}) = \begin{cases} \dfrac{J^*_\infty(\boldsymbol{x}) - J^*_{T-\Delta t}(\boldsymbol{x})}{J^*_{\Delta t}(\boldsymbol{x})} & \text{if } \boldsymbol{x} \neq \boldsymbol{0} \\ \limsup\limits_{\boldsymbol{x}\to\boldsymbol{0}} \psi_T(\boldsymbol{x}) & \text{if } \boldsymbol{x} = \boldsymbol{0} \end{cases} \tag{6.40}$$

on the compact set

$$X_\alpha = \left\{ \bar{\boldsymbol{x}}^*_T(\tau; \boldsymbol{x}_k) \,\middle|\, \tau \in [0, \Delta t]\,,\ \boldsymbol{x}_k \in \Gamma_\alpha\,,\ T \in [\Delta t, \infty) \right\} \subset \mathbb{R}^n\,. \tag{6.41}$$

It is obvious for $\boldsymbol{x} \neq \boldsymbol{0}$ that $\psi_T(\boldsymbol{x})$ is continuous and monotonically decreasing for $T \in [\Delta t, \infty)$. It can be shown for the critical point $\boldsymbol{x} = \boldsymbol{0}$ that $\psi_T(\boldsymbol{0})$ is bounded and $\lim_{T\to\infty} \psi_T(\boldsymbol{0}) = 0$ holds [16]. Thus, we can summarize that $\psi_T(\boldsymbol{x})$ is continuous and bounded for all $\boldsymbol{x} \in X_\alpha$ and monotonically decreasing in T with the limit $\lim_{T\to\infty} \psi_T(\boldsymbol{x}) = 0$ for each fixed $\boldsymbol{x} \in X_\alpha$. At this point, we require *Dini's theorem* to proceed.

Lemma 6.3 (Dini's theorem) *Let $f_n : X \to \mathbb{R}$ be a sequence of continuous functions on a compact set X that monotonically converges towards a function $f : X \to \mathbb{R}$. If f is continuous, then the convergence is uniform on X.* [2]

Dini's theorem allows to draw the conclusion that every sequence $\psi_{T_n} : \Gamma \to \mathbb{R}$ with $\Delta t = T_1 < T_2 < \ldots$ converges uniformly towards $\lim_{T\to\infty} \psi_T(\boldsymbol{x}) = 0$. The definition of uniform convergence implies that for every $\varepsilon \in (0, 1)$ there exists some $\hat{T} < \infty$ such that $\psi_T(\boldsymbol{x}) < \varepsilon$ for all $\boldsymbol{x} \in \Gamma_\alpha$. Thus, from (6.40) and for the point $\boldsymbol{x} = \bar{\boldsymbol{x}}^*_T(\Delta t; \boldsymbol{x}_k) \in X_\alpha$ (for $\boldsymbol{x}_k \in \Gamma_\alpha$) it follows that

$$J^*_\infty(\bar{\boldsymbol{x}}^*_T(\Delta t; \boldsymbol{x}_k)) - J^*_{T-\Delta t}(\bar{\boldsymbol{x}}^*_T(\Delta t; \boldsymbol{x}_k)) \leq \varepsilon J^*_{\Delta t}(\bar{\boldsymbol{x}}^*_T(\Delta t; \boldsymbol{x}_k))\,. \tag{6.42}$$

Some estimates are required to proceed from this point. With *Gronwall's lemma* and its inverse formulation [10], we get

$$||\boldsymbol{x}_k|| e^{-\hat{L}\tau} \leq ||\bar{\boldsymbol{x}}^*_T(\tau; \boldsymbol{x}_k)|| \leq ||\boldsymbol{x}_k|| e^{\hat{L}\tau}\,, \quad \tau \in [0, T]\,, \quad \hat{L} = L_{\boldsymbol{f}}(1 + L_{\boldsymbol{\kappa}})\,, \tag{6.43}$$

where $L_{\boldsymbol{f}}, L_{\boldsymbol{\kappa}} > 0$ are Lipschitz constants of the system function $\boldsymbol{f}$ and of the optimal feedback law $\boldsymbol{\kappa}_T$ (let $L_{\boldsymbol{\kappa}}$ hold for all times $T \in [\Delta t, \infty)$). Using the left-side bound in (6.43) and a similar reasoning as in [4], we can show that for all $T \in [\Delta t, \infty)$ there exist constants $m_J, M_J > 0$ such that the following quadratic bounds hold for $J^*_T(\boldsymbol{x}_k)$

$$m_J ||\boldsymbol{x}_k||^2 \leq J^*_T(\boldsymbol{x}_k) \leq M_J ||\boldsymbol{x}_k||^2\,, \quad \boldsymbol{x}_k \in \Gamma_\alpha\,, \quad T \in [\Delta t, \infty)\,. \tag{6.44}$$

With the estimates (6.43) and (6.44), the right side of (6.42) can be bounded as follows

$$\begin{aligned} \varepsilon J^*_{\Delta t}(\bar{\boldsymbol{x}}^*_T(\Delta t; \boldsymbol{x}_k)) &\overset{(6.44)}{\leq} \varepsilon M_J ||\bar{\boldsymbol{x}}^*_T(\Delta t; \boldsymbol{x}_k)||^2 \\ &\overset{(6.43)}{\leq} \varepsilon M_J e^{2\hat{L}\Delta t} ||\boldsymbol{x}_k||^2 \\ &\overset{(6.44)}{\leq} \varepsilon \frac{M_J}{m_J} e^{2\hat{L}\Delta t} J^*_{\Delta t}(\boldsymbol{x}_k) \\ &\leq \varepsilon \frac{M_J}{m_J} e^{2\hat{L}\Delta t} \int_0^{\Delta t} l(\bar{\boldsymbol{x}}^*_T(\tau; \boldsymbol{x}_k), \bar{\boldsymbol{u}}^*_T(\tau; \boldsymbol{x}_k))\, \mathrm{d}\tau\,. \end{aligned} \tag{6.45}$$

[2] The sequence $f_n : X \to \mathbb{R}$ with the limit $f : X \to \mathbb{R}$ is called *uniformly convergent*, if for every $\varepsilon > 0$ there exists an integer N such that for all $n \geq N$ and for all $\boldsymbol{x} \in X$ we have $|f_n(\boldsymbol{x}) - f(\boldsymbol{x})| < \epsilon$.

