Technische Universität München

Ingenieurfakultät Bau Geo Umwelt

Lehrstuhl für Statik

GRADIENT DESCENT AKIN METHOD

Long Chen

Vollständiger Abdruck der von der Ingenieurfakultät Bau Geo Umwelt der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs

genehmigten Dissertation.

Die Dissertation wurde am 01.04.2021 bei der Technischen Universität München eingereicht und durch die Ingenieurfakultät Bau Geo Umwelt am 21.07.2021 angenommen.

Schriftenreihe des Lehrstuhls für Statik TU München

Band 48

Long Chen

GRADIENT DESCENT AKIN METHOD

München 2021

Abstract

This thesis is a story about the formula

$$\mathbf{s}_\zeta = -\frac{\nabla f(x)}{|\nabla f(x)|} - \zeta \frac{\nabla g(x)}{|\nabla g(x)|}.$$

Zusammenfassung

$$\mathbf{s}_\zeta = -\frac{\nabla f(x)}{|\nabla f(x)|} - \zeta \frac{\nabla g(x)}{|\nabla g(x)|}.$$

Acknowledgments

First of all, I would like to express my sincere thanks to Prof. Kai-Uwe Bletzinger for introducing me to the fascinating field of shape optimization, which he himself has been passionate about for already decades. His constant support and the academic freedom he has provided over the last six years have been invaluable for me.

I sincerely thank Prof. Ernst Rank and Prof. Michael Stingl for their interest in my work and their time and effort in reviewing my thesis. I am grateful to Prof. Fabian Duddeck for chairing the board of examiners.

I thank all colleagues at the Chair of Structural Analysis, particularly, my office colleagues Armin Geiser and Anna Bauer, for the friendly working atmosphere and the wonderful collaboration. I thank Dr. Wüchner for the many fruitful discussions in Digital Twins and I thank Ann-Kathrin Goldbach for her support in proofreading this thesis.

I thank all colleagues at the Bavarian Graduate School of Computational Engineering (BGCE), especially Nina Korshunova, Tino Bog, Tobias Neckel, and Alexander Ditter, with whom I shared many great moments in the BGCE activities. The wholehearted commitment of BGCE professors to our students is enlightening for me. I appreciate the working experiences as one of the coordinators.

Many thanks to Prof. Nicolas Gauger and Dr. Florian Feppon for the fruitful discussions in shape and topology optimization, and to Dr. Jian Cui for the many exchanges beyond optimization.

My work is partially funded by a subproject of SPP 1886 by the German Research Foundation (DFG), the funding is gratefully appreciated.

Finally, I want to express my deepest gratitude to my family: my wife, my son, and my parents, for their love and support. This work is dedicated to them.

Long Chen
Munich, August 2021

CONTENTS

CHAPTER 1

INTRODUCTION

Whenever there is a material, there may exist a shape, and there are certainly forces. This is, however, not an obvious thing at all—it took humanity hundreds of thousands of years to realize this, until Isaac Newton. However, the beauty somehow hidden in the deep relationship of shape and force has been receptive to us already long ago. Just think about the richness of ancient constructions in the history of our civilization.

1.1 Shape and force

Many things we see, be it a star in the universe, a rock on the seashore, or a flag in the wind, their shape is not only a simple representation of the underlying material, but also of the forces acting on it.

It is well-known that people en masse favor the same shapes. *But what makes a shape elegant?* Unfortunately, this question is yet too difficult to be answered. However, the story here may begin with a particular shape, and most people do find it beautiful.

Catenary and hyperbolic cosine function

A catenary, in geometry, is the curve of a hanging chain under self-weight loading and is only fixed-supported at its ends.

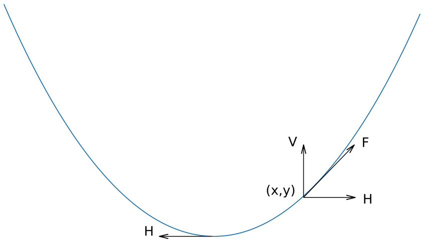

Figure 1.1: Shape and forces of a catenary

In the language of mathematics, a catenary enjoys its own elegance by being expressed by the hyperbolic cosine function,

$$y = \frac{a}{2}\left(e^{\frac{x}{a}} + e^{-\frac{x}{a}}\right) = a\cosh(\frac{x}{a}), \tag{1.1}$$

where $a = \frac{H}{\rho g}$ appears twice with the horizontal force H, the density ρ, and the acceleration of gravity g. The formula (1.1) is obtained by solving a differential equation that is formulated by the equilibrium of forces.

In civil engineering, it was used long ago in the construction of suspension bridges because of its efficient material usage. In the late 19th century, it was Antoni Gaudí who gave real splendor to the catenary by integrating it into many of his works, including the most famous Sagrada Família and Casa Milà. Certainly, it was long known that an optimal arch has the shape of an inverted catenary curve. It may still be speculated that the formula of the catenary had impressed Gaudí when he was thoroughly studying geometry at his young age.

Catenoids, minimal surfaces, and soap films

When rotating a catenary about its directrix, one gets a catenoid. The catenoid is the first known non-trivial minimal surface and has attracted many mathematical studies since its discovery. In engineering design, not surprisingly, it has inspired new constructions both because of its optimal material usage and its shape elegance. Using soap films, Frei Otto was able to create different minimal surfaces and use them to design lightweight structures, particularly, tensile and membrane structures.

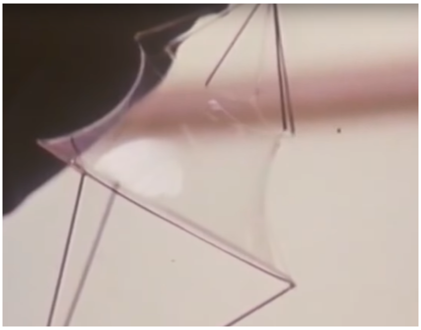

Figure 1.2: Modeling a catenoid with soap skin by Frei Otto (Otto Film [69]).

Munich Olympic roof (1972)

With the ingenious modeling technique with soap films, Otto designed many remarkable structures. This technique, however, reached its limit when it came to the design and construction of the Munich Olympic roof (1972). The roof was simply too large and the handicraft modeling technique lacked a proper dimensional accuracy. The difficulty was solved just in time by Linkwitz and Schek with their newly invented numerical method based on least squares, and the finite element simulation provided by Argyris (Tomlow [86]). The Munich Olympic roof is thus one of the first built structures to use computational methods for modeling, simulation, and optimization in engineering design.

1.2 Computational methods for modeling, simulation, and optimization

Today, computational methods are indispensable in engineering design. They are used to model and simulate buildings, bridges, and tunnels in the event of a disaster. An airfoil is designed and optimized with computational methods before a wing-prototype is built and sent to the wind tunnel experiment. Extensive computational crash simulations are performed in car industries before actual crash tests are conducted. The core of computational methods are mathematical models that describe the underlying physics and computer methods that solve the posed problem reliably and efficiently. Therefore, mathematicians, computer scientists, and engineers are able to work closer than ever before.

Furthermore, computational methods opened up the possibility for new engineering designs that could not have been imagined before. There are (at least) three contributing factors:

1. Advances in numerical methods that are able to model and simulate increasingly complex engineering problems.

2. Advances in the hardware development that provide the computing power needed for the solution of large and difficult problems.

3. Advances in computational optimization methods that can be combined seamlessly with the computational modeling and simulation techniques.

Taking shape optimization as an example: with computational methods, we are not only able to find an optimal shape that has a minimal surface; but we can consider a variety of different design criteria, such as mass, strain energy, deformation, natural frequency, and many others. Finding an optimal shape for a complex system that has, for example, minimal strain energy would have been impossible with experience-driven trial and error designs, and no soap-films-like modeling technique exists for such a task.

1.3 Shape optimization by the gradient descent method

In this thesis, we consider high-fidelity shape optimization that exploits the largest shape space possible. An introduction and review of shape optimization are given in this section.

Abstraction of unconstrained shape optimizations

First, an abstraction of unconstrained shape optimization problems is introduced that closely follows Haslinger et al. [33], and we refer to the same literature for more rigorous discussions. A formulation of any shape optimization problem shall start with a family $\mathcal{O}$ of admissible domains, which contains all possible candidates among which an optimal shape is sought. Let $\Omega \in \mathcal{O}$, and in any Ω we solve a well-posed *state problem* that describes the behavior of a physical system represented by Ω. Further, we define a mapping u that with any Ω associates an element $u(\Omega) \in V(\Omega)$ that is the solution of the *state problem*, i.e.,

$$u : \Omega \mapsto u(\Omega) \in V(\Omega). \tag{1.2}$$

Finally, let $J : (\Omega, y) \mapsto J(\Omega, y) \in \mathbb{R}, \Omega \in \mathcal{O}, y \in V(\Omega)$, be an *objective functional.* Then, an abstract unconstrained shape optimization problem reads

$$\begin{cases} \text{Find } \Omega^\star \in \mathcal{O}, \\ \text{such that } J(\Omega^\star, u(\Omega^\star)) \leq J(\Omega, u(\Omega)) \ \ \forall \Omega \in \mathcal{O}. \end{cases} \tag{1.3}$$

Note, problem (1.3) indicates a global optimal solution for a shape optimization problem. In practice, shape optimizations are often nonconvex. In such cases, finding a local optimal solution can be satisfactory.

Literature review

Shape optimization has a long history that may date back to the brachistochrone curve and roots in the calculus of variations. Another profound work is Hadamard's boundary variation method that constitutes one of the foundations of modern gradient-based shape optimizations. Classical introductions to shape optimization can be found in Haslinger et al. [33] and Sokolowski et al. [83]. In computational mechanics, shape optimization is considered a subset of structural optimization (Haftka et al. [31]). It is characterized by a very large or even infinite number of design variables that describe the varying boundary geometry in an optimization process, whereby one or more structural performances are optimized.

Shape optimization is distinct from another well-known structural optimization problem: topology optimization (Bendsoe et al. [13]). Compared to shape optimization, topology optimization removes smoothness and topological constraints (Allaire [3]), resulting in different optimization formulations. Many topology optimization problems can be formulated in an (equivalent) convex optimization problem, while shape optimizations are typically nonlinear, and often nonconvex (Hoppe et al. [38]). This difference partially contributes to the fact that there are successful implementations of interior-point method (IPM) for large-scale[1] topology optimization problems (Jarre et al. [41], Kennedy [43], Kocvara et al. [48], Maar et al. [57]), but only a few works have presented a shape optimization that uses an IPM as the optimizer (Antil et al. [6], Herskovits et al. [35]). In the latter works, the size of the shape optimization problem is only moderate so that the power of IPM is not fully exploited. One of the most successful methods for nonlinear topology optimization is the method of moving asymptotes (MMA) that was introduced by Svanberg in 1987 (Svanberg [85]). In each iteration, MMA generates and solves an approximated convex problem related to the original one. For shape optimization, however, there is as yet no literature that discusses a large-scale problem using MMA.

A notable difficulty for shape optimization is the computation of shape Hessians, which are complex objects even for moderate problems. Analysis of aerodynamic optimization in Arian et al. [7] shows that shape Hessians are ill-conditioned for three-dimensional problems. Recently, several works compute approximated shape Hessians and use a Newton-based method for the design optimization (Schillings et al. [76], Schmidt et al. [78]). For some disciplines, such as computational fluid dynamics, even the computation of shape gradient can be a challenge. See for example Albring et al. [1] and Reuther et al. [73], which are actively undergoing investigation.

Shape optimization is a subject frequently considered in multidisciplinary design optimization (MDO) (Haftka et al. [32]). To design complex engineering systems, MDO considers multiple disciplines and their interactions. For shape optimization, ongoing efforts are devoted to the computation of the shape gradient for coupled disciplines, for example, for steady-state coupled problems (Kenway et al. [44], Najian Asl et al. [65]) and for transient coupled problems (Korelc [49]). While the development for the coupled-gradient is challenging, the reward is accurate derivatives

[1] In this work, we refer large-scale optimizations to problems that have a high dimensional variable space.

and massive reductions in computational cost, which are essential for large-scale optimizations (Martins et al. [58]).

In many practical circumstances, the shape geometry is reparameterized with finitely many parameters. A shape reparameterization is typically needed if there are producibility restrictions (Liedmann et al. [53]) or if the initial geometric variable is not differentiable (Hwang [40]). Well-parameterized shape geometry often allows the application of standard optimization approaches (Fröhlich et al. [27], Hicken et al. [36], Perez et al. [70]) or Newton-Krylov type methods (Dener et al. [23]). On the other hand, the achievable shape is limited and dependent on the chosen parameterization. High-fidelity shape optimizations, which are directly based on finite element meshes (Linkwitz et al. [54], Zienkiewicz et al. [90]) or level-set methods (Allaire et al. [4], Sethian [80]), exploit the largest shape space possible for real-life problems but lead to challenging optimization problems (Feppon et al. [26], Luft et al. [56]).

Motivation

In general, large-scale shape optimization is mainly performed using gradient descent type methods so far (Schulz et al. [79]). The gradient descent method, originally proposed by Cauchy, is a first order method for unconstrained optimization that uses the negative gradient of the objective function for the variable update. In a computational framework, it writes

$$x_{k+1} = x_k - \alpha \nabla f(x_k), \tag{1.4}$$

where $x \in \mathbb{R}^n$ is the variable vector, $f : \mathbb{R}^n \to \mathbb{R}$ is the objective function, $\alpha > 0$ is the step size parameter, and k is the iteration counter.

In the engineering practice of shape optimization, a variety of constraints may be considered. In addition to the unconstrained problem (1.3), we want to further consider

$$C_i(\Omega^\star, u(\Omega^\star)) \leq 0, \quad i = 1, \ldots, m, \tag{1.5}$$

where $C_i : (\Omega, y) \mapsto C_i(\Omega, y) \in \mathbb{R}, \Omega \in \mathcal{O}, y \in V(\Omega)$ is a *constraint functional.*

However, there is a lack of literature on the general treatment of large-scale constrained shape optimizations, which is the main motivation of this work.

1.4 Main results

We present a *gradient descent akin method* for inequality constrained optimizations: at each iteration, we compute a search direction using a linear combination of the negative and normalized objective and constraint gradient,

$$\mathbf{s}_\zeta = -\frac{\nabla f(x)}{|\nabla f(x)|} - \zeta \frac{\nabla g(x)}{|\nabla g(x)|}, \tag{1.6}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is the objective function, $g : \mathbb{R}^n \to \mathbb{R}$ is the constraint function, $\zeta \in [0, 1)$ is a parameter, and $|\cdot|$ denotes the Euclidean norm. A generalization to multiple constraints is proposed by replacing $g(x)$ in (1.6) with the logarithmic barrier function

$$\Phi(x) = -\sum_{i=1}^{m} \log(-g_i(x)), \tag{1.7}$$

where $g_1, ..., g_m : \mathbb{R}^n \to \mathbb{R}$ are constraint functions, and $\log(\cdot)$ denotes the natural logarithm.

The design of the method is inspired by the singular value decomposition. Using a dynamical systems perspective, we show the method is globally convergent to KKT solutions and locally convergent to local minimizers. The method is demonstrated using both common test cases and applications to large-scale shape optimizations.

Outline

Chapter 2 - 4 introduce some of the fundamentals of continuum mechanics, finite element method, and shape sensitivity analysis, which are the basis of shape optimization.

Chapter 5, 6 introduce some of the fundamentals of gradient-based optimization and dynamical systems and some of their connections. The former is the basis of the method design; the latter provides the basis for the theoretical studies.

Chapter 7, 8 present the design and derivation of the proposed method.

Chapter 9-11 present the theoretical results.

Finally, *Chapter 12* shows computational examples and *Chapter 13* gives a conclusion.

Notations: Since different subjects have been considered in this work, it is difficult to use consistent notations throughout the thesis. For example, while x is conventionally defined as the position vector in continuum mechanics, it is the optimizing variable vector in an optimization problem, and it is also the state vector in the phase space in a dynamical system. While f is commonly used to denote the force vector in finite element analysis, it is the objective function in optimization. With the goal of better readability of each chapter, the common nomenclature of each topic are retained. Symbols are carefully defined wherever I think it is appropriate, attempting to avoid ambiguity as much as possible. I apologize for the abusing of notations.

Note: Part of this thesis is based on two manuscripts of the author, Chen et al. [19] published in the Journal Structural and Multidisciplinary Optimization with the permission of the copyright holder, and the submitted work of Chen et al. [20].

CHAPTER 2

Basics of Continuum Mechanics

“I admire the elegance of your method of computation; It must be nice to ride through these fields upon the horse of true mathematics while the like of us have to make our way laboriously on foot.”

—Einstein to Levi-Civita

2.1 Introduction

Continuum mechanics is a major area where physics and mathematics meet and forms the foundation of civil and mechanical engineering.

Just as the classical Newton mechanics, continuum mechanics deals with the interaction between force and motion, however, it is mainly concerned with deformable bodies and considers the material on the macroscopic scale.

2.2 Tensor

A tensor is an algebraic object related to a vector space that describes a (multilinear) relationship between some scalars, vectors, and other tensors. A tensor does not depend on a chosen coordinate frame, which on the one hand confirms the independence of the physical law from chosen coordinate frames, and on the other hand provides a concise mathematical framework for the formulation and solution of physical problems. Tensors provide an important framework for mechanics, general relativity, and many others.

2.2.1 Covariant and contravariant transformation

Geometric or physical entities, such as position, velocity, or gradient, are described using a basis (coordinate system basis vectors) with components that correspond to the chosen basis. In tensor analysis, covariant and contravariant describe how these components change with a change of basis. In this subsection, we recapitulate covariant and contravariant transformation for vectors. The same idea, however, can be extended for higher-order tensors.

Let V be a vector space of dimension n and let $\mathbf{G}$ and $\hat{\mathbf{G}}$ be two bases of V. Let the change of basis from $\mathbf{G}$ to $\hat{\mathbf{G}}$ be defined by

$$\mathbf{G} \mapsto \hat{\mathbf{G}} = \mathbf{G}\mathbf{A}, \tag{2.1}$$

for some invertible $n \times n$ matrix $\mathbf{A}$ with entries A_{ij}, or $A^i{}_j$, $A_i{}^j$. Whether an index is displayed as a superscript or subscript depends on the transformation properties that are described below. Each base (column) vector $\mathbf{y}_j$ of the transformed basis $\hat{\mathbf{G}}$ is a linear combination of all the base (column) vectors $\mathbf{x}_i$ of the $\mathbf{G}$ basis, i.e.,

$$\mathbf{y}_j = \mathbf{x}_i A^i{}_j, \tag{2.2}$$

by the Einstein summation convention (see Appendix A.1).

A vector $\mathbf{v}$ is expressed in the basis $\mathbf{G}$ uniquely using the components (coordinates) as

$$\mathbf{v} = \mathbf{x}_i v^i. \tag{2.3}$$

In the transformed basis $\mathbf{G}$, the vector is expressed as

$$\hat{\mathbf{v}} = \mathbf{y}_j \hat{v}^j. \tag{2.4}$$

The vector **v** itself is invariant by choice of basis, so we get

$$\mathbf{x}_i v^i = \mathbf{y}_j \hat{v}^j = \mathbf{x}_i A^i{}_j \hat{v}^j. \tag{2.5}$$

Thus we get the transformation between the components under the two bases,

$$v^i = A^i{}_j \hat{v}^j, \tag{2.6}$$

or equivalently,

$$\hat{v}^j = (A^{-1})^j{}_i v^i. \tag{2.7}$$

Comparing (2.2) and (2.7), it appears that the components describing the vector **v** transform *contravariantly* with the change of the basis. The vector **v** is called contravariant vector. Contravariant components are denoted with superscripts as shown above. It is common to denote contravariant vectors as column vectors.

In contrast to the contravariant vector, the covariant vector (covector) has components that *co-vary* with a change of basis. Assume a change of basis as defined in (2.2), the components of a covector **w** transform as

$$\hat{w}_j = w_i A^i{}_j. \tag{2.8}$$

Covectors are often denoted as row vectors and have components that are denoted with subscripts. Gradient of the form $\frac{\partial s}{\partial x^i}$, where s is a scalar field and x^i are the coordinates of a given basis, is one of the most prominent covariant vectors.

2.2.2 Tensor of order (r, s)

In order to represent a geometric or physic entity that is independent of a chosen basis frame, a tensor uses the contra- and covariant transformation for its components. Contravariant and covariant vectors are 1^{st} order tensors and can be represented as a one-dimensional array. Tensors can be of any order. For example, the Cauchy stress tensor that describes a material stress state is a 2^{nd} order tensor and writes as a two-dimensional array. In general, an n-th order tensor can be written in an n-dimensional array.

Definition 1. A tensor of type (r, s) is an assignment of a $(r+s)$-dimensional array

$$T^{i_1 \dots i_r}_{j_1 \dots j_s}$$

to each of its covariant bases

$$\mathbf{G}_{i_p} = (\mathbf{e}_{1p}, \dots, \mathbf{e}_{np}),\ p = 1, \dots, r,$$

and its contravariant bases

$$\mathbf{G}^{j_q} = (\mathbf{e}^{1q}, \dots, \mathbf{e}^{nq}),\ q = 1, \dots, s$$

of n-dimensional vector space such that, if individual changes of the basis are applied

$$\begin{aligned} &\mathbf{G}_{i_p} \mapsto \tilde{\mathbf{G}}_{i_p} = \mathbf{G}_{i_p} \mathbf{A}_{i_p},\ p = 1, \dots, r, \\ &\mathbf{G}^{j_q} \mapsto \tilde{\mathbf{G}}^{j_q} = \mathbf{G}^{j_q} \mathbf{A}^{j_q},\ q = 1, \dots, s, \end{aligned}$$

then, by the Einstein summation convention, the multidimensional array representing the components of the tensor obeys the transformation law

$$T^{i'_1 \dots i'_r}_{j'_1 \dots j'_s} = (A^{-1}_{i_1})^{i'_1}_{i_1} \dots (A^{-1}_{i_r})^{i'_r}_{i_r} T^{i_1 \dots i_r}_{j_1 \dots j_s} (\mathbf{A}^{j_1})^{j_1}_{j'_1} \dots (\mathbf{A}^{j_s})^{j_s}_{j'_s}. \tag{2.9}$$

2.3 Kinematics

The study of motion and deformation of a continuum is called kinematics. The basics for kinematics of shell elements (Kiendl et al. [47], Bischoff et al. [15]) that were used in this work are reviewed in this section. In particular, the deformation, displacement, strain, and their relationships are described.

2.3.1 Two observation basis frames: Cartesian and curvilinear coordinate system

In order to compute the motion and deformation of a material point in continuum, two configurations are used: the undeformed (reference) configuration and the deformed (actual) configuration (see figure 2.1).

For shell structures in the 3-dimensional Euclidean vector space, we use two coordinate systems: a fixed Cartesian coordinate system with base vectors $\mathbf{e}_i = \mathbf{e}^i$; and a *convective* curvilinear coordinate system with base vectors $\mathbf{G}_i$ and $\mathbf{g}_i$, which are "fixed" on the material point in a continuum.

The position vector for a material point P_0 is described in the reference and actual configuration in the fixed Cartesian basis as contravariant vectors

$$\mathbf{X} = X^1\mathbf{e}_1 + X^2\mathbf{e}_2 + X^3\mathbf{e}_3, \qquad \mathbf{x} = x^1\mathbf{e}_1 + x^2\mathbf{e}_2 + x^3\mathbf{e}_3. \tag{2.10}$$

The information for motion of this point is solely contained in the contravariant components X^i and x^i. In the curvilinear basis, the position vectors for a material point write

$$\mathbf{X} = \Theta^1\mathbf{G}_1 + \Theta^2\mathbf{G}_2 + \Theta^3\mathbf{G}_3, \qquad \mathbf{x} = \theta^1\mathbf{g}_1 + \theta^2\mathbf{g}_2 + \theta^3\mathbf{g}_3, \tag{2.11}$$

The curvilinear coordinate frame is further assumed to be *convective*, which means that the coordinates of any point of the continuum keep the same values in the reference and actual state, i.e., $\Theta^i = \theta^i$. Therefore, the information for the motion of any point and the deformation of the continuum in general are contained solely in the covariant basis vectors $\mathbf{G}_i$ and $\mathbf{g}_i$. This curvilinear description is advantageous due to two reasons. First, we can use the powerful tensor calculus to formulate various geometrical and physical entities of interest, such as *curvature*, *Cauchy stress*, and *Green-Lagrangian strain*, using the curvilinear basis vectors. Second, these basis vectors and their variations can be efficiently computed with the Finite Element Method by means of the isoparametric concept, which we will discuss in the next Chapter.