This shows that (6.39) can be written in the form (6.37) with the constant δ defined by

$$\delta = 1 - \varepsilon \frac{M_J}{m_J} e^{2\hat{L}\Delta t} . \tag{6.46}$$

Since $\varepsilon \in (0,1)$ can be freely chosen, there exists a horizon length T_δ for every $\delta \in (0,1)$ such that (6.37) holds for $T \geq T_\delta$.[3]

To conclude about asymptotic stability, the integral in (6.37) can be lower bounded by

$$\int_0^{\Delta t} l(\bar{\boldsymbol{x}}_T^*(\tau;\boldsymbol{x}_k), \bar{\boldsymbol{u}}_T^*(\tau;\boldsymbol{x}_k))\,\mathrm{d}t \geq J_{\Delta t}^*(\boldsymbol{x}_k) \overset{(6.44)}{\geq} m_J ||\boldsymbol{x}_k||^2 \overset{(6.44)}{\geq} \sigma J_T^*(\boldsymbol{x}_k) \tag{6.47}$$

with $\sigma = m_J/M_J \in (0,1)$. From (6.37), we therefore obtain the estimate $J_T^*(\bar{\boldsymbol{x}}_T^*(\Delta t;\boldsymbol{x}_k)) \leq (1-\delta\sigma) J_T^*(\boldsymbol{x}_k)$ with $1-\delta\sigma \in (0,1)$ or, respectively, by means of the initial condition $J_T^*(\boldsymbol{x}_0) \leq \alpha$

$$J_T^*(\boldsymbol{x}_k) \leq \alpha(1-\delta\sigma)^k \quad \Rightarrow \quad ||\boldsymbol{x}_k|| \overset{(6.44)}{\leq} \sqrt{\frac{J_T^*(\boldsymbol{x}_k)}{m_J}} \leq \sqrt{\frac{\alpha}{m_J}}(1-\delta\sigma)^{k/2} . \tag{6.48}$$

This shows that $||\boldsymbol{x}_k||$ decreases asymptotically for $k \in \mathbb{N}_0^+$. To show stability in the continuous-time sense, it is sufficient to consider the upper bound from (6.43)

$$||\bar{\boldsymbol{x}}_T^*(\tau;\boldsymbol{x}_k)|| \overset{(6.43)}{\leq} ||\boldsymbol{x}_k|| e^{\hat{L}\tau} \overset{(6.48)}{\leq} \sqrt{\frac{\alpha}{m_J}} e^{\hat{L}\Delta t}(1-\delta\sigma)^{k/2} , \quad \tau \in [0, \Delta t). \tag{6.49}$$

It can be shown that there exist constants $q_1, q_2 > 0$ such that the trajectories in closed-loop (6.9) are bounded by $||\boldsymbol{x}(t)|| \leq q_1 e^{-q_2(t-t_0)}$, which implies exponential stability of the origin in closed-loop [17]. ■

Exponential stability as the result of Theorem 6.5 is a stronger statement then asymptotic stability in the sense of $\lim_{t\to\infty} ||\boldsymbol{x}(t)|| = 0$. However, the proof of exponential stability requires the optimal feedback law in (6.8) to be locally Lipschitz.

The exact value of T_δ is problem-specific and must be determined numerically (i.e. in simulations). Nevertheless, Theorem 6.5 provides the interesting result that (under the stated assumptions) there always exists a minimum horizon length $T_\delta < \infty$ such that exponential stability holds for $T \geq T_\delta$.

Example 6.2 *We consider again the MPC problem of Example 6.1 and drop the terminal cost (6.34). The MPC formulation therefore reads*

$$\begin{aligned}
\min_{\bar{u}(\cdot)} \quad & J_T(\boldsymbol{x}_k, \bar{u}) = \int_0^T 0.5\bar{x}_1^2 + 0.5\bar{x}_2^2 + \bar{u}^2\,\mathrm{d}\tau && \text{(6.50a)}\\
\text{s.t.} \quad & \dot{\bar{x}}_1 = \bar{x}_2 + (\mu + (1-\mu)\bar{x}_1)\bar{u} , \quad \bar{x}_1(0) = \bar{x}_{1,k} = x_1(t_k) && \text{(6.50b)}\\
& \dot{\bar{x}}_2 = \bar{x}_1 + (\mu - 4(1-\mu)\bar{x}_2)\bar{u} , \quad \bar{x}_2(0) = \bar{x}_{2,k} = x_2(t_k) && \text{(6.50c)}\\
& \bar{u}(\tau) \in [-2, 2] , \quad \tau \in [0, T] . && \text{(6.50d)}
\end{aligned}$$

Figure 6.5 illustrates the influence of the horizon length T on the MPC formulation (6.50) for the values $T = 1.5$ and $T = 2.5$. The boundaries of the compact subsets $\Gamma_\alpha = \{\boldsymbol{x} \in \mathbb{R}^2 :$

[3] Lipschitz estimates are usually very conservative. Therefore, the explicit computation of δ in (6.46) is a constructive approach to prove the existence of δ, which is rather unsuitable for the MPC design itself.

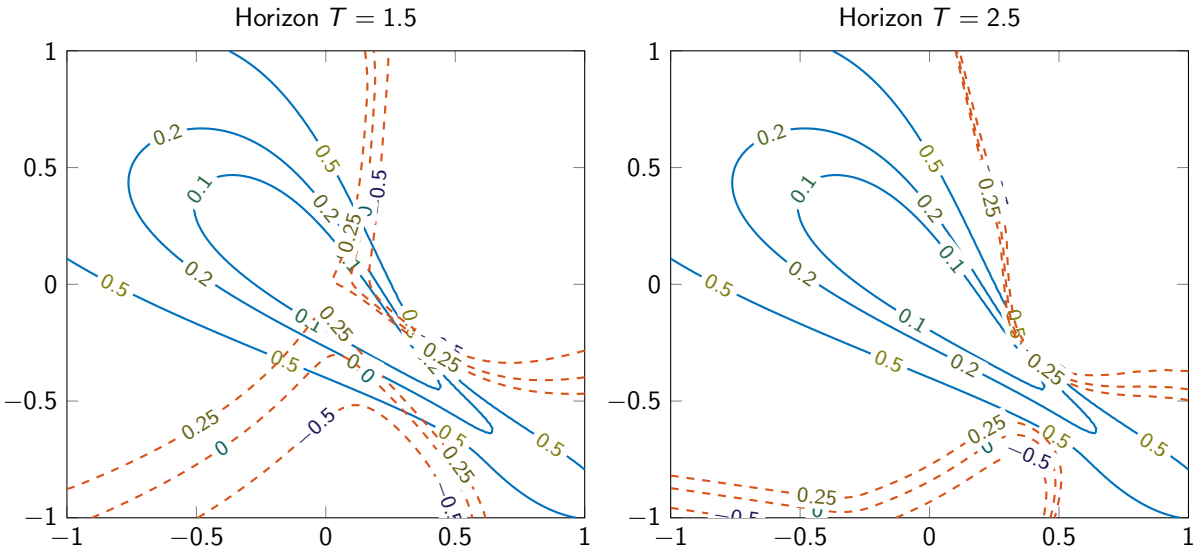


Figure 6.5: Contour lines of the domain of attraction Γ_α (—) and of the reduction factor δ (- -) in the (x_1, x_2)-plane for the prediction horizon $T = 1.5$ and $T = 2.5$ (Example 6.2).

$J^*_\infty(\boldsymbol{x}) \leq \alpha\}$ were computed numerically. In addition, Figure 6.5 plots the contour lines for various values of the factor δ in (6.37).

The contour line for $\delta = 0$ represents the set of points $\boldsymbol{x}_k$ for which the next MPC step does not lead to a decrease of the cost $J^*_T(\bar{\boldsymbol{x}}^*_T(\Delta t; \boldsymbol{x}_k))$. The cases $\delta = 0.25$ and $\delta = -0.5$ correspond to a decrease and increase of the cost functional (6.37). Thus, the set that is enclosed by the contour line $\delta = 0$ corresponds to the domain of attraction for the respective horizon length T. The plots for $T = 1.5$ and $T = 2.5$ in Figure 6.5 illustrate that the stability region increases for a longer prediction horizon.

Figure 6.5 also illustrates that T must be chosen sufficiently large if a given domain Γ_α is to be stabilized. For instance, the contour lines $\delta = 0$ show that stability over the region $\Gamma_{0.2}$ is only achieved for $T = 2.5$.

6.3 Real-time implementation

"All we have to decide is what to do
with the time that is given to us."

Gandalf the Grey

The last two MPC formulations with terminal cost or a sufficiently long horizon are particularly appropriate for an actual process implementation because of the reduced computational effort due to the absence of terminal constraints. Nevertheless, the real-time implementation of MPC is still a challenge, in particular for highly dynamical systems with sampling times in the (sub)millisecond range or if the MPC is implemented on hardware with limited ressources, e.g. for embedded systems.

An intuitive approach for limiting the computational effort is the approximate solution of the optimal control problem, e.g. by using a fixed number of iterations per sampling step. In

the next step, the previous solution can be used to warm-start the optimization method with the intention to successively reduce the suboptimality of the solution over the runtime of the process. This incremental approach significantly differs from the optimal MPC case, where the optimal solution is assumed to be known in each sampling step.

There exist various efficient and suboptimal MPC schemes in the literature with different kinds of terminal constraints and conditions on the optimization algorithm that are not further eloborated at this point. Instead, we will focus on a general suboptimal solution strategy.

6.3.1 Suboptimal solution strategy

Since the main bottleneck for achieving real-time feasibility is the computational effort, we consider an MPC formulation without terminal conditions and state constraints

$$\begin{aligned} \min_{\bar{\boldsymbol{u}}(\cdot)} \quad & J_T(\boldsymbol{x}_k, \bar{\boldsymbol{u}}) = V(\bar{\boldsymbol{x}}(T)) + \int_0^T l(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau)) \, \mathrm{d}\tau && \text{(6.51a)} \\ \text{s.t.} \quad & \dot{\bar{\boldsymbol{x}}}(\tau) = \boldsymbol{f}(\bar{\boldsymbol{x}}(\tau), \bar{\boldsymbol{u}}(\tau)) \,, \quad \bar{\boldsymbol{x}}(0) = \boldsymbol{x}_k = \boldsymbol{x}(t_k) && \text{(6.51b)} \\ & \bar{\boldsymbol{u}}(\tau) \in U \,, \quad \tau \in [0, T] \,. && \text{(6.51c)} \end{aligned}$$

If no terminal cost is to be considered in (6.51a), we set $V(\boldsymbol{x}) = 0$ and $m_V = M_V = 0$ in the quadratic estimate (6.5).

In what follows, we assume the existence of an iterative optimization algorithm that computes a control and state trajectory

$$\bar{\boldsymbol{u}}_T^{(j)}(\tau; \boldsymbol{x}_k) \,, \quad \bar{\boldsymbol{x}}_T^{(j)}(\tau; \boldsymbol{x}_k) \,, \quad \tau \in [0, T] \tag{6.52}$$

in each iteration j. The corresponding (suboptimal) value of the cost functional is written as

$$J_T^{(j)}(\boldsymbol{x}_k) := J_T(\boldsymbol{x}_k, \bar{\boldsymbol{u}}_T^{(j)}) \geq J_T^*(\boldsymbol{x}_k) \,.$$

Moreover, the optimization algorithm is supposed to be linearly convergent, i.e.

$$J_T^{(j+1)}(\boldsymbol{x}_k) - J_T^*(\boldsymbol{x}_k) \leq p\Big(J_T^{(j)}(\boldsymbol{x}_k) - J_T^*(\boldsymbol{x}_k) \Big) \tag{6.53}$$

with $\lim_{j \to \infty} J_T^{(j)}(\boldsymbol{x}_k) = J_T^*(\boldsymbol{x}_k)$. The factor $p \in (0, 1)$ is the linear convergence rate of the optimization algorithm.