The contravariant components of Cartesian bases can be represented by the curvilinear coordinates

$$X^i = X^i(\Theta^1, \Theta^2, \Theta^3), \qquad x^i = x^i(\Theta^1, \Theta^2, \Theta^3) \tag{2.12}$$

The curvilinear basis vectors related to P_0 can then be computed by the partial derivatives of the contravariant components in the Cartesian basis, i.e.,

$$\mathbf{G}_i = \frac{\partial \mathbf{X}}{\partial \Theta^i} = \frac{\partial X^k}{\partial \Theta^i}\mathbf{e}_k, \qquad \mathbf{g}_i = \frac{\partial \mathbf{x}}{\partial \Theta^i} = \frac{\partial x^k}{\partial \Theta^i}\mathbf{e}_k. \tag{2.13}$$

The related contravariant basis $\mathbf{G}^i$ and $\mathbf{g}^i$ are given by

$$\mathbf{G}^i = \frac{\partial \Theta^i}{\partial X^k}\mathbf{e}^k, \qquad \mathbf{g}^i = \frac{\partial \Theta^i}{\partial x^k}\mathbf{e}^k. \tag{2.14}$$

2.3.2 Deformation gradient

As is shown in figure 2.1, the displacement of a material point P is

$$\mathbf{u} = \mathbf{x} - \mathbf{X}. \tag{2.15}$$

A differential line element in both configurations can be computed using the curvilinear bases,

$$d\mathbf{X} = \mathbf{G}_i d\Theta^i, \qquad d\mathbf{x} = \mathbf{g}_i d\Theta^i. \tag{2.16}$$

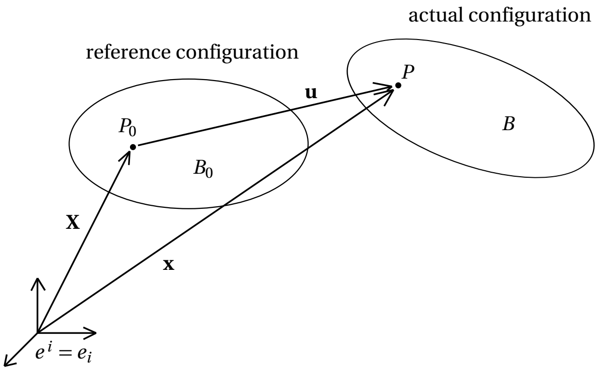

Figure 2.1: Reference and actual configuration

Due to the convective coordinates, the equations (2.16) correspond to the same differential line element in the reference and actual configuration. The main characteristic of the deformation of this differential line element is therefore encoded solely in the two curvilinear bases $\mathbf{G}_i$ and $\mathbf{g}_i$.

To describe the deformation of the differential line element, the deformation gradient $\mathbf{F}$ is defined through the tensor product of $\mathbf{g}_i$ and $\mathbf{G}^i$,

$$\mathbf{F} = \mathbf{g}_i \otimes \mathbf{G}^i, \qquad \mathbf{F}^T = \mathbf{G}^i \otimes \mathbf{g}_i. \tag{2.17}$$

Note that $\mathbf{F}$ is a 2^{nd}-order tensor and is itself a vector space. By the tensor calculus,

$$\mathbf{F}\,\mathbf{G}_i = (\mathbf{g}_j \otimes \mathbf{G}^j)\mathbf{G}_i = \delta_i^j \mathbf{g}_j = \mathbf{g}_i. \tag{2.18}$$

By (2.16) and (2.18), we have essentially that the deformation gradient maps the differential line element from the reference to the actual configuration,

$$\mathbf{F}\,d\mathbf{X} = d\mathbf{x}. \tag{2.19}$$

Therefore, the deformation gradient $\mathbf{F}$ can be calculated using the Jacobian matrix

$$\mathbf{F} = J_{\mathbf{x}} = \begin{bmatrix} \frac{\partial x^1}{\partial X^1} & \frac{\partial x^1}{\partial X^2} & \frac{\partial x^1}{\partial X^3} \\ \frac{\partial x^2}{\partial X^1} & \frac{\partial x^2}{\partial X^2} & \frac{\partial x^2}{\partial X^3} \\ \frac{\partial x^3}{\partial X^1} & \frac{\partial x^3}{\partial X^2} & \frac{\partial x^3}{\partial X^3} \end{bmatrix}. \tag{2.20}$$

2.3.3 A finite strain tensor: Green-Lagrangian strain

From (2.19) it follows from a polar decomposition that the deformation gradient gives information about stretches and the rigid body rotation —it can also be regarded as an affine transformation but without translation. The *strain*, on the other hand, is used to evaluate how much a deformation differs locally from any rigid body displacement (translation and rotation). To this end, various strain tensors were proposed. For large deformations, finite strain tensors must be used instead of infinitesimal strain tensors.

One of such tensors is the *Green-Lagrangian strain tensor*, which can be described using the deformation gradient $\mathbf{F}$ as

$$\mathbf{E} = \frac{1}{2}(\mathbf{F}^T \mathbf{F} - \mathbf{I}) = E_{ij}\, \mathbf{G}^i \otimes \mathbf{G}^j. \tag{2.21}$$

2.4 Stress

Stress is a physical quantity of 2^{nd} order tensor that expresses the internal forces inside a body. Various stresses were proposed, and in this section, we review the *Cauchy stress tensor* and the *Piola-Kirchhoff tensor*.

At the deformed configuration, the *Cauchy stress vector* $\mathbf{t}$ acting on an infinitesimal surface area Δa is defined as

$$\mathbf{t} = \lim_{\Delta a \to 0} \frac{\Delta \mathbf{p}}{\Delta a}, \tag{2.22}$$

where $\Delta \mathbf{p}$ denotes the force acting on the surface Δa. Let $\mathbf{n}$ be the surface normal of Δa. The Cauchy stress tensor $\boldsymbol{\sigma}$, which is the real stress in the deformed configuration of a body, is defined as

$$\mathbf{t} = \boldsymbol{\sigma} \mathbf{n}. \tag{2.23}$$

The first Piola-Kirchhoff (PK1) stress tensor $\mathbf{P}$ can be obtained by

$$\mathbf{P} = \det \mathbf{F} \cdot \boldsymbol{\sigma} \cdot \mathbf{F}^{-1}. \tag{2.24}$$

The second Piola-Kirchhoff (PK2) stress tensor $\mathbf{S}$ can be obtained by

$$\mathbf{S} = \det \mathbf{F} \cdot \mathbf{F}^{-1} \cdot \boldsymbol{\sigma} \cdot \mathbf{F}^{-T}. \tag{2.25}$$

To formulate the internal energy of a continuum, the stress and strain must be energetically conjugated. The PK2 stress tensor and the Green-Lagrangian strain tensor (2.21) are energy conjugate, i.e.,

$$\mathbf{S} = \frac{\partial W^{\text{int}}}{\partial \mathbf{E}}, \tag{2.26}$$

where W^{int} is the strain energy.

2.5 Conservation laws

Being the most important general principles of continuum mechanics, the conservation laws are reviewed in this section.

2.5.1 Conservation of mass

The law of conservation of mass states that the mass of a body remains constant during a deformation process. Let ρ_0 be the mass density of the reference configuration, ρ be the mass density of the actual configuration, then

$$\rho_0 = \rho \det \mathbf{F}. \tag{2.27}$$

2.5.2 Conservation of linear momentum

The *Cauchy equation of motion* states that at each point $\mathbf{x}$ of the actual configuration,

$$\rho \ddot{\mathbf{x}} = \operatorname{div} \boldsymbol{\sigma} + \rho \mathbf{b}, \tag{2.28}$$

where $\mathbf{b}$ is the volume forces measured per unit mass. In the static case, where the acceleration $\ddot{\mathbf{x}}$ vanishes, we have

$$\operatorname{div} \boldsymbol{\sigma} + \rho \mathbf{b} = \mathbf{0}, \tag{2.29}$$

which is the equilibrium equation.
The equilibrium equation can be formulated in the reference configuration by the PK2 stress tensor,

$$\operatorname{div}(\mathbf{F}\boldsymbol{\sigma}) + \rho_0 \mathbf{b}_0 = 0, \tag{2.30a}$$

$$\mathbf{F}\,\mathbf{S}\mathbf{n_0} = \mathbf{t}_0, \tag{2.30b}$$

where $\mathbf{n}_0$ and $\mathbf{t}_0$ are the surface normal and surface traction in the reference configuration, respectively.

2.5.3 Conservation of angular momentum

The conservation law of angular momentum leads to the well-known symmetry of the Cauchy stress tensor,

$$\sigma_{ij} = \sigma_{ji}. \tag{2.31}$$

2.6 Constitutive laws

The constitutive laws for elastic materials describe the relations between energy conjugate stress and strain tensors.

A material is said to be *elastic*, if the Cauchy stress tensor $\boldsymbol{\sigma}$ can be solely determined by the deformation gradient $\mathbf{F}$,

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}(\mathbf{F}). \tag{2.32}$$

The material tensor is of fourth order and is denoted by $\mathbf{C}$,

$$\mathbf{C} = C^{ijkl}\,\mathbf{G}_i \otimes \mathbf{G}_j \otimes \mathbf{G}_k \otimes \mathbf{G}_l \tag{2.33}$$

If a potential exist, then

$$C^{ijkl} := \frac{\partial^2 W^{\text{int}}}{\partial E_{ij} \partial E_{kl}}. \tag{2.34}$$

The St. Venant-Kirchhoff material model, which assumes a linear relation between strains and stresses, indicates

$$\mathbf{S} = \mathbf{C} : \mathbf{E}, \tag{2.35}$$

or,

$$S^{ij} = C_{ijkl} E_{kl}. \tag{2.36}$$

2.7 Basics of calculus of variation

The variational principle is one of the foundations of the finite element method. Here, the basics of variational calculus are reviewed. Consider the functional

$$J = \int_{x_1}^{x_2} L(x, y(x), y'(x)) dx, \tag{2.37}$$

where x_1, x_2 are constants, y and L are twice continuously differentiable with respect to their arguments. Suppose the function $y^\star$ leads to the extrema (stationary) of the functional I. Further, assume there is a slightly "varied" function $\tilde{y}(x)$ to $y^\star$, and vanishes at x_1 and x_2. By introducing a number α close to 0, we obtain a set of functions $y(x,\alpha)$ (Müller [62]),

$$y(x,\alpha) = y^\star(x) + \alpha(\tilde{y}(x) - y^\star(x)). \tag{2.38}$$

Now, we want to study the function $y(x,\alpha)$ in relation to the variable α. It is conventionally to denote

$$\delta = \left.\frac{\partial}{\partial \alpha}\right|_{\alpha=0}. \tag{2.39}$$

Thus, the partial derivative $\left.\frac{\partial}{\partial \alpha}\right|_{\alpha=0} y(x,\alpha)$ reads

$$\delta y(x,\alpha) = \tilde{y}(x) - y^\star(x). \tag{2.40}$$

Denote $\delta y(x,\alpha)$ as δy, which is called the *variation* of the function $y(x)$. Since x and α are independent, we abbreviate $y(x,\alpha) = y(\alpha)$ and $y(x)^\star = y^\star$, then

$$y(\alpha) = y^\star + \alpha\delta y. \tag{2.41}$$

Similarly, let $\delta y'$ be the variation of the function y', then there is a set of functions $y'(\alpha)$,

$$y(\alpha)' = (y^\star)' + \alpha\delta y'. \tag{2.42}$$

Substitute (2.41) and (2.42) into the original functional (2.37), we obtain its *variational formulation* $J(\alpha)$,

$$J(\alpha) = \int_{x_1}^{x_2} L(x, y(\alpha), y'(\alpha))dx, \tag{2.43}$$

which is a function of α and we can now use the infinitesimal calculus to find its stationary solutions. According to the definition of δy in (2.41), the function J has extrema at $\alpha = 0$ since $y = y^\star$, thus

$$\left.\frac{dJ(\alpha)}{d\alpha}\right|_{\alpha=0} = \int_{x_1}^{x_2} \left.\frac{dL(x, y(\alpha), y'(\alpha))}{d\alpha}\right|_{\alpha=0} dx = 0. \tag{2.44}$$

We call $\left.\frac{dJ(\alpha)}{d\alpha}\right|_{\alpha=0}$ the first variation δJ of the functional J. Notice that $y(\alpha) = y^\star$ as $\alpha = 0$, we have

$$\begin{aligned}
\delta J &= \int_{x_1}^{x_2} \left.\frac{dL(x, y(\alpha), y'(\alpha))}{d\alpha}\right|_{\alpha=0} dx \\
&= \int_{x_1}^{x_2} \left(\frac{\partial L}{\partial x}\frac{\partial x}{\partial \alpha} + \frac{\partial L}{\partial y^\star}\frac{\partial y(\alpha)}{\partial \alpha} + \frac{\partial L}{\partial (y^\star)'}\frac{\partial y'(\alpha)}{\partial \alpha} \right) dx \\
&= \int_{x_1}^{x_2} \left(\frac{\partial L}{\partial y^\star}\delta y + \frac{\partial L}{\partial (y^\star)'}\delta y' \right) dx \\
&= \int_{x_1}^{x_2} \left(\frac{\partial L}{\partial y^\star}\delta y \right) dx + \left[\frac{\partial L}{\partial (y^\star)'}\delta y \right]_{x_1}^{x_2} - \int_{x_1}^{x_2} \frac{d}{dx}\left(\frac{\partial L}{\partial (y^\star)'} \right) \delta y \, dx \\
&= \int_{x_1}^{x_2} \left(\frac{\partial L}{\partial y^\star} - \frac{d}{dx}\left(\frac{\partial L}{\partial (y^\star)'} \right) \right) \delta y \, dx.
\end{aligned} \tag{2.45}$$

By the fundamental lemma of calculus of variations, the integrand in the bracket must be zero. We obtain

$$\frac{\partial L}{\partial y^\star} - \frac{d}{dx}\left(\frac{\partial L}{\partial (y^\star)'} \right) = 0, \tag{2.46}$$

which is the *Euler-Lagrange equation.* To use the *Euler-Lagrange equation* to find the extremal function $y^\star$ of the functional J, we let

$$L = L(x, y^\star(x), (y^\star)'(x)), \tag{2.47}$$

and substitute (2.47) into (2.46). Solving the *Euler-Lagrange partial differential equation* gives us the result.

CHAPTER 3

BASICS OF FINITE ELEMENT METHOD

3.1 Introduction

In this work, we consider shape optimization in the context of computational mechanics, and the finite element method is used for the modeling and simulation of the considered engineering problem. This chapter presents some of the fundamentals of the finite element method.

3.2 Variational principles

Principle of virtual work

The principle of virtual work is the variational basis for the displacement-based finite element formulation. It states that if a mechanical system is at the equilibrium state, an arbitrary geometrically compatible variation of the displacement $\delta\mathbf{u}$ does not cause any work. If there exists a potential Π, the principle of virtual work can be derived from the *principle of minimum potential energy*.

Consider a boundary value problem in continuum mechanics with a body volume V and a boundary surface A. We consider the Dirichlet and the Neumann boundary condition in the reference frame consisting of two parts A_u and A_t, on which the displacements $\mathbf{u}$ and forces $\mathbf{t}_0$ are prescribed, respectively,

$$\mathbf{u} = \bar{\mathbf{u}} \text{ on } A_u, \tag{3.1a}$$

$$\mathbf{t}_0 = \bar{\mathbf{t}} \text{ on } A_t. \tag{3.1b}$$

The total potential energy reads

$$\Pi(\mathbf{u}) = \Pi_{int} + \Pi_{ext} = \int_V W_{int}^b \, dV - \int_V W_{ext}^t \, dV - \int_{A_t} W_{ext} \, dA, \tag{3.2}$$

which is a functional of the displacement $\mathbf{u}$. The minimum of the potential can be found by setting its first variation to zero. In the reference configuration and use the Green-Lagrangian strain tensor, it writes

$$\delta\Pi(\mathbf{u}) = \int_V \frac{\partial W_{int}}{\partial \mathbf{E}} : \delta\mathbf{E} \, dV - \int_V \rho_0 \mathbf{b}_0 \cdot \delta\mathbf{u} \, dV - \int_{A_t} \bar{\mathbf{t}} \cdot \delta\mathbf{u} \, dA = 0. \tag{3.3}$$

With the variation of the Green-Lagrangian strain $\delta\mathbf{E}$,

$$\delta\mathbf{E} = \delta\mathbf{F}^T \cdot \mathbf{F} = \mathbf{F}^T \cdot \delta\mathbf{F} = \mathbf{F}^T \cdot \text{grad}\, \delta\mathbf{u}, \tag{3.4}$$

we obtain the principle of virtual work

$$\delta\Pi(\mathbf{u}) = \int_V (\mathbf{F}\,\mathbf{S}) : \text{grad}\, \delta\mathbf{u} \, dV - \int_V \rho_0 \mathbf{b}_0 \cdot \delta\mathbf{u} \, dV - \int_{A_t} \bar{\mathbf{t}} \cdot \delta\mathbf{u} \, dA = 0. \tag{3.5}$$

Nonlinear formulation

If a variational principle exists, then means are immediately available to obtain approximate solutions in the standard integral form suitable for finite element analysis (Zienkiewicz et al. [90]).

In the following, the nonlinear finite element formulation that is based on the variational principle and uses the Newton-Raphson method is shown. At the equilibrium, the principle of virtual work states that,

$$\delta W = \delta W_{int} - \delta W_{ext} = 0 \tag{3.6}$$

Its variational formulation is

$$\int_V \boldsymbol{\sigma} : \delta\boldsymbol{\epsilon} dV - \int_V \rho \mathbf{b} \cdot \delta\mathbf{u} dV - \int_{A_t} \bar{\mathbf{t}} \cdot \delta\mathbf{u} dA = 0. \tag{3.7}$$

Assume a variation of the displacement at r-th dof, δu_r. Then, by the definition of variation δ and the chain rule,

$$\begin{aligned} \delta W =& \frac{\partial W}{\partial u_r}\delta u_r \\ =& \int_V \boldsymbol{\sigma} : \frac{\partial \boldsymbol{\epsilon}}{\partial u_r}\delta u_r dV - \int_V \rho b_r \cdot \delta u_r dV - \int_{A_t} \bar{t}_r \cdot \delta u_r dA \\ =& 0. \end{aligned} \tag{3.8}$$

Thus,

$$\frac{\partial W}{\partial u_r} = \int_V \boldsymbol{\sigma} : \frac{\partial \boldsymbol{\epsilon}}{\partial u_r} dV - \int_V \rho b_r \cdot dV - \int_{A_t} \bar{t}_r \cdot dA = 0. \tag{3.9}$$

To find the stationary solution, we use the Newton-Raphson method,

$$\frac{\partial W}{\partial u_r} + \frac{\partial^2 W}{\partial u_r \partial u_s}\Delta u_s = 0. \tag{3.10}$$

Assume that the external load is displacement independent, we have

$$\frac{\partial^2 W}{\partial u_r \partial u_s} = \int_V \left(\frac{\partial \boldsymbol{\sigma}}{\partial u_s} : \frac{\partial \boldsymbol{\epsilon}}{\partial u_r} + \boldsymbol{\sigma} : \frac{\partial^2 \boldsymbol{\epsilon}}{\partial u_r \partial u_s} \right) dV. \tag{3.11}$$

The Newton method can then be applied to compute the displacement unknowns Δu_s iteratively.

3.3 Weighted residual (Galerkin) method

Another fundamental approach for the formulation of the finite element method is the Galerkin method of weighted residual. Starting from the equation of motions in the reference frame (2.29), we compute its inner product with a displacement variation $\delta\mathbf{u}$

$$\int_V (\mathrm{div}(\mathbf{F}\,\mathbf{S}) + \rho_0 \mathbf{b}_0) \cdot \delta\mathbf{u}\, dV = 0. \tag{3.12}$$

While the term $(\mathrm{div}(\mathbf{F}\,\mathbf{S}) + \rho_0\mathbf{b}_0)$ represents the ''residual'', the term $\delta\mathbf{u}$ can be considered a ''weighting'' function. If the weighting function is arbitrary, then the differential equation in the brackets must be zero at all points in V. If the weighting function is the displacement variation $\delta\mathbf{u}$, then it is the Galerkin method.

In the following, we show the connection between the Galerkin method and the previously discussed variational formulation. Using the product rule for divergence (see Appendix A.7), we get

$$\mathrm{div}(\delta\mathbf{u}\mathbf{F}\,\mathbf{S}) = \delta\mathbf{u} \cdot \mathrm{div}(\mathbf{F}\,\mathbf{S}) + (\mathbf{F}\,\mathbf{S}) : \mathrm{grad}\;\delta\mathbf{u}. \tag{3.13}$$

Integrate (3.13) and by (3.4), we get

$$\int_V \delta\mathbf{u} \cdot \mathrm{div}(\mathbf{F}\,\mathbf{S}) dV = \int_V \mathrm{div}(\delta\mathbf{u}\mathbf{F}\,\mathbf{S}) dV - \int_V (\mathbf{F}\,\mathbf{S}) : \mathrm{grad}\;\delta\mathbf{u}\, dV. \tag{3.14}$$

By the Gauss-Green theorem, the equilibrium of motions in reference frame (2.30b), and the boundary conditions (3.1b), we have

$$\int_V \mathrm{div}(\delta\mathbf{u}\mathbf{F}\,\mathbf{S}) dV = \int_{A_t} (\delta\mathbf{u}\mathbf{F}\,\mathbf{S}) \cdot \mathbf{n}\, dA = \int_{A_t} \delta\mathbf{u} \cdot \bar{\mathbf{t}}\, dA. \tag{3.15}$$

And thus

$$\int_V \delta\mathbf{u} \cdot \mathrm{div}(\mathbf{F}\,\mathbf{S}) dV = -\int_V (\mathbf{F}\,\mathbf{S}) : \mathrm{grad}\;\delta\mathbf{u}\, dV + \int_{A_t} \delta\mathbf{u} \cdot \bar{\mathbf{t}}\, dA. \tag{3.16}$$

Inserting (3.16) into (3.12), we obtain the equation for the principle of virtual work (3.5). The major difference between the method of weighted residual and the principle of virtual work is that the former does not require the existence of a potential. Thus, it can be applied to a wider range of problems, such as inelasticity.

From the strong form to the weak form

The physics problems in continuum mechanics are mathematically modeled using partial differential equations derived from conservation laws. Recall the boundary value problem described by the equilibrium of motions

$$\operatorname{div}(\mathbf{F}\boldsymbol{\sigma}) + \rho_0 \mathbf{b}_0 = 0 \;\text{ in } V, \tag{3.17}$$

that satisfies the boundary conditions (3.1a) and (3.1b).

This formulation is called the strong form. A difficulty of the strong form is that the high requirement of the smoothness of the analytic solutions of the PDEs might be too strict for the real physical solutions. This motivates the so-called *weak formulation* of the problem, which starts with the weighted residual formulation (3.12) and results in the variational formulation (3.5). The displacement variation $\delta\mathbf{u}$ is called *test function* in the context of weak form. While second-order derivatives are involved in the strong form (3.17), only first-order derivatives are involved in the weak formulation (3.5), which is achieved by the procedure (3.13) through (3.15).

Although the analytical solution of the weak form is still difficult, it provides the basis for the numerical analysis, which uses numerical approximations to compute the solutions. One of the most powerful methods for solving differential equations in numerical analysis is the finite element method.

3.4 Discretization

In previous sections, the mathematical models for solving the unknown displacement field $\mathbf{u}$ are presented. One remaining difficulty is to compute each integral term. The finite element method tackles this difficulty by discretization, i.e., it decomposes the domain of interest into a finite number of subdomains, which are the so-called elements. The integration is then performed by 1) element-wise integration and 2) a global assembly procedure of all elements. Analytical integration for discretized elements, however, can still be difficult. In most cases, numerical quadrature needs to be used. For some cases, modifications to the standard numerical quadrature rules can even be used to overcome the difficulties of FEM. A notable example is the use of reduced integration to tackle the shear-locking phenomena of Reissner-Mindlin type elements that model thin-walled structures (see, e.g., Hughes et al. [39]). In the book by Zienkiewicz et al. [90], the authors describe the finite element method as ‘‘a general discretization procedure of continuum mechanics problems posed by mathematically defined statements’’, attaching great importance to discretization for FEM.

In the following, an introductory presentation of the very basics of finite element discretization is given. Note that the richness of the theories and techniques developed by mathematicians and engineers made it impossible to give an overview in this chapter.

3.4.1 Elements and nodes

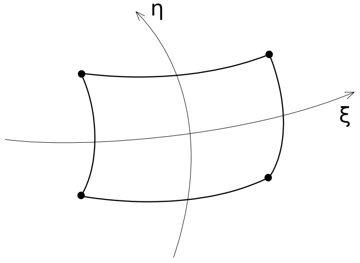

Figure 3.1: A quadrilateral finite element with local coordinates ξ and η.

In this work, we only consider the discretization in space. For thin-walled structures, triangle and quadrilateral elements are widely used. The finite element nodes possess the discretized values, that together with the *shape functions*, represent the continuous physical field of interest or the geometry of the studied object. In figure 3.1, we illustrate a four-noded quadrilateral element with the local coordinates ξ and η. The geometric and physical entities inside an element are interpolated by the nodal values and the corresponding shape functions, which we discuss in the next subsection.

3.4.2 Shape functions and the isoparametric concept

The basic ideas of shape functions are presented with an example of a four-noded quadrilateral element as illustrated in figure 3.1. For each node

i, an associated shape function N_i is defined,

$$\begin{aligned} N_1(\xi,\eta) &= \frac{1}{4}(1-\xi)(1-\eta), \\ N_2(\xi,\eta) &= \frac{1}{4}(1-\xi)(1+\eta), \\ N_3(\xi,\eta) &= \frac{1}{4}(1+\xi)(1+\eta), \\ N_4(\xi,\eta) &= \frac{1}{4}(1+\xi)(1-\eta), \end{aligned} \tag{3.18}$$

where $-1 \leq \xi \leq 1$ and $-1 \leq \eta \leq 1$. Then, after the meshing, the global coordinates describing the geometry for any local position (ξ,η) in an element is given by

$$\mathbf{x}^e(\xi,\eta) = \sum_i^4 N_i(\xi,\eta)x_i, \tag{3.19}$$

where x_i are the nodal coordinates of the element. If, in addition, the displacement of the element is interpolated using the same shape function, i.e.,

$$\mathbf{u}^e(\xi,\eta) = \sum_i^4 N_i(\xi,\eta)u_i, \tag{3.20}$$

where u_i are the nodal displacement of the element. Then, the element is called isoparametric. The isoparametric concept allows a unified way to treat elements and grants more flexible shapes. The cost is more expensive computations of derivatives and numerical integrations.

For example, the derivative computations with respect to the local coordinates involve an additional coordinate transformation step. To see this, we write down the following expression based on the chain rule,

$$\begin{bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix}. \tag{3.21}$$

The matrix $\begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix}$ is often denoted as the Jacobian matrix J in FEM literature. We note that by the strict definition, it is the transposed Jacobian matrix. Nevertheless, it does not affect the actual computation since the partial derivatives (e.g., $\frac{\partial x}{\partial \xi}$) are directly obtainable by (3.19), and their determinants have the same value.

The determinant of this "Jacobian", $\det J$, is important so that it is often computed and stored in the implementation of an element. For example, it is needed for the transformed numerical integration,

$$\iint dxdy = \iint |\det J| d\xi d\eta. \tag{3.22}$$

Therefore, $|\det J|$ can be interpreted as a "scaling" factor, which is the ratio of the area of the element expressed in global coordinates to its area in the local coordinates. When an isoparametric concept is implemented, which means that the area expressed in the local coordinates is the same for all elements, we can use $|\det J|$ to evaluate the mesh quality. In the absence of a local refinement, a smaller ratio of $\frac{\det J_{max}}{\det J_{min}}$ usually shows a superior quality.

3.4.3 Element-wise integration and global assembly of the system of equations

With the tools introduced above, the integral in the weak formulation can now be computed in a two-step procedure. First, numerical integration is performed element-wise. For static problems, the outcome of the integration is a linear equation system of unknown field values discretized at each element node. Then, a global assembly is performed and a global linear equation system is obtained. Here, the isoparametric concept shows its superiority in flexibility, as it allows elements of different shapes to be treated uniformly when combined with a numerical integration. The integrands, which usually involve some differential operators in the global coordinates, are first derived in terms of local coordinates by the chain rules. Whereby the involved Jacobian matrix J or its inverse is readily available by (3.19). By (3.20), it is clear that the differential operations are performed on the shape functions, which allows a standard implementation for each element. The difference between different elements is the nodal values that are usually substituted in the computation of the element stiffness matrix by some linear algebra operations. The element-wise integration is often done by numerical integrations. Numerical schemes such as Gaussian quadrature are very efficient for integrating polynomials.

For linear static problems, the element integration typically has the form

$$\mathbf{k}^e \mathbf{u}^e = \mathbf{f}^e, \tag{3.23}$$

where $\mathbf{k}^e$ is the element stiffness matrix, and $\mathbf{u}^e$ is the unknown displacement vector, $\mathbf{f}^e$ is the element nodal force vector. To illustrate the global assembly procedure, we introduce a global vector $\mathbf{q}$ of nodal values, which is the global contribution of the element nodal values expressed in the local vector $\mathbf{q}^e$. The system global vector $\mathbf{q}$ and the element local vector $\mathbf{q}^e$ are related by the mapping

$$\mathbf{q}^e = \mathbf{P}^e \mathbf{q}, \tag{3.24}$$

where $\mathbf{P}^e$ is an element-dependent matrix whose entries are only zeros and ones. The global stiffness matrix $\mathbf{K}$ and the global force vector $\mathbf{f}$ can then be assembled by adding the contributions of all the elements at the respective global degrees of freedom[1],

$$\mathbf{K} = \sum_e (\mathbf{P}^e)^T \mathbf{k}^e \mathbf{P}^e, \quad \mathbf{f} = \sum_e (\mathbf{P}^e)^T \mathbf{f}^e. \tag{3.25}$$

Finally, the global system of equations is established,

$$\mathbf{K}\mathbf{u} = \mathbf{f}. \tag{3.26}$$

The unknown continuous field that is discretized as the vector $\mathbf{u}$ can then be solved with solvers for system of equations.

[1] The mapping matrix $\mathbf{P}^e$ is introduced only for the purpose of presentation but should not be used in practical implementations (Haslinger et al. [33]).

CHAPTER 4

FINITE ELEMENT BASED SHAPE SENSITIVITY ANALYSIS

4.1 Introduction

Formally, structural shape optimization belongs to PDE-constrained optimizations, where the optimization problem is constrained by at least one partial differential equation (PDE). Large-scale shape optimization faces challenges in PDE-constrained optimization in general and its own difficulties in particular. The former difficulties include the large variable space and high computational cost for each response function. The latter include the computation and regularization of the shape gradient.

The computation of shape gradients is called the shape sensitivity analysis. This chapter introduces the basics for shape sensitivity analysis based on the finite element method.

4.2 PDE-constrained optimization: a perspective from the implicit function theorem

Let x, u define the control and the state variables, respectively. Then, a PDE-constrained optimization may be defined as

$$\begin{aligned} &\min_{x,u} && f(x,u), && f : X \times U \to \mathbb{R}, \\ &\text{subject to} && h(x,u) = 0, && h : X \times U \to Z, \end{aligned} \tag{4.1}$$

where X, U, Z are vector spaces and f, h are sufficiently smooth functions. If the Jacobian matrix about u at a point $c(a,b) = 0$, and $(a,b) \in X \times U$, i.e.,

$$J_{h,u}(a,b) = \left[\frac{\partial h_i}{\partial u_j}(a,b) \right] \tag{4.2}$$

is invertible, then there exists a *unique* continuously differentiable vector-valued function ϕ such that in a neighborhood of (a,b) that satisfies

$$\phi(a) = b, \tag{4.3}$$

and

$$h(x, \phi(x)) = 0. \tag{4.4}$$

Suppose that ϕ exists everywhere in the domain of the function h, then we can reformulate the PDE-constrained optimization (4.1) as an unconstrained optimization of solely the control variables x,

$$\min_{x} \quad f(x, \phi(x)), \;\; f : X \times U \to \mathbb{R}. \tag{4.5}$$

By (4.5), in this work, we refer to a PDE-constrained optimization in the form (4.1) as an unconstrained problem. We call it a constrained optimization problem when additional constraints are considered. It should be noted that the unconstrained formulation (4.5) is fundamentally based on the existence of a unique mapping from x to u by the implicit function theorem. The shape optimization problems considered in this work belong to this class of problems.

4.3 Continuous and discrete shape gradient

Two main lines of approaches exist for the computation of shape gradient, distinguishing themselves in the order of the gradient derivation and system discretization. The continuous approach applies the profound Hadamard theorem to compute the analytic expression for surface shape gradient for given functionals directly. The discrete approach first discretizes the system (e.g., finite element discretization), and then computes the shape gradient for the discretized variables by the established linear equation systems.

Both approaches have their own advantages and can therefore be chosen depending on individual application. We neglect a general discussion of the comparison between both approaches and refer interested readers to Nadarajah et al. [64], Reuther et al. [73], and Schmidt [77].

We draw attention to the challenges in optimizations that both approaches face due to the complexity of shape Hessians. In the continuous approach, although the shape Hessian analysis is much more accessible compared to the discrete approach (Schmidt [77]), the shape Hessian still lacks symmetry and is a complex object even for moderate problems. For discrete approaches, the foremost problems are the discretization-dependency and the lack of regularity of the discrete shape gradient. There are different approaches to tackle both problems in the literature. However, a unified view of the different approaches is missing and may require further investigations. When going from discrete shape gradients to discrete shape Hessians, the problem of mesh-dependency and the lack of regularity may be even more severe. Even if one computes a discrete shape Hessian, one may immediately ask the question of whether the obtained discrete shape Hessian is consistent (in the case of fine discretizations) with the continuous shape Hessians after all the ‘‘remedies’’ for the mentioned two problems were implemented.

In this work, the discrete approach is used, which we discuss in the following. Note that the developed optimization method is applicable to both approaches, continuous and discrete.

4.4 Discrete adjoint sensitivity analysis

For problem (4.1), one of the most intuitive methods to compute the gradient may be the finite difference method. Suppose a finite difference for the i-th entry of the control variables x is Δx_i. Then the partial derivative $\frac{\partial f(x,u)}{\partial x_i}$ may be approximated as

$$\frac{\partial f(x,u)}{\partial x_i} \approx \frac{f(\hat{x}_i, \hat{u}_i) - f(x,u)}{\Delta x_i}, \tag{4.6}$$

where $\hat{x}_i = x + \Delta x_i$ is the perturbed control variable vector, and $\hat{x}_i, \hat{u}_i$ satisfies the governing PDE,

$$h(\hat{x}_i, \hat{u}_i) = 0. \tag{4.7}$$

The finite difference method becomes impractical quickly as the dimension of control variables grows, because the state solution (4.7) must be carried out for each disturbance. To overcome this difficulty, the adjoint method was introduced, which dates back to the work of Bryson [18] in the 1960s in optimal control.

In the following, one way to derive the discrete adjoint method is shown. The basis of the discrete ajoint method is the discretized PDE. The derivative of the objective function $f(x,u)$ with respect to the i-th control x_i in the discretized system writes

$$\frac{df(x,u)}{dx_i} = \frac{\partial f}{\partial x_i} + \left[\frac{\partial f}{\partial u}\right]^T \frac{\partial u}{\partial x_i}. \tag{4.8}$$

Notice that $\frac{\partial f}{\partial u}$ and $\frac{\partial u}{\partial x_i}$ are both denoted as column vectors, therefore we transpose $\frac{\partial f}{\partial u}$ to complete the chain rule.

By the implicit function theorem, the state derivatives $\frac{\partial u}{\partial x_i}$ can be calculated. At a neighborhood where $h(x,u) = 0$, we have

$$\frac{dh(x,u(x))}{dx_i} = \frac{\partial h}{\partial x_i} + \frac{\partial h}{\partial u}\frac{\partial u}{\partial x_i} = 0. \tag{4.9}$$

Notice that $\frac{\partial h}{\partial u}$ is a Jacobian matrix. From (4.9), it follows

$$\frac{\partial u}{\partial x_i} = \left[\frac{\partial h}{\partial u}\right]^{-1} \left[-\frac{\partial h}{\partial x_i}\right]. \tag{4.10}$$

Substituting (4.10) in (4.8), we obtain

$$\frac{df(x,u)}{dx_i} = \frac{\partial f}{\partial x_i} + \left[\frac{\partial f}{\partial u}\right]^T \left[\frac{\partial h}{\partial u}\right]^{-1} \left[-\frac{\partial h}{\partial x_i}\right]. \tag{4.11}$$

(4.11) is called the direct approach for the sensitivity computation. It requires the solution of the linear equation system (4.10) for each state derivative and is therefore still computationally expensive when the dimension of control variables is large. To overcome this difficulty, the adjoint approach makes use of the vector-matrix-vector-product,

$$\mathbf{w}^T \mathbf{A} \mathbf{v} = \mathbf{v}^T \mathbf{A}^T \mathbf{w}. \tag{4.12}$$

To see this, we can write down the first product of the above equation in the summation form

$$\mathbf{w}^T \mathbf{A} \mathbf{v} = \sum_{i,j} w_i A_{ij} v_j. \tag{4.13}$$

Switching the index i and j, we get

$$\mathbf{w}^T \mathbf{A} \mathbf{v} = \sum_{j,i} w_j A_{ji} v_i = \sum_{j,i} v_i A_{ji} w_j = \mathbf{v}^T \mathbf{A}^T \mathbf{w}. \tag{4.14}$$

The derivative of the objective function f with respect to the control variable can then be calculated as

$$\frac{df(x,u)}{dx_i} = \frac{\partial f}{\partial x_i} + \left[-\frac{\partial h}{\partial x_i}\right]^T \left[\frac{\partial h}{\partial u}\right]^{-T} \left[\frac{\partial f}{\partial u}\right]. \tag{4.15}$$

If $\frac{\partial h}{\partial u}$ is an adjoint matrix, i.e., $\frac{\partial h}{\partial u} = \left[\frac{\partial h}{\partial u}\right]^T$, then the above equation writes

$$\frac{df(x,u)}{dx_i} = \frac{\partial f}{\partial x_i} + \left[-\frac{\partial h}{\partial x_i}\right]^T \left[\frac{\partial h}{\partial u}\right]^{-1} \left[\frac{\partial f}{\partial u}\right]. \tag{4.16}$$

As a result, only one linear system has to be solved,

$$\Lambda = \left[\frac{\partial h}{\partial u}\right]^{-1} \left[\frac{\partial f}{\partial u}\right]. \tag{4.17}$$

The vector Λ is called the *adjoint variables.* Note that the linear system (4.17) has a similar form as the solution to the forward problem

$$h(u) = 0, \tag{4.18}$$

where the control variables x are fixed. Using the Newton method,

$$u_{n+1} = u_n - \left[\frac{\partial h(u_n)}{\partial u}\right]^{-1} h(u_n), \tag{4.19}$$

we obtain the state variables iteratively.

4.5 Finite element based shape gradient

In this section, the discrete adjoint analysis with finite element discretization is presented. Emphasize is on the special treatment to calculate the term $\frac{\partial h}{\partial x_i}$, which takes advantage of the finite element approach.

4.5.1 Computation of discrete adjoint variables

The discrete adjoint approach is demonstrated using a linear structural mechanics problem using FEM. Recall the global system of equation (3.26),

$$h(\mathbf{u}) = \mathbf{K}\mathbf{u} - \mathbf{f} = \mathbf{0}. \tag{4.20}$$

Notice that the bold **f** is the load vector that is to be distinguished with the objective function f.

Recall the adjoint state equation

$$\left[\frac{\partial h}{\partial \mathbf{u}}\right]\Lambda = \frac{\partial f}{\partial \mathbf{u}}. \tag{4.21}$$

From (4.20), straightforwardly, we get

$$\frac{\partial h}{\partial \mathbf{u}} = \mathbf{K}. \tag{4.22}$$

Next, the right-hand side $\frac{\partial f}{\partial \mathbf{u}}$ of the adjoint equation (4.21) is computed after the discretization.

Take the example of the linear strain-energy objective function $\mathcal{J}$, which can be computed based on the discretized nodal displacement and external forces,

$$\mathcal{J} = \frac{1}{2}\mathbf{u}^T\mathbf{f}. \tag{4.23}$$

Then, the partial derivative of $\mathcal{J}$ with respect to the discrete displacement $\mathbf{u}$ reads

$$\frac{\partial \mathcal{J}}{\partial \mathbf{u}} = \frac{1}{2}\mathbf{f}. \tag{4.24}$$

The discrete adjoint variables Λ can then be computed by solving the adjoint state equation (4.21). The equation has only to be solved once and the resulting adjoint variables are stored for further computations.

4.5.2 Element-wise computation

For problem (4.20), the partial derivative $\frac{\partial h}{\partial x_i}$ writes

$$\frac{\partial h}{\partial x_i} = \frac{\partial \mathbf{K}}{\partial x_i}\mathbf{u} - \frac{\partial \mathbf{f}}{\partial x_i}. \tag{4.25}$$

Both the tangent terms $\frac{\partial \mathbf{K}}{\partial x_i}$, $\frac{\partial \mathbf{f}}{\partial x_i}$ can be decomposed in the same routine as the finite element method, where the element stiffness and nodal forces are assembled into the global linear equation system (Logg et al. [55, chapter 6]). For certain types of control variable x_i, we can compute the term $\frac{\partial \mathbf{K}^e}{\partial x_i}$ and $\frac{\partial \mathbf{f}^e}{\partial x_i}$ on the element level. And then substitute them in

$$\frac{\partial h^e}{\partial x_i} = \frac{\partial \mathbf{K}^e}{\partial x_i}\mathbf{u}^e - \frac{\partial \mathbf{f}^e}{\partial x_i}. \tag{4.26}$$

Then, the element contribution of the sensitivity is computed as

$$\frac{d f^e}{d x_i} = \frac{\partial f^e}{\partial x_i} - \left[\frac{\partial h^e}{\partial x_i}\right]^T \Lambda^e, \tag{4.27}$$

where Λ^e retrieves the already stored adjoint variables on the respective element. The reason for carrying out (4.26) and (4.27) is by the observation of the local nature of certain control variables. For example, a nodal geometric variable is only affecting the adjacent element stiffness.

Finally, the global sensitivity $\frac{\partial f}{\partial x_i}$ is assembled from all contributing elements,

$$\frac{df}{dx_i} = \sum_e \frac{df^e}{dx_i}. \tag{4.28}$$

4.6 Sensitivity weighting

One difficulty of discrete shape gradient is that it is mesh-dependent. It is inconsistent with continuous shape gradient when the mesh quality is poor. Several works have studied this problem, and the sensitivity weighting (SW) method is implemented in this work. SW was first introduced in Kiendl et al. [45] for shell structures and is further studied in Wang [88]. This section first briefly reviews some of the theoretical studies and then shows the method following the above-mentioned two works.

We start with the continuous shape gradient. Suppose that the objective functional Ψ to be minimized depends on the state solution field u, its gradient ∇u, and the control field x, all defined in some domain $\Omega \in \mathbb{R}^3$,

$$\Psi = \int_\Omega H(u, \nabla u, x) d\Omega, \tag{4.29}$$

where H is the objective density function. In continuous formulation, the variation of Ψ reads

$$\delta\Psi = \int_\Omega g \cdot \delta x d\Omega = \langle g, \delta x \rangle_\Omega, \tag{4.30}$$

where δx is the variation of x, g denotes the continuous shape gradient, and $\langle \cdot, \cdot \rangle_\Omega$ denotes an inner product over the integration domain Ω. The above equation suggests that we can use the gradient descent method to find a minimizer of Ψ, i.e., to use the negative gradient as the search direction

$$d_c = -g. \tag{4.31}$$

The discrete shape gradient can be obtained via standard discrete sensitivity analysis. Equivalently, it can be obtained from the above continuous approach by an additional step that discretizes the control variable x

(Wang [88]). Assume the discretization (3.19) is used, then the continuous control field is discretized as

$$x = \sum_i N_i x_i, \tag{4.32}$$

where N_i is the i-th shape function corresponding to the discretized control variable x_i. Insert (4.32) into (4.30), we obtain

$$\delta\Psi = \int_\Omega g \sum_i N_i \delta x_i \, d\Omega = \sum_i \tilde{g}_i \delta x_i, \tag{4.33}$$

where δx_i is the variation of x_i, and $\tilde{g}_i$ is the i-th entry of the discrete shape gradient,

$$\tilde{g}_i = \int_\Omega g N_i \, d\Omega = \langle g, N_i \rangle_\Omega. \tag{4.34}$$

Comparing (4.33)-(4.34) to (4.30), it is then clear that each entry of the discrete shape gradient has a dependency on the integral of its supporting shape function. In the sensitivity weighting method, the factor for this discretization influence is determined by

$$a_i = \int_\Omega N_i \, d\Omega. \tag{4.35}$$

The weighted gradient is then computed entry-wise,

$$\tilde{g}_i^w = \tilde{g}_i / a_i. \tag{4.36}$$

4.7 Shape regularization and the vertex morphing method

Large-scale shape optimization may be considered an inverse problem, which can be generally stated as the following:
Find system parameter p, such that

$$q = F(p), \tag{4.37}$$

where F is the forward map and q is a certain measure of the system. Many inverse problems can be cast into the solution of an optimization problem.

For shape optimization, we can interpret it as to find the shape of a structure such that certain optimality (see next chapter 5) of an optimization problem is satisfied.

Unlike the forward problem, inverse problems are typically ill-posed in the sense of Hadamard, i.e., there is a violation in one or more of the following properties:

1. a solution exist;
2. the solution is unique;
3. and the solution's behavior changes continuously with the change of initial conditions.

For ill-posed problems, additional assumptions on the solution may be made. Such a process is called the regularization.

The shape optimization problems considered in this work are almost always ill-posed due to the very large discrete design space based on the finite element mesh. To tackle the problem, the Vertex Morphing method (VM) introduced in Hojjat et al. [37] is used.

The idea of the Vertex Morphing is to control the discrete surface coordinates $\mathbf{x} = [x_1, x_2, ..., x_n]^T$ with design controls $\mathbf{s} = [s_1, s_2, ..., s_n]^T$, filtered by a filter function. The explicit filtering used in VM is the convolution of the coordinate field x with a kernel. In the discretized system, it is a matrix-vector multiplication that is the summation of nodal contributions that are weighted with the kernel function. VM distinguishes itself with a standard explicit filtering, in which it applies the filtering process twice.

First, the so-called forward mapping step that uses the linear filtering matrix $\mathbf{R}$ is defined as follows:

$$x_i = R_{ij} s_j. \tag{4.38}$$

Similarly, the change of the control $\delta\mathbf{s}$ is mapped onto the change of the design configuration $\delta\mathbf{x}$

$$\delta x_i = R_{ij} \delta s_j. \tag{4.39}$$

The design controls are the control variables of the gradient-based optimization. The number of variables is equivalent to the number of surface coordinates. Following the chain rule of differentiation, the sensitivities of a response function Ψ with respect to the discretized geometry $\mathbf{x}$ are backward mapped to the design control using the adjoint or backward mapping matrix $\mathbf{R}^*$, with $\mathbf{R}^* = \mathbf{R}^T$ for regular grids,

$$\frac{d\Psi}{ds_i} = \frac{d\Psi}{dx_j}\frac{dx_j}{ds_i} = R_{ji}\frac{d\Psi}{dx_j}. \tag{4.40}$$

Equations (4.39) and (4.40) can be used in a gradient descent framework, ensuring smooth shape updates in each iteration.

CHAPTER 5

BASICS OF GRADIENT-BASED OPTIMIZATION

This chapter reviews some of the fundamentals of gradient-based optimization, including optimality conditions, convexity, and rate of convergence. Special attention is paid to the gradient descent method.

Gradient-based optimization finds its root in Taylor's Theorem (Nocedal et al. [67]), which is stated at first.

Theorem 1 (Taylor's Theorem). *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable. Then, we have*

$$f(x+s) = f(x) + \nabla f(x+\alpha s)^T s, \tag{5.1}$$

for some $\alpha \in (0,1)$. Further, if f is twice continuously differentiable, then

$$f(x+s) = f(x) + \nabla f(x)^T s + \frac{1}{2} s^T \nabla^2 f(x+\alpha s) s, \tag{5.2}$$

for some $\alpha \in (0,1)$.

5.1 Optimality conditions for unconstrained optimization

We first consider unconstrained optimization

$$\min_{x} \quad f(x), \tag{5.3}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a twice continuously differentiable function.
The first-order necessary condition for $x^\star$ to be optimal is

$$\nabla f(x^\star) = 0. \tag{5.4}$$

The second-order necessary condition for $x^\star$ to be optimal is

$$\nabla^2 f(x^\star) \succeq 0. \tag{5.5}$$

The second-order sufficient condition for $x^\star$ to be an optimal solution is

$$\nabla^2 f(x^\star) \succ 0. \tag{5.6}$$

5.2 Convexity

Convexity has a deep connection with optimization and has played an important role in the design and analysis of unconstrained optimization algorithms for decades and will probably continue to do so. In this section, we review some basics of convexity.

5.2.1 Convex set

A set $K \subset \mathbb{R}^n$ is said to be convex if the line segment between any two points in K also lies in K, i.e., for any $x_1, x_2 \in K$, we have the line segment $l_x \subset K$, where

$$l_x = \left\{ x_\alpha : x_\alpha = (1-\alpha)x_1 + \alpha x_2, \;\; \alpha \in [0,1] \right\}. \tag{5.7}$$

An *extreme point* $x^\star$ of a convex set K is a point that does not lie on the interior of any nonzero-length line segment in K. That is,

$$x^\star \notin \left\{ x_\alpha : x_\alpha = (1-\alpha)x_1 + \alpha x_2, \;\; x_1, x_2 \in K, \alpha \in (0,1) \right\}. \tag{5.8}$$

5.2.2 Convex function

A single-valued function $f:\mathbb{R}^n \to \mathbb{R}$ is said to be convex if its *epigraph* is a convex set. The epigraph of a function $f(x)$ is the set of points that lie above or on the graph of f. The graph of f is defined as the set

$$G(f)=\left\{(x,f(x)) : x\in\mathbb{R}^n\right\}, \tag{5.9}$$

which is a subset of $\mathbb{R}^n\times\mathbb{R}$. Similarly, the epigraph of f is also a subset of $\mathbb{R}^n\times\mathbb{R}$. By the definition of the convex set, we have the condition for the convex function

$$f(x_\alpha)\le f(x_0)+\alpha(f(x_1)-f(x_0)), \tag{5.10}$$

where x_α is a point in the set l_x defined in (5.7) with endpoints x_1, x_2, and $(x_1, f(x_1)), (x_2, f(x_2))$ are points on the graph of $f(x)$. In the following, we show important conditions for a convex function.