The suboptimal solution strategy is supposed to stop the optimization algorithm after a fixed number of iterations

$$j = N$$

and to use the first part of the last suboptimal control trajectory $\bar{\boldsymbol{u}}_T^{(N)}(\tau; \boldsymbol{x}_k)$ as control input

$$\boldsymbol{u}(t_k + \tau) = \bar{\boldsymbol{u}}_T^{(N)}(\tau; \boldsymbol{x}_k) \,, \quad \tau \in [0, \Delta t) \tag{6.54}$$

for the system (6.1). If no model uncertainties or disturbances are present, the next state at time instant t_{k+1} is $\boldsymbol{x}_{k+1} = \bar{\boldsymbol{x}}_T^{(N)}(\Delta t; \boldsymbol{x}_k)$. Overall, the closed-loop trajectories are given by

$$\begin{aligned} \boldsymbol{x}(t) &= \boldsymbol{x}(t_k + \tau) = \bar{\boldsymbol{x}}_T^{(N)}(\tau; \boldsymbol{x}_k) \,, \\ \boldsymbol{u}(t) &= \boldsymbol{u}(t_k + \tau) = \bar{\boldsymbol{u}}_T^{(N)}(\tau; \boldsymbol{x}_k) \,, \quad \tau \in [0, \Delta t) \,, \quad k \in \mathbb{N}_0^+ \,. \end{aligned} \tag{6.55}$$

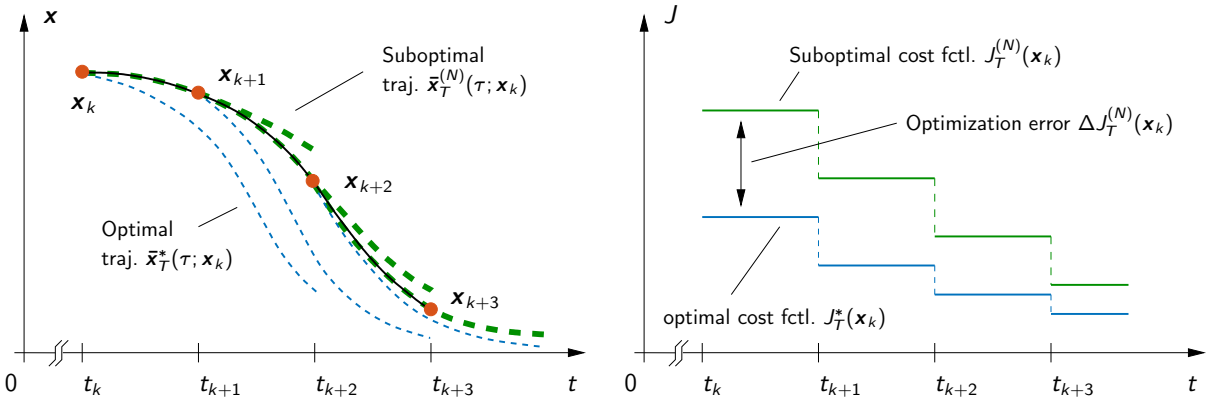


Figure 6.6: Principle of the suboptimal MPC strategy.

The trajectories are suboptimal and therefore lead to an *optimization error*

$$\Delta J_T^{(N)}(\boldsymbol{x}_k) := J_T^{(N)}(\boldsymbol{x}_k) - J_T^*(\boldsymbol{x}_k) \geq 0 \tag{6.56}$$

in each MPC step k. The goal of the suboptimal MPC strategy is to successively reduce the optimization error $\Delta J_T^{(N)}(\boldsymbol{x}_k)$ by using the last control trajectory $\bar{\boldsymbol{u}}_T^{(N)}(\tau; \boldsymbol{x}_k)$ to re-initialize the optimization method in the next step

$$\bar{\boldsymbol{u}}_T^{(0)}(\tau; \boldsymbol{x}_{k+1}) := \bar{\boldsymbol{u}}_T^{(N)}(\tau; \boldsymbol{x}_k)\,, \quad \tau \in [0, T]\,. \tag{6.57}$$

Figure 6.6 illustrates the basic concept behind the suboptimal MPC strategy.

6.3.2 Stability and incremental improvement

For a more compact notation, we introduce the set

$$\mathcal{U}_{[0,T]} = \{\boldsymbol{u}(\cdot) \in L_m^\infty[0, T] : \boldsymbol{u}(t) \in U,\ t \in [0, T]\} \tag{6.58}$$

of all admissible control trajectories.[4] The single iterations of an optimization algorithm $\bar{\boldsymbol{u}}_T^{(j)}(\cdot) \in \mathcal{U}_{[0,T]}$ in general differ from the optimal solution $\bar{\boldsymbol{u}}_T^*(\tau; \boldsymbol{x}_k)$, $\tau \in [0, T]$ of (6.51), which requires to impose more restrictive assumptions than in the previous considerations. In particular, we impose the strict convexity condition

$$||\boldsymbol{u} - \boldsymbol{u}_T^*(\cdot; \boldsymbol{x}_k)||^2_{L^2_m[0,T]} \leq C\,|J_T(\boldsymbol{x}_k, \boldsymbol{u}) - J_T^*(\boldsymbol{x}_k)| \tag{6.59}$$

for some constant $C > 0$. It can be shown that this assumption is always satisfied for linear systems with diagonal-quadratic cost functional.

[4] The *infinite-dimensional* vector space $L_m^p[0, T]$, $1 \leq p < \infty$ is the space of all real-valued functions $u : [0, T] \to \mathbb{R}$ defined on the interval $[0, T]$ that is equipped with the norm

$$||u||_{L^p[0,T]} = \Big(\int_0^T |u(t)|^p\,\mathrm{d}t\Big)^{1/p} < \infty \quad \text{or} \quad ||\boldsymbol{u}||_{L_m^p[0,T]} = \Big(\int_0^T ||\boldsymbol{u}(t)||_2^p\,\mathrm{d}t\Big)^{1/p} < \infty$$

for vector-valued functions $\boldsymbol{u} : [0, T] \to \mathbb{R}^m$. In particular, the supremum norm is defined by $||\boldsymbol{u}||_{L_m^\infty[0,T]} = \sup(||\boldsymbol{u}(t)||_2, t \in [0, T])$.