Take the first-order Taylor series of $f(x_\alpha)$ at point x_0, we get

$$f(x_\alpha)=f(x_0)+\nabla f(x_0)^T\alpha(x_1-x_0)+o(\alpha). \tag{5.11}$$

Insert (5.11) into (5.10), we get the first-order condition for a convex function

$$f(x_1)\ge f(x_0)+\nabla f(x_0)^T(x_1-x_0). \tag{5.12}$$

Taking the first-order Taylor series of $f(x_\alpha)$ at point x_1, we instead get the condition

$$f(x_1)\le f(x_0)+\nabla f(x_1)^T(x_1-x_0). \tag{5.13}$$

(5.12) and (5.13) yields

$$\nabla f(x_0)^T(x_1-x_0)\le\nabla f(x_1)^T(x_1-x_0). \tag{5.14}$$

By $x_1=x_0+(x_1-x_0)$, we take again the first-order Taylor series for $\nabla f(x_1)$ at point x_0, then,

$$\nabla f(x_0)=\nabla f(x_0)+\nabla^2 f(x_0)^T(x_1-x_0)+o(1). \tag{5.15}$$

Inserting (5.15) in (5.14), we get

$$\nabla f(x_0)^T(x_1-x_0)\le\left(\nabla f(x_0)+\nabla^2 f(x_0)^T(x_1-x_0)\right)^T(x_1-x_0). \tag{5.16}$$

Thus,

$$0 \leq (x_1 - x_0)^T \nabla^2 f(x_0)(x_1 - x_0). \tag{5.17}$$

The above inequality holds for arbitrary $x_0, x_1 \in \text{dom}(f)$. Hence we get the second-order condition for a convex function,

$$\nabla^2 f(x) \succeq 0. \tag{5.18}$$

That is, the Hessian matrix of the function is positive semi-definite everywhere. Furthermore, a function f is *strictly convex,* if

$$\nabla^2 f(x) \succ 0, \quad \forall x \in \text{dom}(f). \tag{5.19}$$

5.2.3 Strongly convex

Suppose that f is twice continuously differentiable, then it is *strongly convex* with parameter m if and only

$$\nabla^2 f(x) \succeq mI, \forall x \in \text{dom}(f), \tag{5.20}$$

where I is the identity matrix. The inequality (5.20) indicates that the minimum eigenvalue for $\nabla^2 f(x)$ is at least m for all x.

5.3 Gradient descent method

The gradient descent method, first introduced by Cauchy, is one of the earliest first-order iterative methods for unconstrained minimization. At each iteration, the gradient descent method uses a proportion of the negative gradient to update the variable x, i.e.,

$$x_{k+1} = x_k - \alpha \nabla f(x_k), \tag{5.21}$$

where k is the iteration number and α is the step size.

5.3.1 Steepest descent direction in Euclidean norm

The search direction of the gradient descent method is the steepest descent direction in the Euclidean norm. At each iteration, the first-order Taylor series of f at x_k is

$$f(x_k + d_k) \approx f(x_k) + \nabla f(x_k)^T d_k, \tag{5.22}$$

where $d_k \in \mathbb{R}^n$ can be seen as a small variable update vector. Our goal is to find a vector d_k, so that the objective function f decreases the most. However, the term $\nabla f(x_k)^T d_k$ in (5.22) itself is insufficient to give a meaningful result, since d_k can be chosen arbitrarily large, which results in the decrease of f can be unbounded. To provide a meaningful result, we fix the length of the search vector d_k. Any norm can be used to measure the length. If we use the Euclidean norm, the resulting search vector d_k is the direction[1] of the gradient descent method. Such a direction d_k is also called the steepest descent direction. However, we note that different norms result in different search directions, and they may also be called the steepest descent direction in that particular norm.

The steepest descent direction d_k in the Euclidean norm can be found by the optimization problem

$$\begin{aligned} &\min_{x} && \nabla f(x_k)^T d_k, \\ &\text{subject to} && |d_k| = 1, \end{aligned} \tag{5.23}$$

where $|\cdot|$ denotes the Euclidean norm. By Cauchy-Schwarz inequality, we have

$$|\nabla f(x_k)^T d_k| \leq |\nabla f(x_k)^T||d_k|, \tag{5.24}$$

and the equality is achieved if and only if $\nabla f(x_k)^T$ and d_k are parallel. Therefore,

$$d_k = -\frac{\nabla f(x_k)^T}{|\nabla f(x_k)^T|}, \tag{5.25}$$

which is the negative and normalized gradient of f. To see that the gradient descent method uses this steepest descent direction, we can rewrite (5.21) as

$$x_{k+1} = x_k - \left(\alpha |\nabla f(x_k)^T|\right) \frac{\nabla f(x_k)^T}{|\nabla f(x_k)^T|}. \tag{5.26}$$

Example with a quadratic norm

We show an example with a quadratic norm that is defined by the Hessian matrix of the objective function f. Figure 5.1 illustrates the steepest descent direction d_k in this particular norm. The search direction d_k can be

[1] In this work, a direction is referred to as a vector with unit length.

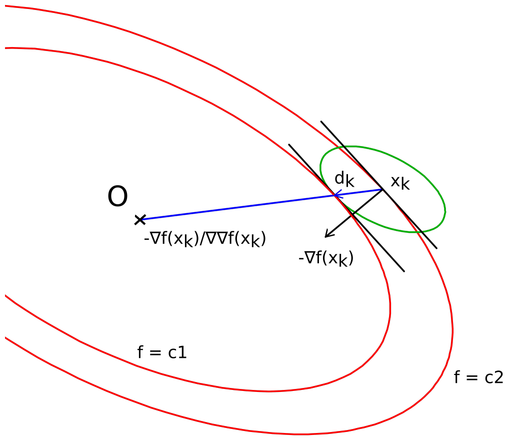

Figure 5.1: Search direction d_k with a quadratic norm defined by the Hessian matrix of a quadratic objective function.

found by the optimization problem

$$\begin{aligned} &\min_{x} && \nabla f(x_k)^T d_k, \\ &\text{subject to} && |d_k|_H = 1, \end{aligned} \tag{5.27}$$

where f is a quadratic function with two variables, and $|d|_H = \sqrt{d^T H d}$ is a quadratic norm that is defined by the Hessian matrix H of the objective function. The green ellipse illustrates the ‘‘unit circle’’ with the center point x_k measured by this quadratic norm. The red curves are the objective function contours. The optimization problem (5.27) can be solved graphically: find a vector d_k that starts with x_k and points to a point x on the green ellipse, such that the dot product $\nabla f(x_k)^T d_k$ is minimized. As a result, the resulting direction d_k is parallel to the blue line that connects x_k and the center of the red ellipse, which is the newton direction $-\frac{\nabla f(x)}{\nabla^2 f(x)}$.

5.3.2 Rate of convergence

The convergence rate for the gradient descent method is well-established. For convex functions we have,

Theorem 2. *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is convex and continuously differentiable, and that its gradient is Lipschitz continuous with constant $L > 0$, i.e.,*

$$|\nabla f(x) - \nabla f(y)| \leq L|x - y|, \tag{5.28}$$

for any x, y. Then, the gradient descent method (5.21) *with a fixed step size $0 < \alpha \leq 1/L$ achieves a solution x_k at k-th iteration that satisfies*

$$f(x_k) - f(x^\star) \leq \frac{|x_0 - x^\star|^2}{2\alpha k}, \tag{5.29}$$

where x_0 is the initialization and $x^\star$ is the optimal solution.

For strongly convex functions we have,

Theorem 3. *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is m-strongly convex and continuously differentiable, and that its gradient is Lipschitz continuous with constant $L > 0$. Then, the gradient descent method* (5.21) *with a fixed step size $0 < \alpha \leq 1/L$ achieves a solution x_k at k-th iteration that satisfies*

$$|x_k - x^\star|^2 \leq (1 - \alpha m)^k |x_0 - x^\star|^2. \tag{5.30}$$

If f is twice-continuously differentiable, then by the mean value theorem and the Lipschitz continuity condition for the gradient (5.28), we have

$$\nabla^2 f(x) \preceq LI, \;\; \forall x \in \mathrm{dom}(f). \tag{5.31}$$

That is, the maximum eigenvalue for $\nabla^2 f(x)$ is at most L for all x. Recall that m-strongly convex condition (5.20) indicates

$$\nabla^2 f(x) \succeq mI, \;\; \forall x \in \mathrm{dom}(f),$$

which is the lower bound of the eigenvalues for $\nabla^2 f$. We can observe from the term $(1 - \alpha m)^k$ that the convergence rate of the gradient descent is highly dependent on the conditioning of $\nabla^2 f(x)$. Notice that

$$0 < (1 - \alpha m) < 1.$$

Then, a larger condition number resulting in $(1 - \alpha m)$ closer to 1 and thus leads to slower convergence.

Comparing (5.29) and (5.30), we observe that the iteration number k characterizes both error measures for the solution but in different forms. To illustrate these two convergence rates, we first denote ϵ as the general error measure for the solution. Then, (5.29) implies that for convex functions, the gradient descent method finds an ϵ-solution in $k \sim o(1/\epsilon)$ iterations, where $o(1/\epsilon)$ is the ergodic sublinear rate for first-order optimization methods. Sometimes, $o(1/\epsilon)$ is called *exponential time*, which is illustrated in figure 5.2. For strongly convex functions, (5.30) implies a convergence rate of $o(\log(1/\epsilon))$, which is also called *liner time* (figure 5.3).

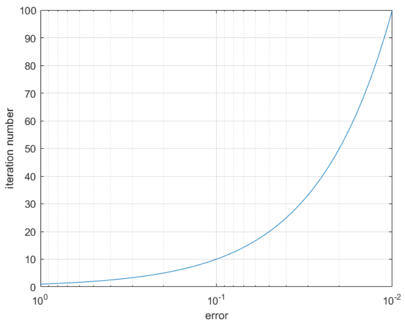

Figure 5.2: Convergence rate $o(1/\epsilon)$.

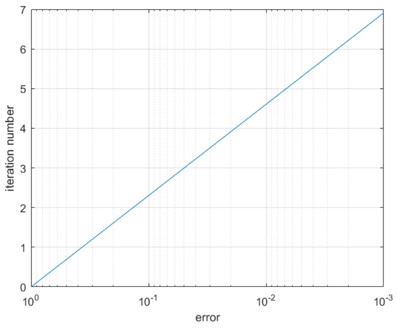

Figure 5.3: Convergence rate $o(\log(1/\epsilon))$.

5.3.3 Co- and contravariant view of the gradient descent method

Recall the gradient descent update formula (5.21),

$$x_{k+1} = x_k - \alpha \nabla f(x_k).$$

Referred to the discussion in section 2.2.1, if the variable vector x_k is a contravariant vector, then the gradient vector $\nabla f(x_k)$ is a covariant vector. In tensor notations, contravariant vectors are denoted as column vectors and covariant vectors are denoted as row vectors. From this perspective, the gradient descent method seems nontrivial as it tries to sum up a column vector with a row vector. Indeed, there is a consequence when we apply a change of basis for the optimization variable x: the variable vector x_k changes contravariantly and the gradient vector $\nabla f(x_k)$ changes covariantly, and the updated variable vector x_{k+1} is generally different from the one formulated in the original basis.

5.4 Optimality conditions for inequality constrained optimization

We consider inequality constrained optimization in the form:

$$\begin{aligned} \min_x \quad & f(x), \\ \text{subject to} \quad & g_i(x) \leq 0, \ i \in \mathcal{I}, \end{aligned} \tag{5.32}$$

where $f, g_i : \mathbb{R}^n \to \mathbb{R}$ are twice continuously differentiable functions, and $\mathcal{I}$ is the finite set of indices,

$$\mathcal{I} = \{1, \dots, m\}. \tag{5.33}$$

The foundation of the optimality conditions for constrained optimization is the *Lagrangian function.* Classically, the Lagrangian function is defined for equality constrained problems, i.e.,

$$\begin{aligned} \min_x \quad & f(x), \\ \text{subject to} \quad & h_i(x) = 0, \ i \in \mathcal{E}. \end{aligned} \tag{5.34}$$

where $f, h_i : \mathbb{R}^n \to \mathbb{R}$ are twice continuously differentiable functions, and $\mathcal{E}$ is a finite set of indices. The associate Lagrangian function writes

$$\mathcal{L}(x) = f(x) + \sum_{i \in \mathcal{E}} \lambda_i h_i(x), \tag{5.35}$$

where λ_i is called the Lagrange multiplier.

The first-order optimality for problem (5.34) is that its associate Lagrangian function being stationary,

$$\nabla_{x,\lambda} \mathcal{L} = 0. \tag{5.36}$$

That is

$$\begin{aligned} \nabla_x f(x) + \sum_{i \in \mathcal{E}} \lambda_i \nabla_x h_i(x) = 0,\ \forall i \in \mathcal{E}, \\ h_i(x) = 0,\ \forall i \in \mathcal{E}, \end{aligned} \tag{5.37}$$

which are the first-order optimality conditions for an equality constrained optimization problem. This is akin to the first-order optimality condition for an unconstrained optimization, which is the stationary of the objective function, i.e,

$$\nabla f(x) = 0. \tag{5.38}$$

However, constrained problems are more sophisticated as, additionally, certain regularity conditions need to be specified to account for the degeneracy at an optimal solution. A degenerate solution results in the non-uniqueness of the Lagrangian multipliers λ_i. Such a regularity condition is also called the constraint qualification. For an introductory presentation, we consider the *linear independence constraint qualification* (LICQ), which requires that the gradients of the active constraints are linearly independent at the solution. The active set $\mathcal{A}$ for a general constrained optimization is defined as follows.

Definition 2 (Active set)**.** For any feasible x, the active set $\mathcal{A}(x)$ contains all the active constraints,

$$\mathcal{A}(x) = \mathcal{E} \cup \{i \in \mathcal{I} : g_i(x) = 0\}. \tag{5.39}$$

In the following, we discuss the optimality conditions for inequality constrained optimization starting from the Lagrangian function (5.35).

5.4.1 First-order optimality

For inequality constrained optimization (5.32), the first-order optimality is more complex than the Lagrangian function being stationary,

$$\nabla_{x,\lambda}\mathcal{L}(x)=0.$$

The reason is that at an optimal solution $x^\star$, the inequalities must not always be active. Let $\tilde{\mathcal{A}}(x^\star)$ denote the active set for $x^\star$, then for inequality constrained optimizations, $\tilde{\mathcal{A}}(x^\star)$ is a subset of $\mathcal{I}$, i.e.,

$$\tilde{\mathcal{A}}(x^\star)\subseteq\mathcal{I}. \tag{5.40}$$

The question is, *how can we give optimality conditions if it is not known a priori, which inequalities are active at the solution?* The well-known Karush-Kuhn-Tucker (KKT) conditions treat this difficulty by separating the active and inactive inequalities.

Theorem 4 (KKT conditions for inequality constrained optimization). *Suppose that $x^\star$ is a local solution for the problem* (5.32) *and that the LICQ holds for inequalities in the active set $\mathcal{A}(x^\star)$, then, there is a unique vector of Lagrangian multipliers $\lambda_i^\star$ such that the following conditions hold at $x^\star$*

$$\nabla_x\mathcal{L}(x^\star)=0, \tag{5.41}$$
$$\lambda_i^\star g_i(x^\star)=0,\ \forall i\in\mathcal{I}, \tag{5.42}$$
$$\lambda_i^\star\geq 0,\ \forall i\in\mathcal{I}, \tag{5.43}$$
$$g_i(x^\star)\leq 0,\ \forall i\in\mathcal{I}. \tag{5.44}$$

The separation of active and inactive inequalities is realized elegantly in (5.42)-(5.44):

1) Active: if $i\in\tilde{\mathcal{A}}(x^\star)$, then, $g_i(x^\star)=0$, and thus $\lambda_i^\star\geq 0$.
2) Inactive: if $i\in\mathcal{I}\setminus\tilde{\mathcal{A}}(x^\star)$, then, $g_i(x^\star)<0$, and thus $\lambda_i^\star=0$.

(5.42) is called the complementarity condition. Furthermore, a *strict complementarity* says that if $g_i(x^\star)=0$, then $\lambda_i^\star>0, \forall i\in\mathcal{A}(x)$.

The KKT conditions are first-order necessary conditions for a local optimal solution.

5.4.2 Second-order optimality

To check whether a solution is a local minimizer, we need to use the second-order sufficient conditions. This can be done by checking the positive definiteness of the projected Hessian matrix of the Lagrangian function onto the null space of the constraint Jacobian matrix. Formerly, set matrix Z, whose columns z_j span the null space of the constraint Jacobian,

$$\{z_j\} = \operatorname{span}\left(\operatorname{Null}\left[\nabla g_i(x^\star)^T\right]_{i\in\mathcal{A}(x^\star)}\right). \tag{5.45}$$

We denote he Hessian of the Lagrangian function about the variable x as

$$\nabla_{xx}\mathcal{L}(x^\star).$$

The projected Hessian of the Lagrangian function is then given as

$$H_Z(x^\star) = Z^T\nabla_{xx}\mathcal{L}(x^\star)Z. \tag{5.46}$$

Theorem 5 (Second-order necessary conditions)**.** *Suppose a KKT solution $x^\star$ that satisfies the strict complementarity. Suppose also that $x^\star$ is a local minimizer of the problem* (5.32)*. Then,*

$$H_Z(x^\star) \succeq 0. \tag{5.47}$$

Theorem 6 (Second-order sufficient conditions)**.** *Suppose a KKT solution $x^\star$ that satisfies the strict complementarity. Suppose also that*

$$H_Z(x^\star) \succ 0. \tag{5.48}$$

Then, $x^\star$ is a local minimizer of the problem (5.32)*.*

CHAPTER 6

A Dynamical systems perspective on optimization

6.1 Introduction

A dynamical system consists of a *phase space*, whose coordinates describe the state at any instant, and a *dynamical rule* that specifies the immediate future of all *state variables*, given only the present values of those same *state variables* (Meiss [60]).

In continuous-time, a dynamical system can be represented by systems of differential equations of the form

$$\frac{dx(t)}{dt} = r(x(t), t), \tag{6.1}$$

where $x \in S$ is the state vector with $S \subset \mathbb{R}^n$ being an open set, $t \in T$ is the time, and $r : S \times T \to S$ is the dynamical rule.

In this work, we mainly consider time-invariant systems, i.e.,

$$\frac{dx(t)}{dt} = s(x(t)), \tag{6.2}$$

where the dynamical rule $s : S \to S$ is time-independent.

Dynamical systems approaches have been used to study optimization methods in many works of literature. Extensive studies on the connections between interior-point flows with linear programming methods can be found in Helmke et al. [34, Chapter 4] and the references therein. In Alvarez et al. [5], the authors presents a second-order gradient-like dynamical system for optimization and mechanics. For quadratic programming problems, Dörr et al. [25] proposes a dynamical system, which results in trajectories that converge to the saddle point of the associated Lagrangian function. Su et al. [84] studies the celebrated Nesterov's accelerated gradient method using a dynamical system as the analysis tool. In Lessard et al. [51], a framework based on dynamical systems is proposed to analyze and design first-order unconstrained optimization methods. Recently, dynamical systems are used to study optimization algorithms in the context of machine learning theory (see, e.g., Arora et al. [8], Chizat et al. [21], and Jordan [42]).

In some literature, optimization methods that use dynamical systems are called trajectory methods. These methods construct optimization paths in a way so that one or all solutions to the optimization problem are *a priori* known to lie on these paths (Diener [24]). Typically, these optimization paths are solution trajectories to ODE of first or second-order. Trajectory methods are mainly studied for unconstrained optimizations for finding local solutions (Behrman [12], Botsaris [16]), and global solutions (Griewank [30], Snyman et al. [82]). Studies for constrained optimization are, however, very limited, see Ali et al. [2] and Wang et al. [87], and the references therein.

6.2 First-order optimization methods: a continuous-time dynamical systems perspective

To study the behavior of the method, we use a dynamical systems perspective, i.e., we use an ODE to model the iterative update process of the optimization. Without loss of generality, assume that the iterative variable update formula of a first-order optimization method writes

$$x_{k+1} = x_k + \alpha\omega(x_k), \tag{6.3}$$

where k is the iteration number, $\alpha > 0$ is the step size, and $\omega : \mathbb{R}^n \to \mathbb{R}^n$ is a smooth function. The optimization path is the discrete trajectory that is the set of x_k, $k = 1, 2, ...p$, with x_p being the last iterate.

Rewrite (6.3), we get

$$\frac{x_{k+1} - x_k}{\alpha} = \omega(x_k). \tag{6.4}$$

We study the limiting behavior as the step size decreases, i.e., $\alpha \downarrow 0$,

$$\lim_{\alpha \to 0} \frac{x_{k+1} - x_k}{\alpha} = \omega(x_k). \tag{6.5}$$

Assume the above limit exists, then, as the step size approaches zero, the solution trajectory becomes continuous. In continuous-time, the equation that models the trajectory writes

$$\frac{dx}{dt} = \omega(x(t)), \tag{6.6}$$

which is a dynamical system that is a system of ordinary differential equations. The link between a dynamical system and a first-order method that has the form (6.3) is then clear: The optimization algorithm represented in system (6.3) can be seen as applying the Euler method for solving the continuous-time system (6.6).

We are especially interested in integral curves of the corresponding dynamical system and their behavior over long time spans. The former is the asymptotic optimization path; and the latter reveals the convergence behavior of the optimization method.

6.3 Initial value problem

In iterative optimizations, there is almost always an initial guess of the variables to start with. Expressed in the language of ODEs, it is an initial value problem (IVP), which is also called the Cauchy problem. In the following, some of the basics of the initial value problem are explained.

An initial value problem is a system that contains a differential equation and an initial condition. For the time-invariant dynamical system (6.2), an IVP writes

$$\begin{cases} \dfrac{dx(t)}{dt} = s(x(t)), \\ x(t_0) = x_0, \end{cases} \tag{6.7}$$

where $s : S \subset \mathbb{R}^n \to \mathbb{R}^n$ with S being an open set, and $x_0 \in S$. A solution to an IVP is a function x that is the solution to the differential equation and satisfies the initial condition

$$x(t_0) = x_0. \tag{6.8}$$

Example

Consider the initial value problem

$$\begin{cases} \dfrac{dx(t)}{dt} = x, \\ x(0) = 4. \end{cases} \tag{6.9}$$

Analytical solution

Solving the differential equation we get

$$x = Ce^t,$$

where C is some constant. By the initial condition, we have

$$Ce^0 = 4,$$

therefore, $C = 4$ and

$$x = 4e^t.$$

Numerical solution

The simplest numerical method to solve the initial value problem may be the Euler forward method. We solve the IVP (6.9) for $0 \le t \le 3$ using the numerical scheme shown in Algorithm 6.1.

Algorithm 6.1: Euler method for the initial value problem (6.9)

```
**Input:** Time step $\Delta t = 0.05$, initial condition $t_0 = 0$, $x_0 = 4$
$x \leftarrow x_0$
**repeat**
    $x \leftarrow x + \Delta t\, x$; $t = t + \Delta t$.
**until** $t = 3.0$;
```

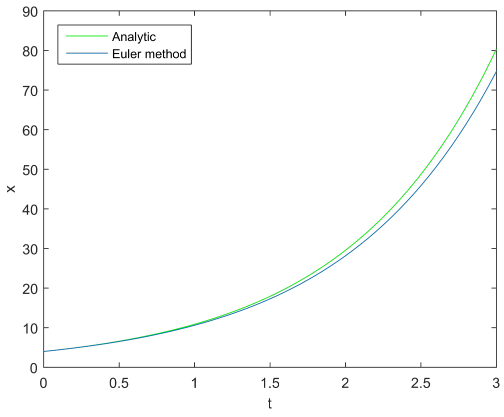

Figure 6.1: Analytic solution (green line) and numerical solution (blue line) for the initial value problem (6.9).

Discussion

Both analytical and numerical solution are plotted in figure 6.1. We note that the result is plotted as the function x of the time t. It is to be distinguished with the trajectory of x in the phase space of a dynamical system. The latter trajectory is the asymptotic optimization path in the variable space.

6.3.1 Uniqueness and existence of the solution of an initial value problem

The Picard's existence theorem gives the conditions under which an initial value problem has a unique solution. Translated to an optimization method, a unique solution means a unique optimization path that leads to the same optimal solution whenever the same initial condition is given. In this sense, the required conditions of uniqueness and existence constrain the problems that an optimization algorithm can be applied for.

Consider the initial value problem (6.7). Formerly, if s is uniformly *Lipschitz continuous* in x and *continuous* in t, then for some value $\epsilon > 0$, there exists a unique solution $x(t)$ to the initial value problem on the interval $\left[t_0 - \epsilon, t_0 + \epsilon\right]$. These conditions are deeply related to convergence analysis of first-order optimization methods. For example, Theorem 2 gives the convergence rate for the gradient descent method under the Lipschitz continuity condition for the function gradients.

6.3.2 Maximal interval of existence

If the conditions of the Picard's existence theorem are satisfied, then, the initial value problem (6.7) has a unique solution $x(t)$ defined on a maximal interval of existence T_{x_0}. Following Grant [29] and Sakka [75], some properties of the maximal interval of existence are summarized below.

Lemma 1 (Maximal interval of existence)**.** *Consider the initial value problem* (6.7)*. Let D be an open subset of $\mathbb{R}^n$ and assume that s is continuously differentiable on D, i.e., $s \in C^1(D)$. Then, for each point $x_0 \in D$, there is a maximal interval $T_{x_0} = (\alpha, \beta)$ on which the IVP has a unique solution $x(t; x_0)$. Furthermore,*

1. *If $\beta < \infty$ $(\alpha > -\infty)$ and if*

$$\lim_{t \to \beta^-} x(t; x_0) = L \quad (\lim_{t \to \alpha^+} x(t; x_0) = L),$$

then $L \in \partial D$.

2. *If the above limit exists and $L \in D$, then $\beta = \infty$, $s(L) = 0$.*

To unpack the lemma, the first point says that if the maximal interval of existence is finite and the solution of the IVP converges to a point $x^\star = L$, then the point $x^\star$ must lie on the boundary of the open set D. Intuitively, it suggest that if the maximal interval of existence is finite and there exists a unique solution within this time interval, then the solution trajectory will go out of the open set D.

The second point says that if the IVP converges to a point $x^\star$ inside the open set D, then the maximal interval of existence is infinite and $s(x^\star)=0$.