Theorem 6.6 (Suboptimal MPC) *Suppose the problem (6.51) is feasible for all $\boldsymbol{x}_k \in \mathbb{R}^n$ and that the following conditions are satisfied on a compact set $\Gamma_\alpha = \{\boldsymbol{x} \in \mathbb{R}^n : J_T^*(\boldsymbol{x}) \leq \alpha\}$ with $\alpha < \infty$:*

1. *The optimal control law in (6.8) is locally Lipschitz continuous.*
2. *$J_T(\boldsymbol{x}_k, \boldsymbol{u})$ is twice continuously differentiable in $\boldsymbol{x}_k$ for each $\boldsymbol{u} \in \mathcal{U}_{[0,T]}$.*
3. *The convexity condition (6.59) is satisfied for all $\boldsymbol{u} \in \mathcal{U}_{[0,T]}$ and $\boldsymbol{x}_k \in \Gamma_\alpha$.*
4. *The optimization method used for solving (6.51) satisfies (6.53) with a linear convergence rate $p \in (0,1)$ for all $\boldsymbol{x}_k \in \Gamma_\alpha$.*
5. *There exists a constant $\delta \in (0,1]$ such that*

$$J_T^*(\bar{\boldsymbol{x}}_T^*(\Delta t; \boldsymbol{x}_k)) \leq J_T^*(\boldsymbol{x}_k) - \delta \int_0^{\Delta t} l(\bar{\boldsymbol{x}}_T^*(\tau; \boldsymbol{x}_k), \bar{\boldsymbol{u}}_T^*(\tau; \boldsymbol{x}_k))\, \mathrm{d}\tau \quad \forall\, \boldsymbol{x}_k \in \Gamma_\alpha\,. \tag{6.60}$$

Then, there exists a minimum number of iterations $\hat{N}$ and a constant $\Delta\hat{J} > 0$ such that for all $N \geq \hat{N}$ and all points $\boldsymbol{x}_0 \in \Gamma_\alpha$ with initial optimization error $\Delta J_T^{(0)}(\boldsymbol{x}_0) \leq \Delta\hat{J}p^{-N}$ the origin of the system (6.1) in closed-loop is exponentially stable and the optimization error (6.56) decays exponentially.

Proof: Similar to the procedure in [14, 11], it can be shown under the given assumptions that there exist constants $a \in (0,1]$ and $b, c, d, e > 0$ such that $J_T^*(\boldsymbol{x}_k)$ and $\Delta J_T^{(N)}(\boldsymbol{x}_k)$ satisfy the estimates

$$\begin{aligned} J_T^*(\boldsymbol{x}_{k+1}) &\leq (1-a) J_T^*(\boldsymbol{x}_k) + b\sqrt{J_T^*(\boldsymbol{x}_k)}\sqrt{\Delta J_T^{(N)}(\boldsymbol{x}_k)} + c\Delta J_T^{(N)}(\boldsymbol{x}_k) \\ \Delta J_T^{(N)}(\boldsymbol{x}_{k+1}) &\leq p^N(1+d)\Delta J_T^{(N)}(\boldsymbol{x}_k) + p^N e J_T^*(\boldsymbol{x}_k) \end{aligned} \tag{6.61}$$

with $\boldsymbol{x}_{k+1} = \bar{\boldsymbol{x}}_T^{(N)}(\Delta t; \boldsymbol{x}_k)$ denoting the state in the next MPC step. By induction it can be shown that for the initial conditions

$$J_T^*(\boldsymbol{x}_0) \leq \alpha\,, \quad \Delta J_T^{(N)}(\boldsymbol{x}_0) \leq \Delta\hat{J} = \gamma^2\alpha \quad \text{mit} \quad \gamma = \frac{-b + \sqrt{b^2 + 2ac}}{2c} \tag{6.62}$$

and a "sufficiently large" convergence rate

$$p^N \leq \frac{(1 - \frac{a}{2})\gamma^2}{(1+d)\gamma^2 + e} \tag{6.63}$$

the estimates

$$J_T^*(\boldsymbol{x}_k) \leq \left(1 - \frac{a}{2}\right)^k \alpha\,, \quad \Delta J_T^{(N)}(\boldsymbol{x}_k) \leq \gamma^2 \left(1 - \frac{a}{2}\right)^k \alpha \tag{6.64}$$

hold. The upper bound for p^N corresponds to a minimum number $\hat{N}$ of MPC iterations per MPC step. If this is satisfied, then $J_T^*(\boldsymbol{x}_k)$ as well as $\Delta J_T^{(N)}(\boldsymbol{x}_k)$ decay exponentially in discrete-time. The proof of exponential stability in closed-loop proceeds similar to the proof of Theorem 6.5, also see [14]. ■

Although the MPC formulation (6.51) could be solved with any of the methods presented in Chapter 5 (Section 5.6 and 5.7), the gradient method (see Section 5.6.4 and Table 5.1) is particularly favorable due to the formulation without terminal constraints. Moreover, it is shown in [6] that the gradient method satisfies the linear convergence property (6.53) under certain convexity and regularity assumptions.

As mentioned in Section 5.6.4, the input constraints (6.51c) can be considered via projection and the line search of the gradient method in Table 5.1 can be realized with an adaptive or explicit approach with a constant computation load in each MPC step [12]. The gradient method is also implemented in the MPC toolbox GRAMPC that uses an augmented Lagrangian formulation with a gradient-based minimization in the inner loop [7].

A further possibility is to solve the optimality conditions resulting from Pontryagin's maximum principle (Section 5.5.1) with a *fixed-point iteration scheme* that can be implemented with minimal computational footprint [11].

6.3.3 Application example

An illustrative MPC example is the laboratory-scale overhead crane in Figure 6.7 with five degrees-of-freedom (DOF) [13]. The crane consists of a fixed tower of height h, a rotating jib with a movable trolley and a cable with an attached load of mass m. The jib rotates about the vertical axis of the tower with the angle ϕ_1. The radial motion of the trolley is measured from the tower axis to the trolley's center of mass in terms of the translational DOF s_1. A cable winch fixed to the movable trolley together with a gimbal allow for the vertical actuation and an arbitrary pendulum motion of the cable with the attached load. The vertical motion is described in terms of the translational DOF s_2 measured from the gimbal to the load. The rotation of the gimbal is considered as a composition of two successive rotations with the rotational DOF ϕ_2 and ϕ_3. In summary, the crane is an underactuated mechanical system with five DOFs and three controls.