Example

To illustrate these properties, we consider a gradient flow in 2D

$$\begin{cases} \dfrac{d\mathbf{x}(t)}{dt} = -\nabla f(\mathbf{x}(t)), \\ \mathbf{x}(0) = (-12, -10), \end{cases} \tag{6.10}$$

where $f(x_1, x_2) = \frac{(x_1-3)^2}{25} + \frac{(x_2-5)^2}{4}$ is a multivariate function.

In figure 6.2, we show trajectory $\widehat{ABC}$ that is the solution to the initial value problem (6.10).

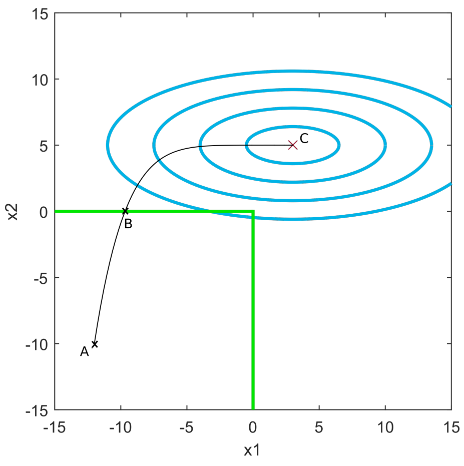

Figure 6.2: Maximal interval of existence on set Ω_1 and Ω_2.

We consider two open sets, the first is defined by the level set $f(\mathbf{x}(0))$,

$$\Omega_1 = \left\{(x_1, x_2) : f(x_1, x_2) < f(-12, -10)\right\}. \tag{6.11}$$

The second open set is defined as

$$\Omega_2 = \left\{(x_1, x_2) : x_1 < 0, x_2 < 0\right\}. \tag{6.12}$$

The boundary $\partial\Omega_2$ is illustrated as green lines in the figure.
By Lemma 1, we can make the following statements:

1) The time for the trajectory to travel from point A to C is infinite, since the gradient flow converges to C and C is a point inside of the open set Ω_1.

2) The time for the trajectory to travel from point A to B is finite, since B is a point on the boundary of the open set Ω_2.

To understand the results, we can interpret the gradient flow (6.10) as an equation of motion, i.e.,

$$\frac{dx}{dt} = v(x), \tag{6.13}$$

where the velocity is equal to the negative gradient $v(x) = -\nabla f(x)$. Obviously, along the path $\widehat{AB}$ from A to B, the velocity v is strictly nonzero, and its direction is positively correlated to the path direction. Further, $\widehat{AB}$ has a finite length. Therefore, the travel time (existence time) of the solution trajectory from A to B is finite. On the other hand, at point C the velocity $v(x)$ vanishes. By the definition of function f, the velocity field is continuous. Thus, the closer the solution $x(t)$ is getting to point C, the smaller the velocity becomes, which results in an infinite determination of the travel time.

6.4 Linear dynamical system

A special class of the dynamical system is the *linear dynamical system.* In continuous-time, it may be written as

$$\frac{dx(t)}{dt} = A(t)x(t) + B(t)u(t), \tag{6.14}$$

where $x(t) \in \mathbb{R}^n$ is the state vector, $u(t) \in \mathbb{R}^m$ is the control vector, $A \in \mathbb{R}^{n\times n}$ is the dynamics matrix, and $B \in \mathbb{R}^{n\times m}$ is the input matrix.

If the system is time-invariant and autonomous, then A is constant and B vanishes, therefore it writes

$$\frac{dx(t)}{dt} = Ax(t). \tag{6.15}$$

The solution of a nonlinear dynamical system (6.2) near a fixed point can be well-approximated by a linear system (6.15). For system (6.2), a fixed point is a point $x^\star$ such that $s(x^\star) = 0$. Assume that the Jacobian matrix for $s(x)$ at $x^\star$ is $A^\star$, then, by Taylor approximation, we have

$$\frac{dx(t)}{dt} \approx s(x^\star) + A^\star(x(t) - x^\star). \tag{6.16}$$

Therefore,

$$\frac{dx(t)}{dt} \approx A^\star x(t) - A^\star x^\star, \tag{6.17}$$

which is a linear dynamical system.

The linear dynamical system thus provides insight to the more difficult nonlinear systems locally at fixed points. In the following, we show some properties of the gradient descent method by looking at it through the lens of linear dynamical systems.

Asymptotic local behavior of the gradient descent method

We consider unconstrained optimization problem (5.3) and use the gradient flow to show asymptotic behavior of the gradient descent method near critical points. Recall the gradient flow for $f(x)$,

$$\frac{dx(t)}{dt} = -\nabla f(x(t)). \tag{6.18}$$

Assume a critical point $x^\star$ of $f(x)$, and choose a local Cartesian frame such that $x^\star$ is the origin, then the gradient flow above can be well-approximated by the linear system at $x^\star$,

$$\frac{dx(t)}{dt} = -Hx(t), \tag{6.19}$$

where H is the Hessian matrix of f and is symmetric by the assumed smoothness of $f(x)$. We further assume H is full rank.

Assume an initial condition near the critical point $x^\star$,

$$x(0) = x_0, \tag{6.20}$$

then, the initial value problem has the approximate solution

$$x(t) = e^{-Ht} x_0, \tag{6.21}$$

where e^{-Ht} is a matrix exponential (see Appendix A.7). Since H is a real symmetric matrix, it can be diagonalized by orthogonal matrices,

$$H = UDU^T, \tag{6.22}$$

where U column-wise contains the eigenvectors of H, and D is the diagonal matrix that contains the eigenvalues

$$D = \mathrm{diag}(\lambda_1, \lambda_2, ..., \lambda_n). \tag{6.23}$$

By the property of the matrix exponential, we have

$$e^{-Ht} = U e^{-Dt} U^T. \tag{6.24}$$

Therefore, the approximate solution of the gradient flow near the critical point $x^\star$ is

$$x(t) = U e^{-Dt} U^T x_0 = U \mathrm{diag}(e^{-\lambda_1 t}, e^{-\lambda_2 t}, ..., e^{-\lambda_n t}) U^T x_0. \tag{6.25}$$

If H is positive-definite, then, as $t \to +\infty$,

$$x(t) \to 0, \tag{6.26}$$

for any initialization x_0 close enough to the critical point $x^\star$. Therefore, the solution of the gradient flow converges to this critical point.

If H has a strict negative eigenvalue λ_i for some $i \in [1, 2, ..., n]$, then, as $t \to +\infty$,

$$|x(t)| \geq |e^{-\lambda_i t} u_i^T x_0|, \tag{6.27}$$

where u_i is the i-th column of the matrix U, which is the i-th eigenvector of H.

Therefore, if $u_i^T x_0 \neq 0$, i.e., if x_0 does not lie in the right null space of the eigenvector u_i, then

$$x(t) \rightarrow \infty, \tag{6.28}$$

in exponential time.

Example

Consider function

$$f(x_1, x_2) = x_1^2 + 3x_1 x_2 - \frac{1}{2} x_2^2, \tag{6.29}$$

which has the gradient

$$\nabla f(x_1, x_2) = \left[2x_1 + 3x_2, 3x_1 - x_2\right]^T, \tag{6.30}$$

and the Hessian matrix

$$H = \begin{bmatrix} 2 & 3 \\ 3 & -1 \end{bmatrix}. \tag{6.31}$$

Apply eigendecomposition to H, we obtain

$$H = UDU^T, \tag{6.32}$$

where

$$U = \begin{bmatrix} 0.5257 & -0.8507 \\ -0.8507 & -0.5257 \end{bmatrix}, \tag{6.33}$$

and

$$D = \begin{bmatrix} -2.8541 & 0 \\ 0 & 3.8541 \end{bmatrix}. \tag{6.34}$$

The eigenvector that correspond to the negative eigenvalue $\lambda_1 = -2.8541$ is $u_1 = [0.5257, -0.8507]^T$. In figure 6.3, the negative gradient vectors around the critical point $x^\star = (0, 0)$ are shown. The red line corresponds to the eigenvector u_1. The green line corresponds to the eigenvector u_2, which is the right null space of u_1 at $x^\star$. As can be observed in the figure below, only initializations that lie on the green line will converge to the critical point $x^\star$.

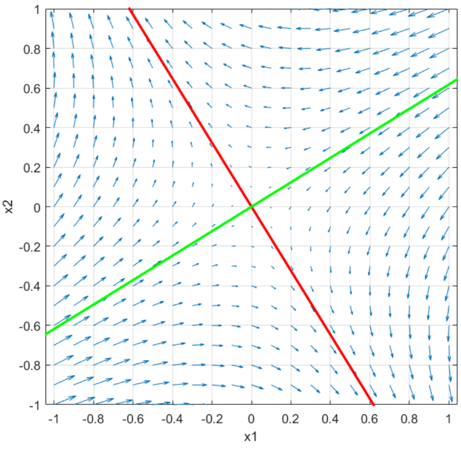

Figure 6.3: Negative gradient vectors near the critical point of the function (6.29).

CHAPTER 7

Modified search direction

7.1 Introduction

This chapter presents the design of the present method: a modified search direction method (MSDM) for inequality constrained optimization (Chen et al. [19]). We consider the problem

$$\begin{aligned} \min_x \quad & f(x) \\ \text{subject to} \quad & g_i(x) \leq 0, \ i = 1, \ldots, m \end{aligned} \tag{7.1}$$

where $f, g_1, \ldots, g_m : \mathbb{R}^n \to \mathbb{R}$ are twice continuously differentiable.

The design of the method is inspired by the singular value decomposition (SVD). First, the basics of the SVD are introduced. Then, the design of the method for single inequality constrained optimization is presented. Finally, we discuss the generalization of the method to multiple constraints that uses the logarithmic barrier function.

7.2 Basics of singular value decomposition

SVD is a generalization of the eigendecomposition of a symmetric matrix to any $m \times n$ matrix. In this work, we only consider real matrices. Formally, the singular-value decomposition of a matrix **M** is a factorization of the form,

$$\mathbf{M} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T = \sum_{i=1}^{\min(m,n)} \boldsymbol{\sigma}_i \mathbf{u}_i \mathbf{v}_i^T \tag{7.2}$$

where **M** is an $m \times n$ matrix, **U** is an $m \times m$ orthogonal matrix, $\boldsymbol{\Sigma}$ is an $m \times n$ rectangular diagonal matrix with non-negative numbers on the diagonal, **V** is an $n \times n$ orthogonal matrix and $\mathbf{V}^T$ is the transpose of **V**.

The diagonal entries σ_i of $\boldsymbol{\Sigma}$ are singular values of **M**. The vectors $\mathbf{u}_i$ and $\mathbf{v}_i$, which are the columns of **U** and **V**, are the *left-singular vectors* and *right-singular vectors* of **M**, respectively. Since **U** and $\mathbf{V}^T$ are orthogonal, the columns of each of them form a set of orthonormal vectors, which can be regarded as base vectors (Golub et al. [28]).

By the orthogonality, by (7.2), we have

$$\mathbf{M}\mathbf{v}_k = \sigma_k \mathbf{u}_k \tag{7.3}$$

7.3 The design of the modified search direction

The idea behind the design of the modified search direction is to find a descent direction of the objective function, such that the centrality conditions are approached. In this section, we first introduce the central path and the centrality condition. Then, the design of the modified search direction for single inequality constrained problem is introduced.

7.3.1 Central path and centrality conditions

Central path is a "fundamental mathematical object" in studying inequality constrained optimization (Bayer et al. [11]). One way to motivate the definition of the central path is to use the logarithmic barrier method, which belongs to the class of the interior-point method.

The idea of the logarithmic barrier function is to approximately formulate the inequality constrained problem as an equality constrained problem to which Newton's method can be applied (Boyd et al. [17]). Consider problem (7.1) and assume it is convex. The logarithmic barrier is defined as

$$\Phi(x) = -\sum_{i=1}^{m} \log(-g_i(x)),\ i = 1, \ldots, m, \tag{7.4}$$

and the approximated optimization problem is given as

$$\begin{aligned} \text{minimize}\quad & f(x) + \frac{1}{t}\Phi(x) \\ & = f(x) + \frac{1}{t}\sum_{i=1}^{m} -\log(-g_i(x)), \end{aligned} \tag{7.5}$$

where t is a positive parameter. The logarithmic barrier (7.4) grows without bound as $g_i(x) \to 0^-$. As $t \uparrow +\infty$, the solution of the approximated unconstrained problem (7.5) converges to the solution of the original constrained problem (7.1) asymptotically.

From the approximated problem (7.5), the central path can be defined.

Definition 3 (Central path)**.** The *central path* associated with problem (7.1) is defined as the set of *central points* $x^\star(t)$, $t > 0$. The necessary and sufficient conditions, which characterize the central point, are

$$\begin{aligned} 0 &= \nabla f(x^\star(t)) + \frac{1}{t}\nabla\Phi(x^\star(t)) \\ &= \nabla f(x^\star(t)) + \frac{1}{t}\sum_{i=1}^{m} \frac{1}{-g_i(x^\star(t))}\nabla g_i(x^\star(t)). \end{aligned} \tag{7.6}$$

Although the approximated problem (7.5) has the form of unconstrained optimization, and the condition (7.6) looks identical to the corresponding first-order optimality condition, it shall not be considered as a regular unconstrained problem. In fact, two things are hidden in the reformulated problem (7.5): 1) the parameter t must be strictly positive, and 2) the logarithmic barrier function $\Phi(x)$ is defined only if $g_i(x) < 0, i = 1, 2, \ldots, m$.

In the following, we elaborate the central point condition (7.6). First, we define

$$\lambda_i^\star(t) = \frac{1}{-t g_i(x^\star(t))}, \; i = 1, \ldots, m. \tag{7.7}$$

$\lambda_i^\star(t)$ is called *dual feasible* if the condition (7.6) holds (Boyd et al. [17]). We then have the following *centrality conditions* at a central point $x^\star$,

$$\begin{aligned} & g_i(x^\star) \leq 0, \; i = 1, \ldots, m \\ & \lambda^\star(t) \succeq 0, \\ & \nabla f(x^\star) + \sum_{i=1}^{m} \lambda_i^\star(t) \nabla g_i(x^\star) = 0, \\ & -\lambda_i^\star(t) g_i(x^\star) = 1/t, \; i = 1, \ldots, m \end{aligned} \tag{7.8}$$

The only difference between the KKT conditions (5.41)- (5.44) and the centrality conditions (7.8) is that the complementarity condition (5.42) is replaced by $-\lambda_i^\star g_i(x^\star) = 1/t$. Obviously, the centrality conditions recover the KKT conditions with the limiting behavior as $t \to +\infty$.

Figure 7.1 shows the central path for a linear programming problem with 7 inequality constraints. At the central point $x^\star(t = t_k)$, the centrality conditions (7.8) are fulfilled, i.e., the objective gradient ∇f is parallel with the gradient of the logarithmic barrier $\nabla \Phi$.

7.3.2 The steepest descent direction for the objective and constraint function in ℓ^2-norm

We consider single inequality constrained optimization. In section 5.3, we have shown that the direction of the gradient descent is the steepest descent direction in Euclidean norm,

$$d_f(x) = -\frac{\nabla f(x)}{|\nabla f(x)|}. \tag{7.9}$$

Similarly, the ‘‘steepest descent direction’’ for the constraint function can be written as

$$d_g = -\frac{\nabla g(x)}{|\nabla g(x)|}. \tag{7.10}$$

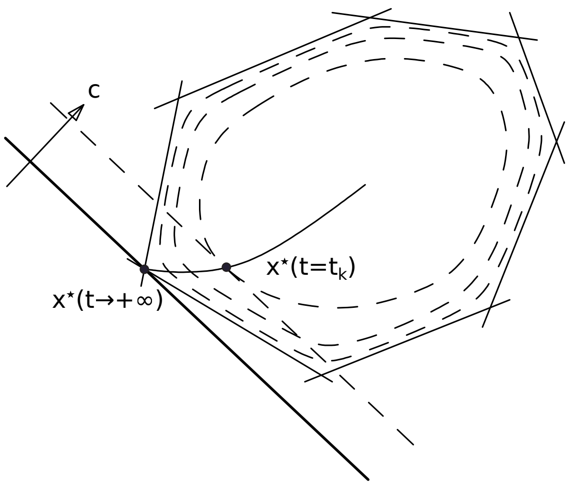

Figure 7.1: Central path for a linear programming problem with 7 linear constraints. The dashed curves show the three contour lines of the logarithmic barrier function. The level curve of the objective function is tangent to the barrier function's contour line at $x^{\star}(t_k)$. The central path converges to the optimal point $x^{\star}(t \to \infty)$ (Boyd et al. [17]).

The directional derivative along any vector **s** for function $f(x)$ and $g(x)$ reads

$$\begin{aligned} \nabla_{\mathbf{s}} f(x) &= \nabla f(x)^T \mathbf{s}, \\ \nabla_{\mathbf{s}} g(x) &= \nabla g(x)^T \mathbf{s}. \end{aligned} \tag{7.11}$$

Note that $\nabla_{\mathbf{s}} f(x)$ and $\nabla_{\mathbf{s}} g(x)$ are now scalars. By (7.11), we have

$$\begin{aligned} \frac{\nabla_{\mathbf{s}} f(x)}{|\nabla f(x)|} &= \frac{\nabla f(x)^T}{|\nabla f(x)|}\mathbf{s}, \\ \frac{\nabla_{\mathbf{s}} g(x)}{|\nabla g(x)|} &= \frac{\nabla g(x)^T}{|\nabla g(x)|}\mathbf{s}, \end{aligned} \tag{7.12}$$

which can be written in the matrix form

$$\begin{pmatrix} \frac{\nabla_{\mathbf{s}} f(x)}{|\nabla f(x)} \\ \frac{\nabla_{\mathbf{s}} g(x)}{|\nabla g(x)} \end{pmatrix} = \begin{pmatrix} \frac{\nabla f(x)^T}{|\nabla f(x)|} \\ \frac{\nabla g(x)^T}{|\nabla g(x)|} \end{pmatrix} \mathbf{s}. \tag{7.13}$$

Set

$$\mathbf{m} = \begin{pmatrix} \frac{\nabla f(x)^T}{|\nabla f(x)|} \\ \frac{\nabla g(x)^T}{|\nabla g(x)|} \end{pmatrix}, \tag{7.14}$$

which is a sensitivity matrix that contains the ℓ^2-normalized gradient of the objective and constraint function. This sensitivity matrix maps any search direction **s** to the directional derivatives of the objective function $f(x)$ and constraint function $g(x)$, both scaled by the inverse of the ℓ^2-norm of the respective function gradient. The input-output system represented by the sensitivity matrix **m** is studied by SVD in the following.

7.3.3 Singular value decomposition of the sensitivity matrix m

We apply SVD to the sensitivity matrix **m**:

$$\mathbf{m} = \mathbf{U\Sigma V}^T = \sum_{i=1}^{\min(2,n)} \boldsymbol{\sigma}_i \mathbf{u}_i \mathbf{v}_i^T, \tag{7.15}$$

where n is the number of control (design) variables. For large-scale problems, we have $n \gg 2$. Therefore, two left-singular vectors $\mathbf{u}_i$ and two right-singular vectors $\mathbf{v}_i$ (i =1 or 2) are obtained by SVD.

Compare (7.3) with (7.13) and by the definition of SVD, we can make the following statements:

1. the left-singular vector $\mathbf{u}_i$ has two entries and the right-singular vector $\mathbf{v}_i$ has n entries.
2. The first entry $\mathbf{u}_{i1}$ corresponds to the directional change in the objective function $\frac{\nabla_{\mathbf{s}} f(x)}{|\nabla f(x)|}$ and the second entry $\mathbf{u}_{i2}$ corresponds to the directional change in the constraint function $\frac{\nabla_{\mathbf{s}} g(x)}{|\nabla g(x)|}$.
3. The right-singular vector $\mathbf{v}_i$ corresponds to the direction **s**.

A variational view

Recall the definition for the variation (2.38):

$$y(x,\alpha) = y^{\star}(x) + \alpha(\tilde{y}(x) - y^{\star}(x)),$$

and (2.39):

$$\delta = \left.\frac{\partial}{\partial \alpha}\right|_{\alpha=0}.$$

And compare the functional variation (2.44) with the definition of the directional derivative of a function $f(x)$ along direction $\mathbf{s}$,

$$\nabla_{\mathbf{s}} f(\mathbf{x}) = \nabla f(\mathbf{x})^T \mathbf{s} = \left.\frac{\partial}{\partial \alpha}\right|_{\alpha=0} f(\mathbf{x} + \alpha \mathbf{s}),$$

we may interpret the results of applying SVD to the sensitivity matrix $\mathbf{m}$ via a variational perspective,

$$\mathbf{m}\delta\mathbf{v}_i = \sigma_i \delta\mathbf{u}_i, i = 1,2. \qquad (7.16)$$

Standardization paradigm for the singular vectors

In an optimization process, the search direction should be a proper descent direction of the objective function if the optimization problem is formulated as a minimization problem. According to previous discussions, each $\delta\mathbf{v}_i$ can be used as a base search direction for the variable update. We propose a standardization paradigm based on the orthogonality of the singular vectors.

- $\mathbf{v}_1$: by taking $\delta\mathbf{v}_1$ as the variable update, we obtain a change in the objective as well as in constraint function $[\frac{\nabla_{\mathbf{s}} f(x)}{|\nabla f(x)|}, \frac{\nabla_{\mathbf{s}} g(x)}{|\nabla g(x)|}]^T = \sigma_1 \delta\mathbf{u}_1$, which is a *decrease* in the objective function and an *increase* in the constraint function.
- $\mathbf{v}_2$: by taking $\delta\mathbf{v}_2$ as the variable update, we obtain a change in the objective as well as in the constraint function $[\frac{\nabla_{\mathbf{s}} f(x)}{|\nabla f(x)|}, \frac{\nabla_{\mathbf{s}} g(x)}{|\nabla g(x)|}]^T = \sigma_2 \delta\mathbf{u}_2$, which is a *decrease* in the objective function and a *decrease* in the constraint function.

The signs of the singular vectors can be changed to meet the paradigm by simultaneously multiplying -1 to the left-singular vector and the respective right-singular vector.

Based on the standardization paradigm, a proper descent direction of the objective function can have the form

$$\mathbf{s}_\beta = \beta_1 \mathbf{v}_1 + \beta_2 \mathbf{v}_2, \tag{7.17}$$

where $\beta_1, \beta_2 \in \mathbb{R}$ are non-negative parameters.

Characteristics of $\mathbf{v}_1$ and $\mathbf{v}_2$ at central path

At the central path, by (7.6), the gradients of the objective and constraint function are parallel. Thus, the sensitivity matrix $\mathbf{m}$ has rank 1, and

$$\begin{aligned} \sigma_2 &= 0, \\ \mathbf{m} &= \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T. \end{aligned} \tag{7.18}$$

Resulting in the variable update $\delta \mathbf{v}_1$ being parallel with the steepest descent direction $d_f = -\frac{\nabla f(x)}{|\nabla f(x)|}$,

$$d_f^T \mathbf{v}_1 = 1. \tag{7.19}$$

7.3.4 Modifying the steepest descent direction

We denote the angle between $\mathbf{v}_1$ and d_f as α_1 and the angle between $\mathbf{v}_2$ and d_f as α_2, then,

$$\begin{aligned} \cos \alpha_1 &= \frac{d_f^T \mathbf{v}_1}{|d_f||\mathbf{v}_1|}, \\ \cos \alpha_2 &= \frac{d_f^T \mathbf{v}_2}{|d_f||\mathbf{v}_2|}. \end{aligned} \tag{7.20}$$

Thus, we can rewrite the steepest descent direction d_f as

$$d_f = \cos \alpha_1 \mathbf{v}_1 + \cos \alpha_2 \mathbf{v}_2. \tag{7.21}$$

Modifying the formula (7.21) by introducing a parameter $c \geq 1$ to increase the contribution of $\mathbf{v}_2$, we get the modified search direction

$$\mathbf{s}_c = \cos \alpha_1 \mathbf{v}_1 + c \cdot \cos \alpha_2 \mathbf{v}_2. \tag{7.22}$$

The modified search direction $\mathbf{s}_c$ fulfills (7.17) and is thus a decent direction of the objective function.

By (7.19) and recall that d_f and $\mathbf{v}_1$ are unit vectors, then, at the central path, it holds

$$\cos\alpha_1 = 1. \tag{7.23}$$

We call $\cos\alpha_1$ the *correlation factor* for the centrality conditions and use it to monitor the optimization process. The closer the correlation factor $\cos\alpha_1$ is to 1, the better the centrality conditions are fulfilled.

The basic characteristics of the modified search direction is demonstrated with a computational example in the next section.

7.4 Basic characteristics

Consider the following 2D optimization problem:

$$\begin{aligned} &\min_{x_1,x_2} && f = (x_1-2)^2 + (x_2-2)^2 \\ &\text{subject to} && g = -\frac{1}{10}(x_1-3)^2 - x_2 + 3 \le 0. \end{aligned} \tag{7.24}$$

Singular value decomposition of the sensitivity matrix m

We choose an initial point in the feasible domain $\mathbf{x} = (-12, -4)$ and its sensitivity matrix $\mathbf{m}$ can be calculated as follows:

$$\mathbf{m} = \begin{pmatrix} \frac{1}{|\nabla f|}\frac{\partial f}{\partial x_1} & \frac{1}{|\nabla f|}\frac{\partial f}{\partial x_2} \\ \frac{1}{|\nabla g|}\frac{\partial g}{\partial x_1} & \frac{1}{|\nabla g|}\frac{\partial g}{\partial x_2} \end{pmatrix} = \begin{pmatrix} -0.9191 & -0.3939 \\ 0.9487 & -0.3162 \end{pmatrix}. \tag{7.25}$$

Applying SVD to the sensitivity matrix $\mathbf{m}$, we obtain the left-singular vectors that arranged in the standardization paradigm proposed in section 7.3.3

$$\mathbf{u}_1 = \begin{pmatrix} -0.7071 \\ 0.7071 \end{pmatrix}, \quad \mathbf{u}_2 = \begin{pmatrix} -0.7071 \\ -0.7071 \end{pmatrix}, \tag{7.26}$$

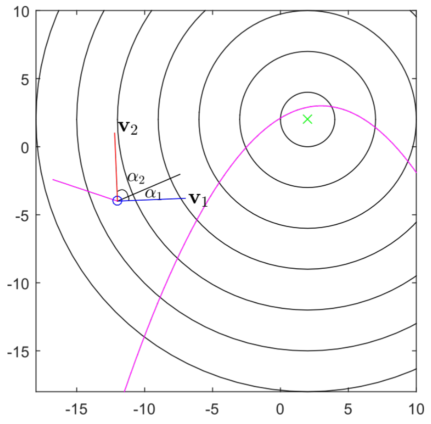

Figure 7.2: Vectors $\mathbf{v}_1, \mathbf{v}_2$ at $\mathbf{x} = (-12, -4)$ for problem (7.24).

and the respective right-singular vector $\mathbf{v}_1$ and $\mathbf{v}_2$,

$$\mathbf{v}_1 = \begin{pmatrix} 0.9991 \\ 0.0416 \end{pmatrix}, \quad \mathbf{v}_2 = \begin{pmatrix} -0.0416 \\ 0.9991 \end{pmatrix}, \tag{7.27}$$

as well as the singular values,

$$\sigma_1 = 1.3219, \quad \sigma_2 = 0.5026. \tag{7.28}$$

The right-singular vectors $\mathbf{v}_1, \mathbf{v}_2$ and their related angles α_1, α_2 (defined in (7.20)) are illustrated in figure 7.2.

Variable update using $\mathbf{v}_1$ and $\mathbf{v}_2$

We show computational experiments that update the variables using $\mathbf{v}_1$ and $\mathbf{v}_2$. The iterative update formula reads

$$x_{k+1} = x_k + \alpha \mathbf{v}_i, \quad i = 1, 2, \tag{7.29}$$

where α is some tuned step size.

Figure 7.3 shows the consecutive variable updates using $\mathbf{v}_1$: at each iteration, we obtain a new variable that decreases the objective function and increases the constraint function. The resulting optimization trajectory reaches the boundary of the constraint faster than the gradient descent trajectory.

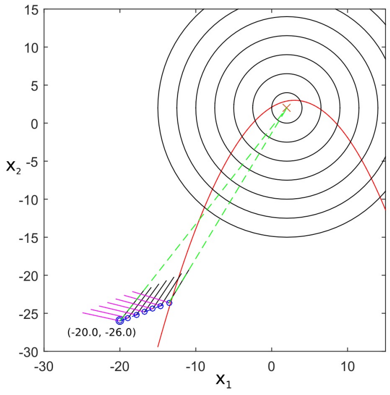

Figure 7.3: An iterative update of the variable using $\mathbf{v}_1$ for the 2D optimization problem (7.24).

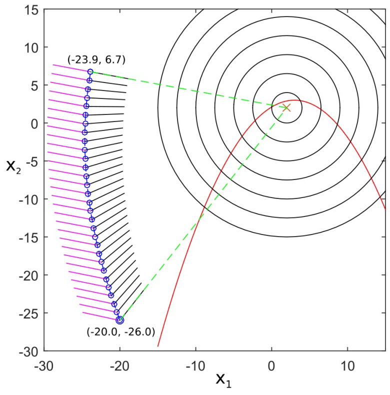

Figure 7.4: An iterative update of the variable using $\mathbf{v}_2$ for the 2D optimization problem (7.24).

Figure 7.4 shows the consecutive variable updates using $\mathbf{v}_2$: at each iteration, we obtain a new variable that decreases both the objective function and the constraint function. The optimization converges to a point $\mathbf{x}^\star$ near $(-23.9, 6.7)$, where the constraint gradient is parallel to the objective function gradient. At $\mathbf{x}^\star$, the centrality conditions are fulfilled.

Figure 7.5 shows the plot for the *correlation factor* that monitors the optimization progress. We observe that the *correlation factor* increases steadily and converges to 1. Therefore, the centrality conditions are approached iteratively when using $\mathbf{v}_2$ for the variable update. In the following, we show various results obtained by applying the modified search direction to the 2D optimization problem (7.24).

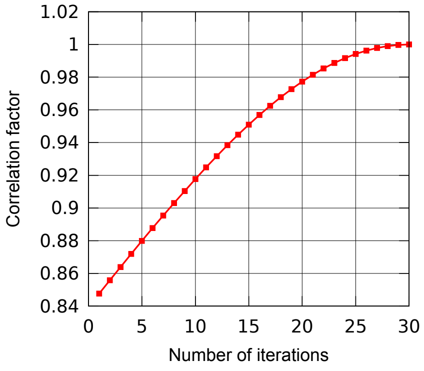

Figure 7.5: The correlation factors during the variable update process in figure 7.4 are shown.

Variable update using the modified search direction

We show computational results that use the modified search direction (7.22) for the variable update. In each iteration, we update the variable with

$$x_{k+1} = x_k + \alpha \mathbf{s}_c, \tag{7.30}$$

where $\alpha > 0$ is some step size.

In figure 7.6, the modified search direction method with $c = 10.0$ is shown. The blue squares indicate the optimization variable at each optimization iteration, the black lines show the negative objective gradient directions; and the pink lines show the negative constraint gradient directions.

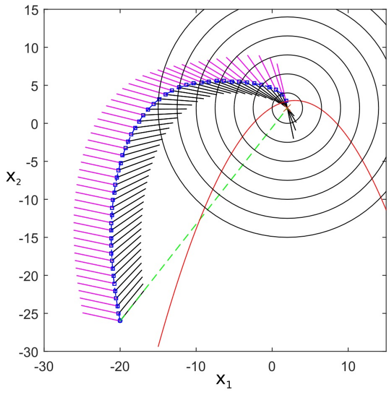

Figure 7.6: Optimization with the modified search direction with parameter $c = 10.0$.

The initial variable is $\mathbf{x}_0 = (-20, -26)$. As one can observe, while the optimization proceeds, the angle between the black and pink lines increases until both lines become close to parallel. It approaches the optimal solution by traversing the interior of the feasible domain.

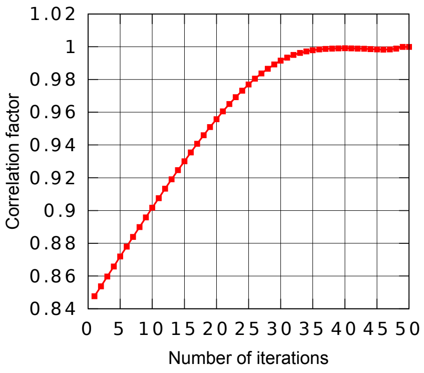

Figure 7.7: Correlation factors that monitor the optimization process shown in figure 7.6.

In figure 7.7, we plot the *correlation factor* of the optimization process that is illustrated in figure 7.6. The *correlation factor* increases iteratively during the optimization process. The variable updates follow the optimization path where the *correlation factor* is close to 1 until the constraint becomes active. In other words, the centrality conditions are approached iteratively, and a solution is found by following the central path.

In figure 7.8, we show the different optimization paths for different parameters c. One can observe: the bigger the chosen parameter c, the further the optimization trajectory is getting pushed away from the boundary of the constraint.

Figure 7.9 shows the optimization processes with different initializations, and the parameter $c = 5.0$ is chosen. Starting from six different points, the modified search direction method can successfully achieve the local minimum by traversing inside the feasible domain.

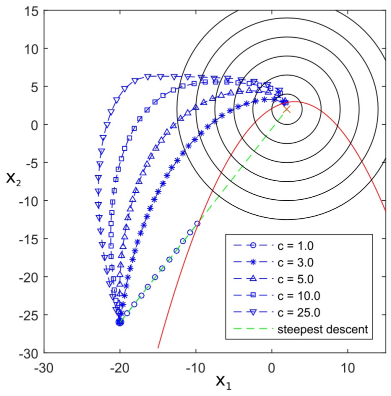

Figure 7.8: Optimization paths with different parameters c.

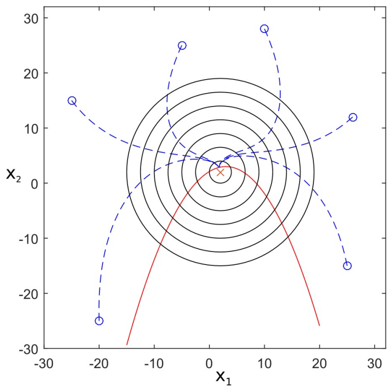

Figure 7.9: Optimization paths with parameter $c = 5.0$ and different initializations.

7.5 Modified search direction method for multiple inequality constraints

With the logarithmic barrier defined in (7.4), the present modified search direction method is generalized to solve multiple inequality constrained problems. The basic idea is to assemble the gradient information of all inequality constraint gradients into a single gradient vector. The gradient of the logarithmic barrier function reads

$$\nabla \Phi(x) = \sum_{i=1}^{m} -\frac{1}{g_i(x)} \nabla g_i(x). \tag{7.31}$$

$\nabla \Phi(x)$ is the sum of all constraint gradients scaled with the respective $-\frac{1}{g_i(x)}$, which is a positive factor. The factor $-\frac{1}{g_i(x)}$ grows quickly as the constraint function approaches zero. Intuitively, the closer the variable x gets to the boundary of a constraint, the larger its contribution to $\nabla \Phi(x)$.

Substitute $\nabla \Phi(x)$ in the sensitivity matrix (7.14), we get the sensitivity matrix for multiple constraints

$$\mathbf{m} = \begin{pmatrix} \frac{\nabla f(x)^T}{|\nabla f(x)|} \\ \frac{\nabla \Phi(x)^T}{|\nabla \Phi(x)|} \end{pmatrix}. \tag{7.32}$$

The modified search direction is then computed in the same way as described in section 7.3.

Example

Consider the following problem:

$$\min_{x_1, x_2} \quad f = (x_1 - 2)^2 + (x_2 - 2)^2, \tag{7.33}$$

which is subjected to the inequality constraints g_i, with $i = 1, ..., 5$:

$$\begin{aligned}
&g_1 = -2x_1 - x_2 + 8 \leq 0 \\
&g_2 = -\frac{1}{10}(x_1 - 3)^2 - x_2 + 3 \leq 0 \\
&g_3 = x_2 - 7 \leq 0 \\
&g_4 = x_1 - 15 \leq 0 \\
&g_5 = -x_1 - x_2 + 5.4888 \leq 0
\end{aligned} \tag{7.34}$$

As shown in figure 7.10, starting from three different initial points, the present method successfully finds the optimal solution with the parameter $c = 5.0$.

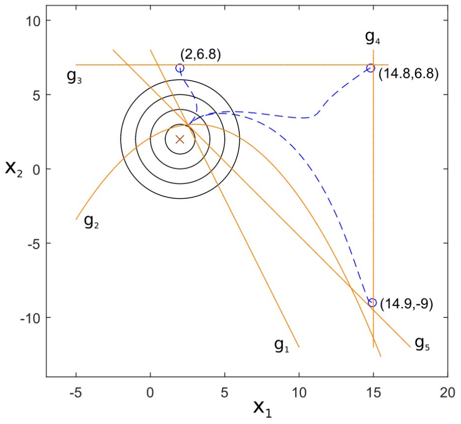

Figure 7.10: Optimization paths with three different initializations.

The dashed lines represent the different optimization paths, the solid lines visualize the five constraints, and the circles illustrate the contours of the objective function. It is also noteworthy that there is a degeneracy of the constraints at the optimum solution, and the proposed method is able to tackle this difficulty.

CHAPTER 8

DERIVATION OF THE FORMULA

We show the derivation of the present formula

$$\mathbf{s}_\zeta = -\frac{\nabla f(x)}{|\nabla f(x)|} - \zeta \frac{\nabla g(x)}{|\nabla g(x)|}, \quad 0 \le \zeta < 1 \tag{8.1}$$

from the modified search direction (7.22),

$$\mathbf{s}_c = \cos\alpha_1 \mathbf{v}_1 + c \cdot \cos\alpha_2 \mathbf{v}_2, \quad c > 0.$$

Recall the sensitivity matrix $\mathbf{m}$ at x,

$$\mathbf{m}(x) = \begin{pmatrix} \frac{\nabla f(x)^T}{|\nabla f(x)|} \\ \frac{\nabla g(x)^T}{|\nabla g(x)|} \end{pmatrix}. \tag{8.2}$$

In the following, we use $\nabla f, \nabla g$ instead of $\nabla f(x), \nabla g(x)$ to lighten the notation.

Recall the SVD for the sensitivity matrix $\mathbf{m}$

$$\mathbf{m} = \mathbf{U\Sigma V}^T = \sum_{i=1}^{2} \boldsymbol{\sigma}_i \mathbf{u}_i \mathbf{v}_i^T.$$

The columns $\mathbf{u}_i$ and $\mathbf{v}_i$ are orthonormal. Thus, we have

$$\mathbf{m}\mathbf{m}^T = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T(\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T)^T = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T(\mathbf{V}\boldsymbol{\Sigma}^T\mathbf{U}^T) = \mathbf{U}\boldsymbol{\Sigma}\boldsymbol{\Sigma}^T\mathbf{U}^T, \tag{8.3}$$

which is the diagonalization of the symmetric matrix $\mathbf{m}\mathbf{m}^T$. Therefore, it suggests computing $\mathbf{U}$ and $\boldsymbol{\Sigma}\boldsymbol{\Sigma}^T$ by applying the eigendecomposition to $\mathbf{m}\mathbf{m}^T$. Since $\boldsymbol{\Sigma}$ is a diagonal matrix, the singular values of $\mathbf{m}$ are square roots of eigenvalues of $\mathbf{m}\mathbf{m}^T$.

First, $\mathbf{m}\mathbf{m}^T$ writes

$$\mathbf{m}\mathbf{m}^T = \begin{pmatrix} \frac{\nabla f^T}{|\nabla f|} \\ \frac{\nabla g^T}{|\nabla g|} \end{pmatrix} \begin{pmatrix} \frac{\nabla f}{|\nabla f|} & \frac{\nabla g}{|\nabla g|} \end{pmatrix} = \begin{pmatrix} 1 & \cos\theta \\ \cos\theta & 1 \end{pmatrix}, \tag{8.4}$$

where θ is the angle between ∇f and ∇g,

$$\cos\theta = \frac{\nabla f^T \nabla g}{|\nabla f||\nabla g|}. \tag{8.5}$$

By the standardization paradigm presented in section 7.3.3, we can factorize the matrix $\mathbf{m}\mathbf{m}^T$ and obtain the eigenvalues

$$\begin{aligned} \lambda_1 &= 1 - \cos\theta, \\ \lambda_2 &= 1 + \cos\theta, \end{aligned} \tag{8.6}$$

and the eigenvectors, respectively,

$$\begin{aligned} \mathbf{u}_1 &= \begin{pmatrix} -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}^T, \\ \mathbf{u}_2 &= \begin{pmatrix} -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{pmatrix}^T. \end{aligned} \tag{8.7}$$

The singular values are the square roots of λ_1, λ_2,

$$\begin{aligned} \sigma_1 &= \sqrt{1 - \cos\theta}, \\ \sigma_2 &= \sqrt{1 + \cos\theta}. \end{aligned} \tag{8.8}$$

By the property of SVD, we have

$$\begin{pmatrix} \frac{\nabla f^T}{|\nabla f|} \\ \frac{\nabla g^T}{|\nabla g|} \end{pmatrix} = \begin{pmatrix} -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{pmatrix} \begin{pmatrix} \sqrt{1-\cos\theta} & 0 \\ 0 & \sqrt{1+\cos\theta} \end{pmatrix} \begin{pmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \end{pmatrix}. \tag{8.9}$$

Thus,

$$\begin{aligned}\mathbf{v}_1 &= \frac{1}{\sqrt{2-2\cos\theta}}\left(-\frac{\nabla f}{|\nabla f|}+\frac{\nabla g}{|\nabla g|}\right),\\ \mathbf{v}_2 &= \frac{1}{\sqrt{2+2\cos\theta}}\left(-\frac{\nabla f}{|\nabla f|}-\frac{\nabla g}{|\nabla g|}\right),\end{aligned} \tag{8.10}$$

Furthermore,

$$\begin{aligned}\cos\alpha_1 &= -\left\langle \frac{\nabla f}{|\nabla f|}, \mathbf{v}_1\right\rangle = \frac{\sqrt{1-\cos\theta}}{\sqrt{2}},\\ \cos\alpha_2 &= -\left\langle \frac{\nabla f}{|\nabla f|}, \mathbf{v}_2\right\rangle = \frac{\sqrt{1+\cos\theta}}{\sqrt{2}}.\end{aligned} \tag{8.11}$$

Recall the definition of the modified search direction

$$\mathbf{s}_c = \cos\alpha_1\mathbf{v}_1 + c\cdot\cos\alpha_2\mathbf{v}_2.$$

Inserting (8.10) and (8.11) into the modified search direction (7.22), we get

$$\mathbf{s}_c = -\frac{\nabla f}{|\nabla f|} - \frac{(c-1)}{2}\left(\frac{\nabla f}{|\nabla f|}+\frac{\nabla g}{|\nabla g|}\right). \tag{8.12}$$

As we are mainly interested in the direction of the vector field $\mathbf{s}_c$, we can rewrite it as

$$\mathbf{s}_\zeta = -\frac{\nabla f}{|\nabla f|} - \zeta\frac{\nabla g}{|\nabla g|},$$

where $\zeta = \frac{c-1}{c+1}$. With $c \in [1, +\infty)$, we have $\zeta \in [0, 1)$.

CHAPTER 9

GLOBAL BEHAVIOR AND CONVERGENCE

In this chapter, the asymptotic global behavior of the method is shown. The main focus is to show that the method globally finds KKT solutions. We also show that the method finds critical points of the objective function in the feasible set.

To study the theory of the method, we start with the single constrained optimization problem

$$\begin{aligned} \min_{x} \quad & f(x), \\ \text{subject to} \quad & g(x) \leq 0. \end{aligned} \tag{9.1}$$

A generalization of the results to multiple constraints is presented in chapter 11. Note, the first two sections, 9.1 and 9.2, provide basic assumptions and results for the investigations in chapters 9 - 11.

9.1 Assumptions

For problem (9.1), we make mild assumptions:

(A1) Coercive condition for the objective function $f(x)$, i.e.,

$$\lim_{x\to\infty} f(x) = +\infty;$$

(A2) $\nabla f(x) \neq 0$ in the feasible set $\Omega = \{x : g_i(x) \leq 0,\ i = 1, \dots, m\}$;

(A3) $\nabla g_i(x) \neq 0,\ i = 1, \dots, m$ in the feasible set Ω;

(A4) $f, g_1, \dots, g_m : \mathbb{R}^n \to \mathbb{R}$ are twice continuously differentiable functions.

If function $f(x)$ is *coercive* and continuous, then it has a global minimizer. We do not assume the functions to be convex. Therefore there may exists more than one local minimizer for the optimization problem. Further, if the function $f(x)$ is twice continuously differentiable, then the minimizers are among the critical points of $f(x)$. The assumptions (A2) and (A3) make sure that the present search direction vector field (8.1) is well-defined in the feasible set Ω. Note that to show different results of the method, assumptions (A1)-(A4) must not be simultaneously satisfied.

9.2 Preliminaries

Recall the search direction vector field (8.1),

$$\mathbf{s}_\zeta(x) = -\frac{\nabla f(x)}{|\nabla f(x)|} - \zeta \frac{\nabla g(x)}{|\nabla g(x)|}, \quad 0 \leq \zeta < 1.$$

We study the present method via a dynamical systems perspective, i.e.,

$$\begin{cases} \dfrac{dx}{dt} = \mathbf{s}_\zeta(x), \\ x|_{t=0} = x_0, \end{cases} \tag{9.2}$$

which is an initial value problem. Let the solution trajectory of (9.2) be $x(t; \zeta, x_0)$, which is the trajectory in the phase space that travels with time t and is dependent on the parameter ζ and the initial condition x_0. Further, we denote the maximal interval of existence of $x(t; \zeta, x_0)$ in $\mathbb{R}^n$ as T_{ζ, x_0}. Again, T_{ζ, x_0} is dependent on the parameter ζ and the initial condition x_0. We denote the set where the system (9.2) is well-defined as E.

Definition 4 (Set E)**.**

$$E := \{x : |\nabla f(x)| \neq 0, |\nabla g(x)| \neq 0, x \in \mathbb{R}^n\}. \tag{9.3}$$

9.2.1 Normalized central path condition

Recall the gradient condition for the central point (7.8), for single inequality constrained optimization problem (9.1), it writes

$$\nabla f(x(\tau)) + \lambda(\tau)\nabla g(x(\tau)) = 0, \tag{9.4}$$

where the previous path parameter t is rewritten as τ to avoid ambiguity with the time t in a dynamical system.

Normalizing the gradient $\nabla f(x)$ and $\nabla g(x)$ in (9.4), we obtain

$$\frac{\nabla f(x)}{|\nabla f(x)|} + \frac{\nabla g(x)}{|\nabla g(x)|} = 0, \tag{9.5}$$

which we call the *normalized central path condition*. Comparing (9.5) with (9.4), the central path parameter τ has vanished. The equation (9.5) characterizes the central path, which differs from (9.4), which instead characterizes a point on the central path.

Furthermore, we propose a geometric condition that characterizes the central path,

$$\cos\theta(x) = \frac{\nabla f(x)^T \nabla g(x)}{|\nabla f(x)||\nabla g(x)|} = -1, \tag{9.6}$$

where θ is the angle between $\nabla f(x)$ and $\nabla g(x)$.

Observe that $\cos\theta(x)$

1) is a continuously differentiable function of x, and

2) it reaches its own minimum -1 when x is at the central path,

we can conveniently define a neighborhood of the central path with the level set of $\cos\theta(x)$.

Definition 5 (μ-neighborhood of the central path)**.** A μ-neighborhood of a central path is defined by the function level set of $\cos\theta(x)$,

$$\Theta_\mu := \{x : g(x) \le 0, \cos\theta < -\mu\}, \mu \in [0,1]. \tag{9.7}$$

Intuitively, Θ_μ is a cone-like neighborhood around the central path. Obviously, Θ_μ shrinks to the central path as $\mu \to 1^-$.

9.2.2 Lipschitz continuity for $\mathbf{s}_\zeta$

We show regularity of the proposed search direction vector field $\mathbf{s}_\zeta$. Namely, it is Lipschitz continuous in a certain domain of interest.

First, we show that the solution trajectory $x(t;\zeta,x_0)$ always stays in a bounded set.

Lemma 2. *Suppose that assumptions (A1)(A3) and (A4) hold. Then, the solution trajectory $x(t;\zeta,x_0)$ always stays in a bounded set $\Omega_{f(x_0)}$,*

$$x(t;\zeta,x_0) \in \Omega_{f(x_0)} = \{x : f(x) \le f(x_0)\}. \tag{9.8}$$

This is to say that $x(t;\zeta,x_0)$ always stays in a set bounded by the level set of the objective function given by the initial condition, $f(x) = f(x_0)$.

Proof. To prove the Lemma, we show that the objective function is monotonic decreasing along the solution trajectory. The deformation of the objective function f along $x(t;\zeta,x_0)$ reads,

$$\begin{aligned} \frac{d}{dt}\big(f(x(t;\zeta,x_0))\big) &= \frac{df(x)}{dx}\cdot\frac{dx(t;\zeta,x_0)}{dt} \\ &= \nabla f(x)^T\left(-\frac{\nabla f(x)}{|\nabla f(x)|} - \zeta\frac{\nabla g(x)}{|\nabla g(x)|}\right) \\ &= -|\nabla f(x)|\left(\frac{\nabla f(x)^T\nabla f(x)}{|\nabla f(x)||\nabla f(x)|} + \zeta\frac{\nabla f(x)^T\nabla g(x)}{|\nabla f(x)||\nabla g(x)|}\right) \\ &= -|\nabla f(x)|\big(1+\zeta\cos\theta(x)\big). \end{aligned} \tag{9.9}$$

With $\zeta \in [0,1)$ and $\cos\theta(x) \in [-1,1]$, we have

$$1 + \zeta\cos\theta(x) > 0. \tag{9.10}$$

Therefore,

$$\frac{df(x(t;\zeta,x_0))}{dt} < 0. \tag{9.11}$$

Thus, $f(x(t;\zeta,x_0))$ is monotonic decreasing and the proof is complete. □

The intersection of the bounded set $\Omega_{f(x_0)}$ and the feasible set gives the *bounded feasible set* Ω_{x_0}.

Definition 6 (Bounded feasible set Ω_{x_0})**.** Given a feasible initialization $x_0 \in \Omega$, the bounded feasible set Ω_{x_0} is defined by

$$\Omega_{x_0} := \{x : x \in \Omega_{f(x_0)} \cap \Omega\}. \tag{9.12}$$

In Ω_{x_0}, we show that the present vector field $\mathbf{s}_\zeta$ is Lipschitz continuous.