The equations of motion can be derived via Lagrange's formalism. Under the assumption that the actuated DOF ϕ_1, s_1, and s_2 are controlled by sufficiently fast cascaded control loops, the dynamics of the system are described by the second–order differential equations

$$\ddot{s}_1 = u_1 \tag{6.65a}$$

$$\ddot{s}_2 = u_2 \tag{6.65b}$$

$$\ddot{\phi}_1 = u_3 \tag{6.65c}$$

$$\begin{aligned}\ddot{\phi}_2 = \frac{1}{s_2 \cos\phi_3}\Big(&- 2\dot{s}_2\dot{\phi}_1 \cos\phi_2 \sin\phi_3 - 2\dot{s}_2\dot{\phi}_2 \cos\phi_3 - 2\dot{\phi}_1\dot{\phi}_3 s_2 \cos\phi_2 \cos\phi_3 \\ &+ 2s_2\dot{\phi}_2\dot{\phi}_3 \sin\phi_3 - s_1\dot{\phi}_1^2 \cos\phi_2 + s_2\dot{\phi}_1^2 \sin\phi_2 \cos\phi_2 \cos\phi_3 \\ &- g \sin\phi_2 + \cos\phi_2 u_1 - s_2 \cos\phi_2 \sin\phi_3 u_3\Big)\end{aligned} \tag{6.65d}$$

$$\begin{aligned}\ddot{\phi}_3 = \frac{1}{s_2}\Big(&- 2\dot{s}_1\dot{\phi}_1 \cos\phi_3 - 2\dot{s}_2\dot{\phi}_3 + 2\dot{s}_2\dot{\phi}_1 \sin\phi_2 + 2s_2\dot{\phi}_1\dot{\phi}_2 \cos\phi_2 \cos^2\phi_3 \\ &+ s_1\dot{\phi}_1^2 \sin\phi_2 \sin\phi_3 - s_2\dot{\phi}_2^2 \sin\phi_3 \cos\phi_3 + s_2\dot{\phi}_1^2 \cos^2\phi_2 \sin\phi_3 \cos\phi_3 \\ &- g \cos\phi_2 \sin\phi_3 - \sin\phi_2 \sin\phi_3 u_1 + (s_2 \sin\phi_2 - s_1 \cos\phi_3) u_3\Big)\end{aligned} \tag{6.65e}$$

independent of the load mass m. The accelerations $\ddot{s}_1$, $\ddot{s}_2$, $\ddot{\phi}_1$ of the actuated DOF denote the control inputs of the crane system that are constrained by

$$\begin{aligned}u_1 &\in [u_1^-, u_1^+] = [-2, +2]\ \mathrm{m/s^2} \\ u_2 &\in [u_2^-, u_2^+] = [-2, +2]\ \mathrm{m/s^2} \\ u_3 &\in [u_3^-, u_3^+] = [-2, +2]\ \mathrm{rad/s^2}\ .\end{aligned} \tag{6.66}$$

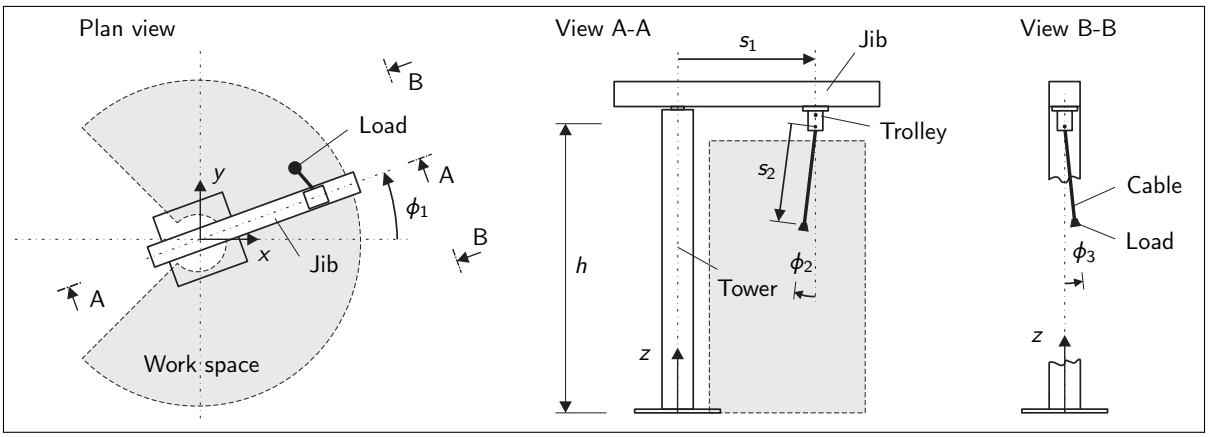


Figure 6.7: Schematic drawing of the overhead crane five degrees-of-freedom [13].

These values are obtained from the maximum motor currents with additional safety margins. The equations of motion (6.65) can be written in the general nonlinear form

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}), \quad \boldsymbol{x}(0) = \boldsymbol{x}_0$$

with the state and input vectors

$$\boldsymbol{x} = [s_1, s_2, \phi_1, \phi_2, \phi_3, \dot{s}_1, \dot{s}_2, \dot{\phi}_1, \dot{\phi}_2, \dot{\phi}_3]^\mathrm{T}, \quad \boldsymbol{u} = [u_1, u_2, u_3]^\mathrm{T}. \tag{6.67}$$

The dynamical model of the crane therefore represents a nonlinear multiple–input system with $m = 3$ controls and $n = 10$ state variables.

The test scenario for the laboratory crane is a setpoint change from the initial position $\boldsymbol{x}_0$ to a desired one $\boldsymbol{x}_{des}$ with

$$\begin{aligned} \boldsymbol{x}_0 &= [0.7\ \mathrm{m}, 0.7\ \mathrm{m}, -60^\circ, 0, 0, 0, 0, 0, 0, 0]^\mathrm{T} \\ \boldsymbol{x}_{des} &= [0.2\ \mathrm{m}, 0.25\ \mathrm{m}, 60^\circ, 0, 0, 0, 0, 0, 0, 0]^\mathrm{T}. \end{aligned} \tag{6.68}$$

The setpoint change describes a turn of the jib about ϕ_1 with a simultaneous change of the trolley position s_1 and the cable length s_2. The cost functional (6.51a) to accomplish the setpoint change is chosen as the quadratic form

$$J_T(\boldsymbol{x}_k, \bar{\boldsymbol{u}}) = \Delta\bar{\boldsymbol{x}}^\mathrm{T}(T)\, \boldsymbol{P}\, \Delta\bar{\boldsymbol{x}}(T) + \int_0^T \Delta\bar{\boldsymbol{x}}^\mathrm{T}(\tau)\, \boldsymbol{Q}\, \Delta\bar{\boldsymbol{x}}(\tau) + \bar{\boldsymbol{u}}^\mathrm{T}(\tau)\, \boldsymbol{R}\, \bar{\boldsymbol{u}}(\tau)\, \mathrm{d}\tau \tag{6.69}$$

with $\Delta\bar{\boldsymbol{x}} = \bar{\boldsymbol{x}} - \boldsymbol{x}_\mathrm{d}$ and the diagonal matrices (neglecting SI units)

$$\begin{aligned} \boldsymbol{P} &= \mathrm{diag}(\,10, 10, 10, 10, 10, 1, 1, 1, 1, 1\,) \\ \boldsymbol{Q} &= \mathrm{diag}(\,1, 1, 1, 1, 1, 0.1, 0.1, 0.1, 1, 1\,) \\ \boldsymbol{R} &= \mathrm{diag}(\,0.01, 0.01, 0.01\,). \end{aligned}$$

The prediction horizon and the sampling time are set to $T = 1.5\,\mathrm{s}$ and $\Delta t = 2\,\mathrm{ms}$.