Lemma 3 (Lipschitz continuity for $\mathbf{s}_\zeta$ in Ω_{x_0})**.** *Suppose that assumptions (A1)-(A4) hold and let the bounded feasible set Ω_{x_0} be nonempty. Then, the vector field $\mathbf{s}_\zeta(x), \zeta \in [0,1]$ is Lipschitz continuous in Ω_{x_0}, i.e., for all x, y in Ω_{x_0},*

$$|\mathbf{s}_\zeta(x) - \mathbf{s}_\zeta(y)| < L|x-y|, \tag{9.13}$$

where $L > 0$ is a Lipschitz constant.

Proof. The bounded feasible set Ω_{x_0} is compact. Thus, by assumption (A4), $\nabla f(x)$ and $\nabla g(x)$ are Lipschitz continuous in Ω_{x_0}. Further, by assumptions (A2) and (A3), $\nabla f(x)$ and $\nabla g(x)$ are bounded away from 0, i.e.,

$$|\nabla f(x)| \geq a, \ \ |\nabla g(x)| \geq b, \ \ \forall x \in \Omega_{x_0}, \tag{9.14}$$

for some positive numbers a, b. Therefore, $\frac{\nabla f(x)}{|\nabla f(x)|}$ and $\frac{\nabla g(x)}{|\nabla g(x)|}$ are Lipschitz continuous. Thus, we have a Lipschitz continuity for $\mathbf{s}_\zeta = -\frac{\nabla f(x)}{|\nabla f(x)|} - \zeta \frac{\nabla g(x)}{|\nabla g(x)|}$ in Ω_{x_0}. □

9.2.3 Deformation of the constraint function along the solution trajectory

We show basic result of the deformation of $g(x)$ along the solution trajectory $x(t;\zeta,x_0)$.

Lemma 4. *Suppose that assumptions (A1)(A3) and (A4) hold, then the constraint function g decreases along the trajectory $x(t;\zeta,x_0)$ out of the cone neighborhood Θ_ζ and increases in Θ_ζ.*

Proof. We compute the deformation of constraint function g along the solution trajectory $x(t;\zeta,x_0)$:

$$\begin{aligned}
&\frac{d}{dt}\big(g(x(t;\zeta,x_0))\big) = \frac{dg(x)}{dx}\cdot\frac{dx(t;\zeta,x_0)}{dt}\\
&=\nabla g(x)^T\left(-\frac{\nabla f(x)}{|\nabla f(x)|}-\zeta\frac{\nabla g(x)}{|\nabla g(x)|}\right)\\
&=-|\nabla g(x)|\left(\frac{\nabla g(x)^T\nabla f(x)}{|\nabla g(x)||\nabla f(x)|}+\zeta\frac{\nabla g(x)^T\nabla g(x)}{|\nabla g(x)||\nabla g(x)|}\right)\\
&=-|\nabla g(x)|(\cos\theta(x)+\zeta).
\end{aligned}\tag{9.15}$$

By the definition of the μ-neighborhood Θ_μ we get the proof directly. □

9.3 Globally finding KKT solutions

In this section, we show the main result for the global behavior of the present method in terms of first-order optimality (KKT solutions).

First, we prove that the trajectory $x(t;\zeta,x_0)$ must go out the feasible set, i.e., it reaches $g(x(t^\sharp))=0$, for any feasible initialization x_0. We further show that the time $t^\sharp$ is upper bounded by $\mathcal{O}(\frac{1}{1-\zeta})$. We then show that as $\zeta\to 1^-$, $\cos\theta(x(t^\sharp))\to -1$, indicating that $x(t^\sharp)$ is a central point. Summarizing these results, we finally show that the trajectory $x(t;\zeta,x_0)$ globally finds KKT solutions.

9.3.1 Reaching the boundary of the feasible set

We prove that the solution trajectory $x(t;\zeta,x_0)$ must reach the boundary of the feasible set, i.e., it reaches $g(x(t^\sharp))=0$, for some $t^\sharp>0$.

Theorem 7. *Suppose that assumptions (A1) - (A4) hold. Then*

(i) *the trajectory $x(t;\zeta,x_0)$ must go out of the feasible set with $\zeta\in[0,1)$;*

(ii) the minimum time in which the trajectory reaches the boundary of the feasible set is at most $\frac{C}{1-\zeta}$ with C independent of ζ.

Proof. We proof (i) by contradiction.

Suppose that $x(t;\zeta,x_0)$ stays in the feasible set Ω for any $t < T_{\zeta,x_0}$. Then, we can show that the maximal interval of existence T_{ζ,x_0} is infinite.

By assumptions (A2) and (A3), the vector field s_ζ keeps C^1 continuous in a neighborhood, since there is no critical point of $f(x)$ and $g(x)$ in the feasible set. By Lemma 3, $\mathbf{s}_\zeta(x)$ is uniformly Lipschitz continuous in Ω_{x_0}. Then, Picard's existence theorem implies that $T_{\zeta,x_0} = +\infty$.

We now show the contradiction. In the feasible set Ω, by assumption (A2), we have

$$|\nabla f(x(t;\zeta,x_0))| \geq A, \tag{9.16}$$

with some positive numbers A. On the other hand, Lemma 2 ensures that the objective function $f(x)$ is bounded, i.e.,

$$|f(x(t;\zeta,x_0)| \leq B, \quad \forall t < T_{\zeta,x_0}, \tag{9.17}$$

for some positive number B. Integrating (9.9) shows

$$\begin{aligned}\int_0^\infty -|\nabla f(x(t;\zeta,x_0))|(1+\zeta\cos\theta(x(t;\zeta,x_0)))dt \\ = f(x(T_{\zeta,x_0};\zeta,x_0)) - f(x_0) \geq -2B.\end{aligned} \tag{9.18}$$

Notice $(1+\zeta\cos\theta(x(t;\zeta,x_0))) > 0$, for any $\zeta \in [0,1)$. Therefore,

$$\int_0^\infty |\nabla f(x(t;\zeta,x_0)|dt < 2B, \tag{9.19}$$

which contradicts (9.16). This completes the proof for (i).

To proof (ii), let $T^\sharp_{\zeta,x_0}$ be the time in which the trajectory first reaches the boundary of the feasible set, then (9.16) and (9.17) hold for $0 < t < T^\sharp_{\zeta,x_0}$. Thus,

$$\int_0^{T^\sharp_{\zeta,x_0}} |\nabla f|(1+\zeta\cos\theta)dt < f(x_0) - f(x(T_{\zeta,x_0};\zeta,x_0)) \leq 2B.$$

Substitute the lower bound A for $|\nabla f(x)|$, and the lower bound -1 for $\cos\theta$, then

$$\int_0^{T^\sharp_{\zeta,x_0}} A(1-\zeta)dt < 2B. \tag{9.20}$$

Hence

$$T^\sharp_{\zeta,x_0} A(1-\zeta) < 2B, \tag{9.21}$$

which implies

$$T^\sharp_{\zeta,x_0} \leq \frac{2B}{A(1-\zeta)}. \tag{9.22}$$

Let $C = \frac{2B}{A}$, Then $T^\sharp_{\zeta,x_0} < \frac{C}{1-\zeta}$ holds. This ends the proof for (ii). □

9.3.2 Finding KKT solutions at the boundary

We show that as $\zeta \to 1^-$, then, $\cos\theta((x(t^\sharp))) \to -1$.

Theorem 8. *Suppose that assumptions (A1) - (A4) hold. Assume $\zeta \in [0,1)$ and $x_0 \in \Omega$. Let $x^\sharp_\zeta$ be the first point where the trajectory $x(t;\zeta,x_0)$ reaches the boundary of the feasible set Ω, then*

$$x^\sharp_\zeta \in \{x : g(x) = 0, \cos\theta(x) \leq -\zeta\}. \tag{9.23}$$

This is to say that the point $x^\sharp_\zeta$ belongs to the closure of the ζ-neighborhood Θ_ζ of the central path. Especially, the limit of $x^\sharp_\zeta$ as $\zeta \to 1^-$ is at the intersection of the central path and the boundary of the feasible set.

Proof. By Theorem 7, the trajectory $x(t;\zeta,x_0)$ must reach the boundary of the feasible set Ω under assumptions (A1)-(A4). Let $x^\sharp_\zeta = x(t^\sharp;\zeta,x_0)$ be the intersection point of the solution trajectory and the boundary of Ω. We have, obviously,

$$\begin{cases} g(x(t^\sharp;\zeta,x_0) = 0, \\ g(x(t;\zeta,x_0)) < 0, t < t^\sharp. \end{cases} \tag{9.24}$$

Hence

$$\frac{d}{dt}\Big(g(x(t^{\sharp};\zeta,x_0))\Big) \geq 0. \tag{9.25}$$

By (9.15), we have

$$-|\nabla g(x(t^{\sharp};\zeta,x_0))|(\cos\theta(x(t^{\sharp};\zeta,x_0))+\zeta) \geq 0. \tag{9.26}$$

Therefore,

$$\cos\theta(x(t^{\sharp};\zeta,x_0)) \leq -\zeta. \tag{9.27}$$

This ends the proof.

□

Remark 1. According to Theorem 8, as $\zeta \to 1^-$, $x(t^{\sharp})$ is a KKT solution. This is straightforward as the Lagrange multiplier $\lambda^\star$ associated with the Lagrangian function $\mathcal{L}(x,\lambda)$ for the considered problem is

$$\lambda^\star = \frac{|\nabla f|}{|\nabla g|},$$

and

$$\nabla_x \mathcal{L}(x_\zeta^{\sharp},\lambda^\star) = \nabla f + \lambda^\star \nabla g = 0.$$

Under assumptions (A2) and (A3), $\lambda^\star > 0$. With $g(x_\zeta^{\sharp}) = 0$, the strict complementarity holds. To find out whether the point $x_\zeta^{\sharp}$ is a local solution, one needs to check the second-order sufficient conditions, which is the main subject of the next chapter.

Remark 2. Based on Theorem 8, we propose an error measure $\epsilon > 0$ for the optimization solution,

$$\epsilon = 1 - \zeta. \tag{9.28}$$

According (9.23), the solution $x_\zeta^{\sharp}$ satisfies

$$x_\zeta^{\sharp} \in \{x : g(x) = 0, \cos\theta \leq -1+\epsilon\}. \tag{9.29}$$

As discussed in Remark 1, the KKT conditions are satisfied for $x_\zeta^\sharp$ as $\cos\theta \to -1$. A small value $\epsilon > 0$ seems to be a natural error measure for the present method. Additionally, ϵ is defined using the parameter ζ that determines the shape of the optimization trajectory.

With the error measure ϵ, we can interpret Theorem 7(ii) as follows. It implies that the solution time of the trajectory to find a first-order ϵ-optimal solution is at most $\mathcal{O}(1/\epsilon)$, which is the ergodic rate of convergence for first-order methods.

9.4 Global convergence to critical points of $f(x)$

In the previous section, we assume that there is no critical point of $f(x)$ in the feasible set Ω. This implies that there is no potential optimal solution in Ω. In general, this is certainly not the only case since there may exist many local solutions inside the feasible set.

In this section, we show that the present method is able to find such a critical point in the feasible set. Thus, our assumptions, naturally, cancel out (A2).

A time-reparameterized system

First, notice that the vector field $\mathbf{s}_\zeta$ is nonsmooth at critical points of $f(x)$. By Picard's existence theorem, there is a difficulty in determining the maximal interval of existence, as it may be finite[1]. To circumstance this difficulty, we use a time-reparameterized system that is orbit-equivalent to the original system (9.2).

Consider a time-reparameterized Y-system for (9.2),

$$\begin{cases} \dfrac{dy}{d\tau} = |\nabla f(y(\tau))| \mathbf{s}_\zeta(y(\tau)), \\ y(0) = x_0. \end{cases} \tag{9.30}$$

[1] If $\zeta = 0$, the present system reduces to the normalized gradient flow for unconstrained minimization, which is shown to be finite-time convergent using nontrivial nonsmooth stability analysis Cortés [22, Theorem 8].

To see the orbit-equivalence of (9.30) and (9.2), we study the reparameterization in time:

$$d\tau = \frac{1}{|\nabla f(x(t))|} dt. \tag{9.31}$$

(9.31) is well-defined in E (see Definition 4).
Then,

$$\tau = \int_0^t \frac{1}{|\nabla f(x(u))|} du = \psi(t), \tag{9.32}$$

and

$$t = \phi(\tau) = \psi^{-1}(\tau). \tag{9.33}$$

Set $y(\tau) = x(\phi(\tau)) = x(t)$, then $y(\tau)$ satisfies

$$\begin{aligned} \frac{dy(\tau)}{d\tau} &= \frac{dx(t)}{d\tau} \\ &= \frac{dx(t)}{dt}\frac{dt}{d\tau} \\ &= \frac{dx(t)}{dt}|\nabla f(x(t))| \\ &= |\nabla f(x(t))| s_\zeta(x(t)). \end{aligned} \tag{9.34}$$

Therefore,

$$\frac{dy(\tau)}{d\tau} = |\nabla f(y(\tau))| s_\zeta(y(\tau)), \tag{9.35}$$

which is the Y-system (9.30). Thus, we have

Lemma 5. *The Y-system* (9.30) *is orbit-equivalent with the X-system* (9.2) *in* E.

Global convergence to critical points of $f(x)$

With Lemma 5, we now show that the solution trajectory $x(t;\zeta,x_0)$ is globally convergent to critical points of $f(x)$.

Theorem 9. *Suppose that the assumptions (A1)(A3)(A4) hold and* $\nabla g(x) \neq 0$ *in the whole* $\mathbb{R}^n$*. Then, the trajectory converges to a connected subset of critical points of the objective function* $f(x)$ *with* $\zeta \in [0,1)$*. Especially, if the critical points of the objective function are isolated, then*

$$\lim_{t \to T^-_{\zeta,x0}} x(t;\zeta,x_0) = x_c, \quad \forall \zeta \in [0,1), \tag{9.36}$$

where $\nabla f(x_c) = 0$.

Proof. First, recall that system (9.2) is not well-defined when $\nabla f(x) = 0$. At these critical points, the vector field $\mathbf{s}_\zeta$ is nonsmooth. Therefore, the maximal interval T_{ζ,x_0} may be finite. To overcome this difficulty, we use the Y-system (9.30). By Lemma 5, the Y-system has the same orbit as (9.2) in the subset $E \subset \mathbb{R}^n$. It has a unique solution $y(\tau;\zeta,x_0)$ with an infinite existence interval by Picard's existence theorem.

For $\zeta \in [0,1)$, consider an integral along $y(\tau;\zeta,x_0)$,

$$f(y(T;\zeta,x_0)) = f(x_0) - \int_0^T |\nabla f|^2 (1+\zeta\cos\theta)(y(\tau;\zeta,x_0))d\tau.$$

Lemma 2 ensures

$$\int_0^T |\nabla f|^2 (1+\zeta\cos\theta)(y(\tau;\zeta,x_0))d\tau = f(x_0) - f(y(T;\zeta,x_0)) \le M,$$

with some positive number M independent of T. Hence

$$\int_0^\infty |\nabla f(y(\tau;\zeta,x_0))|^2 d\tau \le \frac{M}{1-\zeta} < +\infty. \tag{9.37}$$

We now show $|\nabla f(y(\tau;\zeta,x_0))|$ is uniformly Lipschitz continuous in τ. Notice that

$$\frac{d}{d\tau}|\nabla f(y(\tau;\zeta,x_0))| = \frac{d}{d\tau}\left(\sum_{j=1}^{n}\left(\frac{\partial f}{\partial y_j}\right)^2\right)^{\frac{1}{2}}$$

$$= \frac{1}{2}\left(\sum_{j=1}^{n}\left(\frac{\partial f}{\partial y_j}\right)^2\right)^{-\frac{1}{2}}\sum_{j=1}^{n}\frac{d}{d\tau}\left(\frac{\partial f}{\partial y_j}\right)^2$$

$$= \frac{1}{2}\frac{1}{|\nabla f|}\sum_{j=1}^{n}2\frac{\partial f}{\partial y_j}\frac{d}{d\tau}\left(\frac{\partial f}{\partial y_j}\right)$$

$$= \frac{1}{|\nabla f|}\sum_{j=1}^{n}\frac{\partial f}{\partial y_j}\sum_{k=1}^{n}\frac{\partial}{\partial y_k}\left(\frac{\partial f}{\partial y_j}\right)\frac{dy_k}{d\tau}$$

$$= \frac{1}{|\nabla f|}\sum_{j=1,k=1}^{n}\frac{\partial f}{\partial y_j}\frac{\partial^2 f}{\partial y_j y_k}\frac{dy_k}{d\tau}$$

$$= -\frac{1}{|\nabla f|}\sum_{j=1,k=1}^{n}\frac{\partial f}{\partial y_j}\frac{\partial^2 f}{\partial y_j\partial y_k}|\nabla f|\left(\frac{1}{|\nabla f|}\frac{\partial f}{\partial y_k}+\frac{\zeta}{|\nabla g|}\frac{\partial g}{\partial y_k}\right).$$

Notice that for $\zeta\in[0,1)$, we have

$$\left|\frac{d}{d\tau}|\nabla f(y(\tau;\zeta,x_0))|\right|$$
$$\leq \sum_{k=1,j=1}^{n}\left|\frac{\partial f}{\partial y_j}\right|\left|\frac{\partial^2 f}{\partial y_k\partial y_j}\right|\left(\frac{1}{|\nabla f|}\left|\frac{\partial f}{\partial y_k}\right|+\frac{1}{|\nabla g|}\left|\frac{\partial g}{\partial y_k}\right|\right).$$

By Cauchy-Schwarz inequality, we have

$$\left|\frac{d}{d\tau}|\nabla f(y(\tau;\zeta,x_0))|\right|$$
$$\leq |\nabla f|\sqrt{\sum_{k=1,j=1}^{n}\left|\frac{\partial^2 f}{\partial y_k\partial y_j}\right|^2}\sqrt{\sum_{k=1}^{n}\left(\frac{1}{|\nabla f|}\left|\frac{\partial f}{\partial y_k}\right|+\frac{1}{|\nabla g|}\left|\frac{\partial g}{\partial y_k}\right|\right)^2}.$$

Further, we have

$$\begin{aligned}
&\sum_{k=1}^{n}\left(\frac{1}{|\nabla f|}\left|\frac{\partial f}{\partial y_k}\right|+\frac{1}{|\nabla g|}\left|\frac{\partial g}{\partial y_k}\right|\right)^2\\
&=\sum_{k=1}^{n}\left(\frac{1}{|\nabla f|^2}\left|\frac{\partial f}{\partial y_k}\right|^2+2\frac{1}{|\nabla f||\nabla g|}\left|\frac{\partial f}{\partial y_k}\frac{\partial g}{\partial y_k}\right|+\frac{1}{|\nabla g|^2}\left|\frac{\partial g}{\partial y_k}\right|^2\right)\\
&\le\sum_{k=1}^{n}2\left(\frac{1}{|\nabla f|^2}\left|\frac{\partial f}{\partial y_k}\right|^2+\frac{1}{|\nabla g|^2}\left|\frac{\partial g}{\partial y_k}\right|^2\right)=4.
\end{aligned}$$

Therefore,

$$\left|\frac{d}{d\tau}|\nabla f(y(\tau;\zeta,x_0))|\right|\le 2|\nabla f|\sqrt{\sum_{k=1,j=1}^{n}\left|\frac{\partial^2 f}{\partial y_k\partial y_j}\right|^2}. \tag{9.38}$$

By assumption (A4) and Lemma 2, the right-hand side of the above equation is bounded. There is a constant l so that

$$\left|\frac{d}{d\tau}|\nabla f(y(\tau;\zeta,x_0))|\right|\le l,\ \forall\tau.$$

By the mean value theorem, we have a uniformly Lipschitz continuity for $|\nabla f(y(\tau))|$ in y:

$$\Big||\nabla f(y(\tau';\zeta,x_0))|-|\nabla f(y(\tau'';\zeta,x_0))|\Big|\le l|\tau'-\tau''|,\quad \forall\tau',\tau''. \tag{9.39}$$

Now, we claim

$$\lim_{\tau\to+\infty}|\nabla f(y(\tau;\zeta,x_0))|=0. \tag{9.40}$$

If (9.40) does not hold, then there is a sequence of $\tau_j\to+\infty$ and a positive constant b so that

$$|\nabla f(x(\tau_j;\zeta,x_0))|\ge b>0.$$

Choosing $\delta=\frac{b}{2l}$, then

$$\begin{aligned}
|\nabla f(x(\tau;\zeta,x_0))|\ge&|\nabla f(x(\tau_j;\zeta,x_0))|\\
&-\Big||\nabla f(x(\tau_j;\zeta,x_0))|-|\nabla f(x(\tau;\zeta,x_0))|\Big|\\
&\ge|\nabla f(x(\tau_j;\zeta,x_0))|-l|\tau_j-\tau|\ge b-\delta l=\frac{b}{2},
\end{aligned}$$

for any $|\tau_j - \tau| \leq \delta$. Therefore,

$$\begin{aligned}\int_0^{\infty} |\nabla f(x(\tau;\zeta,x_0))| d\tau &\geq \sum_{j=1}^{\infty} \int_{\tau_j-\delta}^{\tau_j+\delta} |\nabla f(x(\tau;\zeta,x_0))| d\tau \\ &\geq \sum_{j=1}^{\infty} \int_{\tau_j-\delta}^{\tau_j+\delta} \frac{b}{2} d\tau = \sum_{j=1}^{\infty} \delta b = +\infty.\end{aligned} \tag{9.41}$$

This is a contradiction to (9.37). Hence,

$$\lim_{\tau\to+\infty} |\nabla f(y(\tau;\zeta,x_0))| = 0. \tag{9.42}$$

This proves that $y(\tau;\zeta,x_0)$ approaches the connected subset of the critical points. An isolated condition makes sure that

$$\lim_{\tau\to+\infty} y(\tau;\zeta,x_0) = x_c, \tag{9.43}$$

for some critical point x_c of the objective function. Recall that the Y-system and the original X-system are orbit-equivalent in E, therefore

$$\lim_{t\to T^-_{\zeta,x0}} x(t;\zeta,x_0) = x_c, \quad \forall \zeta \in [0,1).$$

Thus, our proof is completed.

□

CHAPTER 10

LOCAL BEHAVIOR AND CONVERGENCE

In this chapter, the local convergence behavior of system (9.2) is shown. In particular, we show that the method is locally convergent to second-order optimal solutions, provided that all saddles are strict. To our best knowledge and refer to Nouiehed et al. [68], *the present method is the first known first-order method that finds a second-order optimal solution in the presence of inequality constraints.*

The present chapter is organized as follows. First, we introduce the *relative convex condition* in 2D, which is a curvature relation between the level sets of the objective and constraint function, and we show it is equivalent to the second-order sufficient condition at the solution. Then, we generalize the 2D relative convex condition to higher dimensional problems using the second fundamental form of differential geometry. As a key observation in this work, we found that this curvature relation is hidden inside the Jacobian matrix of the present search direction $\mathbf{s}_{\zeta}$. Finally, we show our main results for local convergence by studying the linearized system.

10.1 Preliminary studies in 2D

Consider the nonconvex problem

$$\begin{aligned} \min_{x_1,x_2} \quad & f(x_1,x_2) = (x_1-2)^2 + (x_2-2)^2, \\ \text{s.t.} \quad & g(x_1,x_2) = -\frac{1}{10}(x_1-3)^2 - x_2 + 3 \leq 0. \end{aligned} \tag{10.1}$$

In figure 10.1, we plot the optimization trajectory with $\zeta = 0.9999$ together with a few depicted contours of both the objective function and the constraint function. We plot three points A, B, and C on the central path, where the contours of the $f(x_1,x_2)$ and $g(x_1,x_2)$ are tangent to each other. We choose an initial condition $\mathbf{x}^0$ that is located close to point A. We observe: instead of heading to the left side of the central path, the optimization trajectory finds its way to the right side. It then follows the central path, but leaves at point C and reaches the other central path. It eventually finds the optimal solution by following second central path. The question is now, *why does the optimization trajectory head to one side (point B) of a central path over another (point A)?* The answer may lie in the difference in the curvatures of the contours of the objective and constraint function between point A and B. We observe the following fact:

Let κ_f and κ_g be the *signed curvature* of the contour curves of objective and constraint function at the central path, respectively, both with the unit normal vector $\frac{\nabla f}{|\nabla f|}$ perpendicular to the contour tangent (we refer Pressley [71, p. 35] for a formal definition of the signed curvature). Then

- at point A: $\kappa_f - \kappa_g > 0$;
- at point B: $\kappa_f - \kappa_g < 0$.

Based on this observation, we conjecture the behavior of the optimization trajectory: As $\zeta \to 1^-$, the optimization trajectory is able to approach and follow a central path, on which the central point satisfies the *relative convex condition*, which, for two-dimensional problems, is defined below.

Definition 7 (Relative Convex for 2D Problems)**.** For problem (9.1), a point $x^\star$ on the central path is said to be *relative convex* if

$$\kappa_f(x^\star) - \kappa_g(x^\star) < 0. \tag{10.2}$$

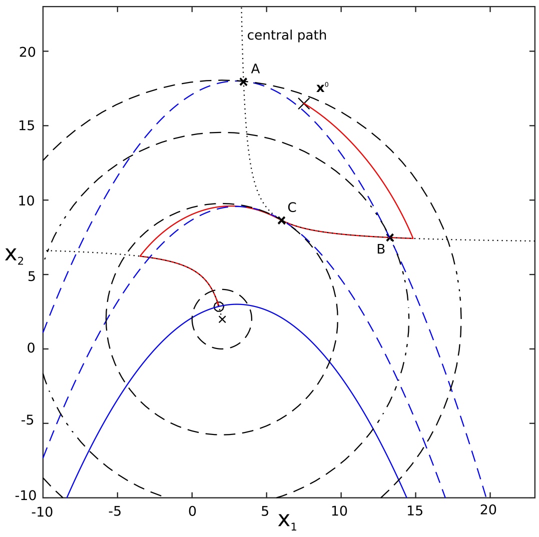

Figure 10.1: A study on the behavior of the optimization trajectory for problem (10.1). Red line is the optimization trajectory. Black dashed circles are the contours of the objective function, while the blue dashed curves show the contours of the constraint function. The dotted black lines are the central paths.