The suboptimal MPC scheme is implemented with the toolbox GRAMPC that is based on an augmented Lagrangian formulation with a projected gradient method for the inner minimization problem. The canonical equations (see Table 5.1) within GRAMPC are solved with the second-order Heun method on an equidistant discretization of the time interval $[0, T]$ with 30 grid points.

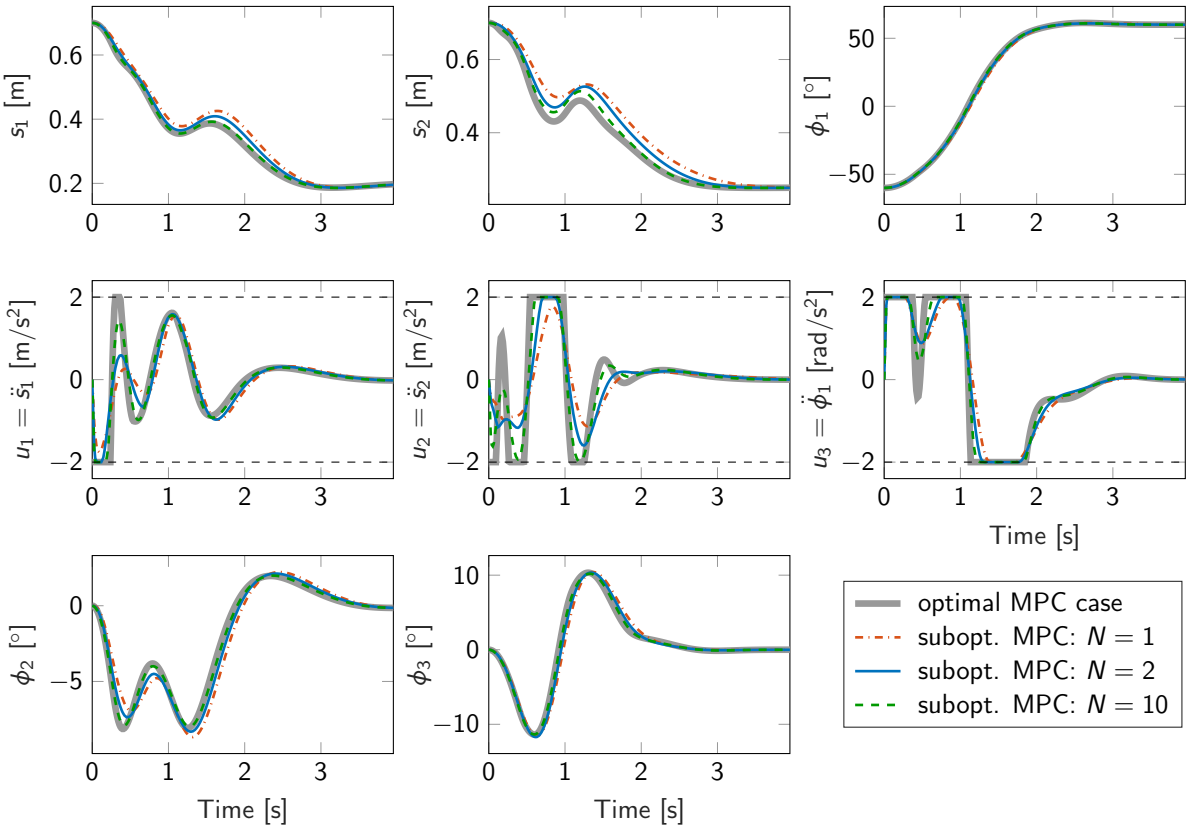


Figure 6.8: MPC simulation results for the setpoint change (6.68) of the overhead crane [13].

Figure 6.8 shows the simulation results for the setpoint change (6.68) with different numbers of iterations per MPC step N. The optimal MPC case is shown as reference. A good control performance can already be obtained for one iteration ($N = 1$) per sampling step. Increasing N mainly affects the control trajectories that show a more aggressive behavior and better exploitation of the constraints (6.66). It can also be seen that the trajectories approach the optimal solution for increasing values of N.

Figure 6.9 also shows the time profiles of the cost functional $J_T^{(N)}(\boldsymbol{x}_k)$ and optimization error $\Delta J_T^{(N)}(\boldsymbol{x}_k)$ that was introduced in (6.56). Both $J_T^{(N)}(\boldsymbol{x}_k)$ and $\Delta J_T^{(N)}(\boldsymbol{x}_k)$ decay exponentially, which shows the exponential stability in closed-loop and the incremental improvement of the suboptimal MPC scheme [14, 12].

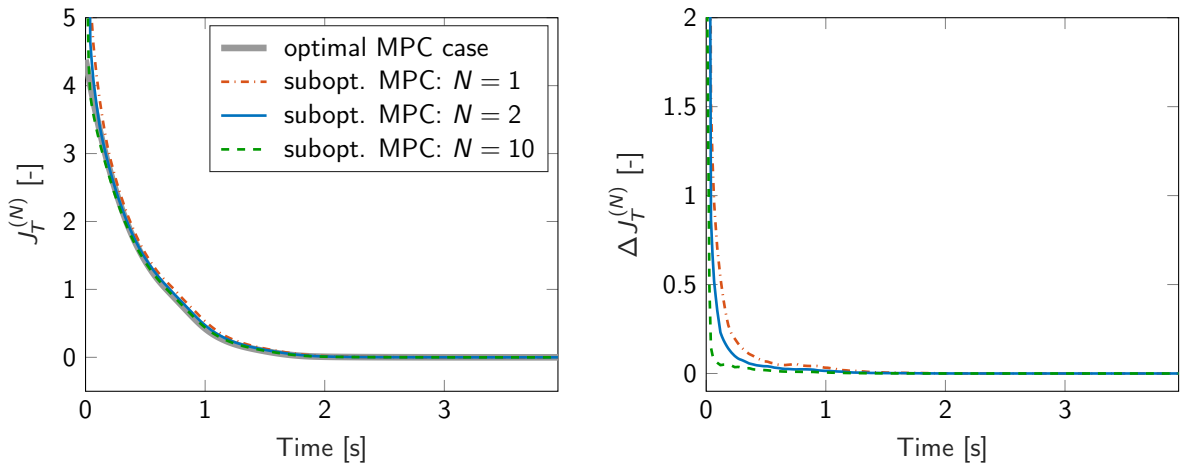


Figure 6.9: Cost $J_T^{(N)}(\boldsymbol{x}_k)$ and optimization error $\Delta J_T^{(N)}(\boldsymbol{x}_k)$ corresponding to Figure 6.8.

In addition, Table 6.1 lists the computation times of GRAMPC in terms of N as well as the standard deviation as a measure of the constant CPU load and real-time feasibility. The CPU results are obtained on a MacBook Pro with Intel Core i9 CPU.

Gradient steps N	CPU time [ms]	
	avg.	std. dev.
1	0.012	0.002
2	0.019	0.003
3	0.026	0.004
5	0.040	0.005
10	0.079	0.007

Table 6.1: Computation times of GRAMPC for the overhead crane.

6.4 Software overview

Several MPC toolboxes are available today that are either open source or commercial. The following lines are a non-exhaustive selection of MPC toolboxes for linear and nonlinear systems. Further information can e.g. be found in the survey paper [9].

MPC toolboxes for linear systems

- CVXGEN (commercial, free for academic use)
  `https://cvxgen.com/`
- FiOrdOs (open source licence)
  `http://fiordos.ethz.ch/`
- Matlab MPC Toolbox (commercial)
  `https://de.mathworks.com/products/mpc.html`
- μAO-MPC (free)
  `http://ifatwww.et.uni-magdeburg.de/syst/muAO-MPC/`

MPC toolboxes for nonlinear systems

- ACADO (open source licence)
  `http://acado.github.io/`
- C/GMRES (free)
  `http://www.ids.sys.i.kyoto-u.ac.jp/~ohtsuka/code/`
- GRAMPC (open source licence)
  `https://sourceforge.net/projects/grampc/`
- HQP (open source licence)
  `http://hqp.sourceforge.net`
- VIATOC (open source licence)
  `https://sourceforge.net/projects/viatoc/`

6.5 References

[1] L.D. Berkovitz. *Optimal Control Theory*. New York: Springer, 1974.

[2] E. Camacho and C. Alba. *Model Predictive Control*. 2nd. London: Springer, 2004.

[3] H. Chen. *Stability and Robstness Considerations in Nonlinear Model Predictive Control*. Vol. 674. Düsseldorf: Fortschritt-Berichte VDI Reihe 8, 1997.

[4] 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.

[5] M. Diehl, H.J. Ferreau, and N. Haverbeke. "Efficient numerical methods for nonlinear MPC and moving horizon estimation". In: *Nonlinear Model Predictive Control - Towards New Challenging Applications*. Ed. by L. Magni, D.M. Raimondo, and F. Allgöwer. 2009, pp. 391–417.

[6] J.C. Dunn. "On L^2 conditions and the gradient projection method for optimal control problems". In: *SIAM Journal on Control and Optimization* 34.4 (1996), pp. 1270–1290.

[7] T. Englert, A. Völz, F. Mesmer, S. Rhein, and K. Graichen. "A software framework for embedded nonlinear model predictive control using a gradient-based augmented Lagrangian approach (GRAMPC)". In: *Optimization and Engineering* 20.3 (2019), pp. 769–809. DOI: 10.1007/s11081-018-9417-2.

[8] R. Findeisen. *Nonlinear Model Predictive Control: A Sampled-Data Feedback Perspective*. Vol. 1087. http://dx.doi.org/10.18419/opus-4059. Fortschritt-Berichte VDI Reihe 8, 2006.

[9] R. Findeisen, K. Graichen, and M. Mönnigmann. "Eingebettete Optimierung in der Regelungstechnik – Grundlagen und Herausforderungen". In: *at – Automatisierungstechnik* 66.11 (2018), pp. 877–902.

[10] H.E. Gollwitzer. "A note on a functional inequality". In: *Proc. American Mathematical Society* 23.3 (1969), pp. 642–647.

[11] K. Graichen. "A fixed-point iteration scheme for real-time model predictive control". In: *Automatica* 48.7 (2012), pp. 1300–1305.

[12] K. Graichen, M. Egretzberger, and A. Kugi. "Ein suboptimaler Ansatz zur schnellen modellprädiktiven Regelung nichtlinearer Systeme". In: *Automatisierungstechnik* 58 (2010), pp. 447–456.

[13] K. Graichen, M. Egretzberger, and A. Kugi. "Suboptimal model predictive control of a laboratory crane". In: *Proc. 8th IFAC Symposium "Nonlinear Control Systems" (NOLCOS)*. Bologna (Italy), Sept. 2010, pp. 397–402.

[14] K. Graichen and A. Kugi. "Stability and incremental improvement of suboptimal MPC without terminal constraints". In: *IEEE Transactions on Automatic Control* 55.11 (2010), pp. 2576–2580.

[15] L. Grüne and J. Pannek. *Nonlinear Model Predictive Control: Theory and Algorithms*. London: Springer, 2011.

[16] A. Jadbabaie, J. Primbs, and J. Hauser. "Unconstrained receding horizon control with no terminal cost". In: *Proc. American Control Conference (ACC)*. Arlington, VA, June 2001, pp. 3055–3060.

[17] H.K. Khalil. *Nonlinear Systems*. 2nd. Prentice Hall, 1996.

[18] E.B. Lee and L. Markus. *Foundations of Optimal Control*. New York: Wiley, 1967.

[19] D. Limon, T. Alamo, F. Salas, and E.F. Camacho. "On the stability of constrained MPC without terminal constraint". In: *IEEE Transaction on Automatic Control* 51.5 (2006), pp. 832–836.

[20] D.Q. Mayne, J.B. Rawlings, C.V. Rao, and P.O.M. Scokaert. "Constrained model predictive control: stability and optimality". In: *Automatica* 36.6 (2000), pp. 789–814.