The conjectured condition can also be used to explain why the optimization trajectory leaves the central path at point C, where $\kappa_f = \kappa_g$, and heads towards another central path.

10.2 Relative curvature condition in higher dimensions

We generalize the definition of the 2D relative convex condition (10.2) to $n > 2$. This is done by studying the *second fundamental form* of the level set manifold of the objective function f and the constraint function g. In differential geometry, the second fundamental form of a surface imitates the curvature of a curve and is invariant under parameter transformation (Pressley [71, chapter 7]). In computational mechanics, it is the fundamental object used for analyzing curvatures of thin-shell structures, see, e.g., Basar et al. [9] and Kiendl [46].

10.2.1 The second fundamental form for surfaces in $\mathbb{R}^3$

We first briefly review the classical Gaussian definition of the second fundamental form for a parametric surface S in $\mathbb{R}^3$. In the work *Disquisitiones generales circa superficies curvas*, Gauss introduced a coordinate system $x = (x_1, x_2, x_3)$ for the analysis. In this particular frame, S is the *graph* of a twice continuously differentiable function, $x_3 = S(x_1, x_2)$, and the plane $x_3 = 0$ is tangent to the surface S at the origin $(0,0)$. This leads to the vanishing lower order terms in the second-order Taylor approximation of S,

$$S(x_1, x_2) \approx \frac{1}{2} x^T H_s x = L\frac{x_1^2}{2} + M x_1 x_2 + N\frac{x_2^2}{2}, \tag{10.3}$$

where H_s is the Hessian matrix of S with respect to x_1, x_2 at $(0,0)$. The second fundamental form of the surface S at the origin is defined as the quadratic form

$$\mathrm{II}_s = L d x_1^2 + 2M d x_1 d x_2 + N d x_2^2. \tag{10.4}$$

The benefit of the choice of the coordinate frame is then obvious: the second fundamental tensor coefficients L, M, N can be directly obtained by computing the Hessian matrix of the function $x_3 = S(x_1, x_2)$. In the following, we refer to such a coordinate frame as the Gaussian frame. We generalize it to higher dimensional manifolds and use it to study the relative convex condition.

10.2.2 Relative convex condition in higher dimensions

Following the Gaussian frame, we choose a coordinate system $(x_1, x_2, ..., x_n)$ with the origin be a point on the central path in the feasible set, and x_n lies in the direction of $\frac{\nabla f}{|\nabla f|}$, i.e.,

$$\nabla f(0,...,0) = [0,...,0, \frac{\partial f}{\partial x_n}]^T \neq \mathbf{0}. \tag{10.5}$$

In the following, we denote $\tilde{x} = (x_1, x_2, ..., x_{n-1})$ for brevity. By the implicit function theorem, we have a function $x_n = \phi(\tilde{x})$, which satisfies

$$f(x_1, ..., x_{n-1}, \phi(\tilde{x})) \equiv f(\underbrace{0,...0,0}_{\text{n entries}}) = \textit{constant}. \tag{10.6}$$

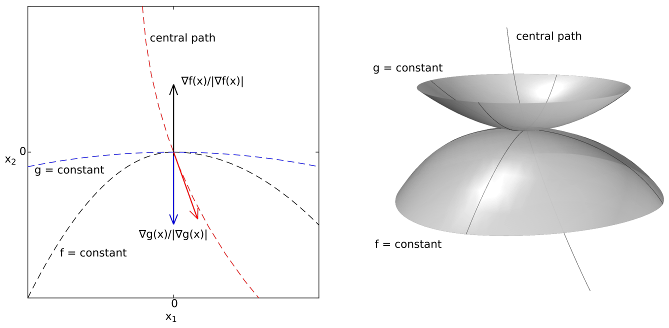

Figure 10.2: Curvature relations in 2D (left) and in higher dimensions (right).

Furthermore, we have in a neighborhood of the origin,

$$\frac{\partial \phi}{\partial x_k} = -\frac{\partial f}{\partial x_k} \Big/ \frac{\partial f}{\partial x_n}, \quad 1 \le k \le n-1. \tag{10.7}$$

By $\frac{\partial f}{\partial x_k} = 0$ we have

$$\frac{\partial \phi(\mathbf{0})}{\partial x_k} = 0, \quad 1 \le k \le n-1. \tag{10.8}$$

Let H_ϕ be the Hessian matrix of ϕ about variable $\tilde{x}$. Then, the second fundamental form for the level set of the objective function f at the origin is the quadratic form

$$\mathrm{II}_\phi = 2d\tilde{x}^T H_\phi(\mathbf{0}) d\tilde{x}. \tag{10.9}$$

Similarly, we define the function $x_n = \psi(\tilde{x})$ that satisfies the implicit function g= *constant* and has the Hessian matrix H_ψ. Then, the second fundamental form for the level set of the constraint function g at the origin is

$$\mathrm{II}_\psi = 2d\tilde{x}^T H_\psi(\mathbf{0}) d\tilde{x}. \tag{10.10}$$

Comparing (10.2), we introduce the relative convex condition in higher dimensions,

$$\mathrm{II}_\phi - \mathrm{II}_\psi < 0. \tag{10.11}$$

In the sense of positive definite matrix, we have

$$H_\phi(\mathbf{0}) \prec H_\psi(\mathbf{0}). \tag{10.12}$$

We note that (10.12) is invariant under coordinate transformation on the hypersurface (i.e., on $\tilde{x}$): the second fundamental coefficients of both manifolds are components of a covariant tensor of second-order, and the surface coordinate differentials $d\tilde{x}$ are contravariant vectors. An illustration of the relative convex condition in different dimensions is given in figure 10.2.

At $(x_1, \dots, x_{n-1}) = (0, \dots, 0)$, we have

$$\begin{aligned}\frac{\partial^2 \phi}{\partial x_k \partial x_j} &= \frac{\partial}{\partial x_k}\left(\frac{\partial \phi}{\partial x_j}\right) = \frac{\partial}{\partial x_k}\left(-\frac{\partial f}{\partial x_j} / \frac{\partial f}{\partial x_n}\right) \\ &= -\frac{\frac{\partial f}{\partial x_n}\frac{\partial^2 f}{\partial x_k \partial x_j} - \frac{\partial f}{\partial x_j}\frac{\partial^2 f}{\partial x_k \partial x_n}}{\left(\frac{\partial f}{\partial x_n}\right)^2} \\ &= -\left(\frac{\partial f}{\partial x_n}\right)^{-1}\frac{\partial^2 f}{\partial x_k \partial x_j}.\end{aligned} \tag{10.13}$$

Notice that

$$\nabla^2_{\tilde{x}} f = \left(\frac{\partial^2 f}{\partial x_k \partial x_j}\right)_{1 \le k, j \le n-1}. \tag{10.14}$$

Thus, we have

$$H_\phi(\mathbf{0}) = -\left(\frac{\partial f}{\partial x_n}\right)^{-1} \nabla^2_{\tilde{x}} f(\mathbf{0}). \tag{10.15}$$

Similarly, we have for $H_\psi(\mathbf{0})$

$$H_\psi(\mathbf{0}) = -\left(\frac{\partial g}{\partial x_n}\right)^{-1} \nabla^2_{\tilde{x}} g(\mathbf{0}). \tag{10.16}$$

Due to the positive definiteness (10.12), we have

$$-\left(\frac{\partial f}{\partial x_n}\right)^{-1} \nabla^2_{\tilde{x}} f(\mathbf{0}) \prec -\left(\frac{\partial g}{\partial x_n}\right)^{-1} \nabla^2_{\tilde{x}} g(\mathbf{0}). \tag{10.17}$$

Recall at the origin we have

$$\frac{\partial f}{\partial x_n} = |\nabla f|, \quad \frac{\partial g}{\partial x_n} = -|\nabla g|.$$

Therefore we have

$$\frac{1}{|\nabla f|}\nabla^2_{\tilde{x}} f(\mathbf{0}) + \frac{1}{|\nabla g|}\nabla^2_{\tilde{x}} g(\mathbf{0}) \succ 0. \tag{10.18}$$

Set

$$\tilde{C} = \frac{1}{|\nabla f|}\nabla^2_{\tilde{x}} f(\mathbf{0}) + \frac{1}{|\nabla g|}\nabla^2_{\tilde{x}} g(\mathbf{0}), \tag{10.19}$$

which we call the *relative curvature matrix*.

Definition 8 (High-dimensional relative convex condition)**.** A point x on the central path is called nondegenerate if $\tilde{C}(x)$ is invertible; it is *relative convex* if $\tilde{C}(x)$ is a positive definite matrix.

Remark 3. Let a point $x^\star$ satisfy the KKT conditions. The relative convex condition $\tilde{C}(x^\star) \succ 0$ is equivalent to the second-order sufficient conditions for constrained optimization. This is straightforward as the matrix

$$\tilde{H} = |\nabla f(x^\star)|\tilde{C}(x^\star) = \nabla^2_{\tilde{x}} f(\mathbf{0}) + \frac{|\nabla f|}{|\nabla g|}\nabla^2_{\tilde{x}} g(\mathbf{0}) \succ 0$$

is equivalent to the *projected Hessian* being positive definite (Nocedal et al. [67, p. 348]), with a Lagrange multiplier $\lambda^\star$ satisfying the KKT conditions and strictly complementarity holding,

$$\lambda^\star = \frac{|\nabla f|}{|\nabla g|} > 0, \; g(x^\star) = 0.$$

Here, we deduce the relative convex condition from the perspective of the difference in the curvatures of the function contours in the feasible set. It is defined on the central path and may be seen as a perturbed version of the second-order sufficient conditions.

10.3 Jacobian matrix of $\mathbf{s}_\zeta$

At the origin, the Jacobian matrix for $\frac{\nabla f}{|\nabla f|}$ reads

$$J_f^n = \left[\frac{\partial}{\partial x_j} \left(\frac{\partial_{x_i} f}{|\nabla f|} \right) \right]_{1 \le i, j \le n} . \tag{10.20}$$

Recall at the origin,

$$\frac{\nabla f}{|\nabla f|} = \frac{1}{|\nabla f|} \left[\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_{n-1}}, \frac{\partial f}{\partial x_n} \right]^T = [0, \ldots, 0, 1]^T . \tag{10.21}$$

For $1 \le i \le n-1$ and $1 \le j \le n$,

$$\begin{aligned} \frac{\partial}{\partial x_j} \left(\frac{\partial_{x_i} f}{|\nabla f|} \right) &= \frac{1}{|\nabla f|^2} \left(|\nabla f| \frac{\partial^2 f}{\partial x_i \partial x_j} - \frac{\partial f}{\partial x_i} \frac{\partial |\nabla f|}{\partial x_j} \right) \\ &= \frac{1}{|\nabla f|} \frac{\partial^2 f}{\partial x_i \partial x_j} . \end{aligned} \tag{10.22}$$

For $i = n$ and $1 \le j \le n$, $\frac{\partial_{x_i} f}{|\nabla f|}$ has the maximum value 1, resulting in its partial derivatives being zero,

$$\frac{\partial}{\partial x_j} \left(\frac{\partial_{x_n} f}{|\nabla f|} \right) = 0. \tag{10.23}$$

Thus, the Jacobian matrix J_f^n for $\frac{\nabla f}{|\nabla f|}$ at the origin writes

$$J_f^n = \left[\begin{array}{ccc|c} & & & \frac{\partial}{\partial x_n} \left(\frac{\partial_{x_1} f}{|\nabla f|} \right) \\ & J_f^{n-1} & & \vdots \\ & & & \frac{\partial}{\partial x_n} \left(\frac{\partial_{x_{n-1}} f}{|\nabla f|} \right) \\ \hline 0 & \cdots & 0 & 0 \end{array} \right], \tag{10.24}$$

where J_f^{n-1} is an $(n-1) \times (n-1)$ matrix,

$$J_f^{n-1} = \frac{1}{|\nabla f|} \left(\partial_{x_i} \partial_{x_j} f \right)_{1 \le i, j \le n-1} = \frac{1}{|\nabla f|} \nabla_{\tilde{x}}^2 f(x). \tag{10.25}$$

Similarly, the Jacobian matrix J_g^n for $\frac{\nabla g}{|\nabla g|}$ at the origin reads

$$J_g^n = \left[\begin{array}{ccc|c} & & & \frac{\partial}{\partial x_n}\left(\frac{\partial_{x_1} g}{|\nabla g|}\right) \\ & J_g^{n-1} & & \vdots \\ & & & \frac{\partial}{\partial x_n}\left(\frac{\partial_{x_{n-1}} g}{|\nabla g|}\right) \\ \hline 0 & \cdots & 0 & 0 \end{array}\right], \tag{10.26}$$

where J_g^{n-1} is an $(n-1)\times(n-1)$ matrix,

$$J_g^{n-1} = \frac{1}{|\nabla g|}\left(\partial_{x_i}\partial_{x_j} g\right)_{1\le i,j\le n-1} = \frac{1}{|\nabla g|}\nabla_{\tilde{x}}^2 g(x). \tag{10.27}$$

Therefore, the Jacobian matrix of $-\mathbf{s}_\zeta$ writes

$$\begin{aligned} \frac{\partial}{\partial x_j}\left(\frac{\nabla f}{|\nabla f|} + \zeta\frac{\nabla g}{|\nabla g|}\right)_{1\le j\le n} &= J_f^n + \zeta J_g^n \\ &= \left(J_f^n + J_g^n\right) + (\zeta-1)J_g^n, \end{aligned} \tag{10.28}$$

with

$$J_f^n + J_g^n = \left[\begin{array}{ccc|c} & & & \frac{\partial}{\partial x_n}\left(\frac{\partial_{x_1} f}{|\nabla f|} + \frac{\partial_{x_1} g}{|\nabla g|}\right) \\ & J_f^{n-1} + J_g^{n-1} & & \vdots \\ & & & \frac{\partial}{\partial x_n}\left(\frac{\partial_{x_{n-1}} f}{|\nabla f|} + \frac{\partial_{x_{n-1}} g}{|\nabla g|}\right) \\ \hline 0 & \cdots & 0 & 0 \end{array}\right]. \tag{10.29}$$

Note that at origin, the relative curvature matrix writes

$$\tilde{C} = \frac{1}{|\nabla f|}\nabla_{\tilde{x}}^2 f(\mathbf{0}) + \frac{1}{|\nabla g|}\nabla_{\tilde{x}}^2 g(\mathbf{0}) = J_f^{n-1} + J_g^{n-1},$$

which leads to our key observation: *At the central path, the relative curvature matrix $\tilde{C}$ defined in* (10.19) *is hidden inside the Jacobian matrix of the search direction vector field* $\mathbf{s}_\zeta$.

10.4 The linearized system

With (10.28) and (10.29), we can linearize $\mathbf{s}_\zeta$ at the origin by Taylor approximation.

A local coordinate frame

To avoid extensive local coordinate transformations, we choose a new coordinate frame, say $\tilde{x}$ again, which basis vectors are the eigenvectors of the matrix $\tilde{C}$. This is possible because the second fundamental form is invariant under parameter transformation in $\tilde{x}$ and that by assumption (A4), the relative curvature matrix $\tilde{C}$ is symmetric.
In the new coordinate frame, the matrix $\tilde{C}$ transforms to a diagonal matrix $\tilde{C}_\lambda$ with its eigenvalues $\lambda_i, i = 1, \ldots, n-1$,

$$\tilde{C} = \begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_{n-1} \end{bmatrix}. \tag{10.30}$$

Hence, the matrix (10.29) can be rewritten as

$$\left[\frac{\partial}{\partial x_j}\left(\frac{\nabla f}{|\nabla f|} + \frac{\nabla g}{|\nabla g|}\right)\right]_{1\le j\le n} = \left[\begin{array}{ccc|c} \lambda_1 & & & \mu_1 \\ & \ddots & & \vdots \\ & & \lambda_{n-1} & \mu_{n-1} \\ \hline 0 & \cdots & 0 & 0 \end{array}\right], \tag{10.31}$$

where $\mu_i = \frac{\partial}{\partial x_n}\left(\frac{\partial_{x_i} f}{|\nabla f|} + \frac{\partial_{x_i} g}{|\nabla g|}\right), 1 \le i \le n-1$.

The tangent of the central path

We denote the tangent line of the central path at the origin as

$$x_i = l_i x_n, \quad 1 \le i \le n-1. \tag{10.32}$$

Recall that $\left(\frac{\nabla f}{|\nabla f|}+\frac{\nabla g}{|\nabla g|}\right)$ is the normalized central path condition. Then, the directional derivatives of $\left(\frac{\nabla f}{|\nabla f|}+\frac{\nabla g}{|\nabla g|}\right)$ along the direction (10.32) are zero. Hence,

$$\lambda_i l_i + \mu_i = 0, \;\; 1 \le i \le n-1. \tag{10.33}$$

Taylor approximation for $\mathbf{s}_\zeta$

With the expressions introduced above, the Taylor expansion of $\mathbf{s}_\zeta$ writes

$$-\frac{\nabla f}{|\nabla f|} - \zeta \frac{\nabla g}{|\nabla g|} = -\left[\begin{array}{ccc|c} \lambda_1 & & & -\lambda_1 l_1 \\ & \ddots & & \vdots \\ & & \lambda_{n-1} & -\lambda_{n-1} l_{n-1} \\ \hline 0 & \cdots & 0 & 0 \end{array}\right]\begin{bmatrix} x_1 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix}$$

$$+(1-\zeta)\left[\begin{array}{ccc|c} & & & b_1 \\ & \left(a_{ij}\right)_{1\le i,j\le n-1} & & \vdots \\ & & & b_{n-1} \\ \hline 0 & \cdots & 0 & 0 \end{array}\right]\begin{bmatrix} x_1 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \zeta - 1 \end{bmatrix} + o(\rho), \tag{10.34}$$

where $a_{ij} = \frac{1}{|\nabla g|}\left(\partial_{x_i}\partial_{x_j} g\right), \; 1 \le i, j \le n-1$ and $b_i = \frac{\partial}{\partial x_n}\left(\frac{\partial_{x_i} g}{|\nabla g|}\right), \; 1 \le i \le n-1$. Summarizing the above analysis we have

Lemma 6. *If a point on the central path L is nondegenerate, then the system (9.2) may locally be considered as a perturbation of the linear system:*

$$\begin{cases} \dfrac{dx_i}{dt} = -\lambda_i x_i + \lambda_i l_i x_n + (1-\zeta)\left(\displaystyle\sum_{j=1}^{n-1} a_{ij} x_j + b_i x_n\right), \;\; 1 \le i \le n-1 \\ \dfrac{dx_n}{dt} = -(1-\zeta). \end{cases} \tag{10.35}$$

10.5 Asymptotic local convergence behavior

In the following, we use the matrix representations

$$\begin{cases} \Lambda_\zeta = \left[-\lambda_i \delta_{ij} + (1-\zeta) a_{ij}\right]_{(n-1)\times(n-1)}, \\ B_\zeta = \left[\lambda_i l_i + (1-\zeta) b_i\right]_{(n-1)\times 1}, \end{cases} \tag{10.36}$$

where δ_{ij} is the Kronecker-delta.

Lemma 7. *Given an initial point $x_0 = [x_1^0, x_2^0, ..., x_n^0]^T$, $x_n^0 > 0$, the linearized equation system (10.35) has the solution*

$$\begin{cases} \tilde{x} = e^{-\Lambda_\zeta t} V_{0,\zeta} + \Lambda_\zeta^{-1} B_\zeta x_n(t) + (1-\zeta)\Lambda_\zeta^{-2} B_\zeta, \\ x_n(t) = x_n^0 - (1-\zeta)t, \end{cases} \tag{10.37}$$

with $\tilde{x} \in \mathbb{R}^{n-1}$ being a column vector and

$$V_{0,\zeta} = \tilde{x}_0 - \Lambda_\zeta^{-1} B_\zeta x_n^0 - (1-\zeta) A_\zeta^{-2} B_\zeta. \tag{10.38}$$

Proof. Given an initial point $x_0 = \left[x_1^0, x_2^0, ..., x_n^0\right]^T$, $x_n^0 > 0$, by (10.35) we have

$$x_n(t) = x_n^0 - (1-\zeta)t. \tag{10.39}$$

Insert (10.39) into the first equation of (10.35), we get

$$\frac{dx_i}{dt} = -\lambda_i x_i + (1-\zeta)\sum_{i=1}^{n-1} a_{ij} x_j + [\lambda_i l_i + (1-\zeta) b_i][x_n^0 - (1-\zeta)t]. \tag{10.40}$$

Using the matrix representations in (10.36), we have

$$\frac{d\tilde{x}}{dt} = -\Lambda_\zeta \tilde{x} + [x_n^0 - (1-\zeta)t] B_\zeta. \tag{10.41}$$

(10.41) has the solution (see Appendix A.6)

$$\tilde{x} = e^{-\Lambda_\zeta t} V_{0,\zeta} + \Lambda_\zeta^{-1} B_\zeta x_n(t) + (1-\zeta)\Lambda_\zeta^{-2} B_\zeta, \tag{10.42}$$

where $V_{0,\zeta}$ is a vector of constants depending on the initial point x_0 and ζ,

$$V_{0,\zeta} = \tilde{x}_0 - \Lambda_\zeta^{-1} B_\zeta x_n^0 - (1-\zeta) A_\zeta^{-2} B_\zeta.$$

□

To observe the behavior of the linearized system (10.35) as $\zeta \to 1^-$, we eliminate the variable t with $t = \frac{x_n^0 - x_n(t)}{1-\zeta}$, and rewrite the solution (10.42) as

$$\tilde{x} = e^{-\Lambda_\zeta \frac{x_n^0 - x_n(t)}{1-\zeta}} V_{0,\zeta} + \Lambda_\zeta^{-1} B_\zeta x_n + (1-\zeta)\Lambda_\zeta^{-2} B_\zeta. \tag{10.43}$$

Notice that $\frac{dx_n(t)}{dt} < 0$, therefore $x_n(t) = x_n^0 - (1-\zeta)t < x_n^0$.

Theorem 10. *Let $\zeta \to 1^-$ and the point on the central path be relative convex, then the solution of the system (10.35) converges to the tangent of the central path $\mathcal{L}$ defined in (10.32).*

Proof. We study the solution (10.43).Following the observation described above, $x_n < x_n^0$. let $\zeta \to 1^-$, then $\frac{(x_n^0 - x_n)}{1-\zeta} \to +\infty$. With the relative convex condition, we have $\lambda_i > 0, i = 1, ..., n-1$. If $\zeta \to 1^-$, the eigenvalues of $\Lambda_\zeta > \frac{1}{2}\min(\lambda_i) > 0$. Resulting in $\Lambda_1 \succ 0$ by denoting $\Lambda_{\zeta \to 1^-}$ as Λ_1.
By the matrix exponential, we have

$$e^{-\Lambda_\zeta \frac{x_n^0 - x_n}{1-\zeta}} \to \mathbf{0}_{(n-1)\times(n-1)}, \text{ as } \zeta \to 1^-. \tag{10.44}$$

Denote $B_{\zeta \to 1^-}$ as B_1, the solution (10.43) converges to

$$\tilde{x} = \Lambda_1^{-1} B_1 x_n = \left[l_i x_n\right]_{(n-1)\times 1}, \tag{10.45}$$

which is the tangent of the central path $\mathcal{L}$ as is given in (10.32). □

10.6 Local convergence to local minimizers

We show that as $\zeta \to 1^-$, the present system (9.2) almost always converges to local minimizers with random initializations in a small neighborhood around a critical point, provided all saddles are strict. Notice, the convergence here means that a part of the optimization trajectory converges to a central path where a local minimizer locates.

First, we give a formal definition for the *saddle point* in the considered constrained optimization (9.1) using the *relative curvature matrix* $\tilde{C}$ defined in (10.19). Our focus is on strict saddle points that have directions where the curvature inscribed in $\tilde{C}$ is strictly negative. Referred to Definition

8, we call such a point a *strict relative saddle*. Referred to Remark 3, the projected Hessian matrix of the Lagrangian function for these points has at least one negative eigenvalue.

Definition 9 (Strict relative saddle)**.** A nondegenerate central point $x^\star \in \Omega$ is a strict relative saddle, if $\lambda_{\min}(\tilde{C}) < 0$, where $\lambda_{\min}(H)$ is the smallest eigenvalue of the matrix H.

We show local convergence behavior of the present system at a small neighborhood of a central point $x^\star$.

Theorem 11. *Suppose a point $x^\star$ on the central path is nondegenerate and strict relative saddle. Let $\zeta \to 1^-$, then the solution of system (10.35) will leave a neighborhood of the central point $x^\star$, provided the initialization $x_0 = [x_1^0, x_2^0, \cdots, x_n^0]^T$ with $x_n^0 > x_n^\star$ is uniformly random in a small neighborhood around $x^\star$.*

Proof. Let $\zeta \to 1^-$ throughout the proof. Applying the eigendecomposition for Λ_ζ, we get

$$\Lambda_\zeta = P_\zeta \mathrm{diag}(\lambda_{1,\zeta}, \lambda_{2,\zeta}, ..., \lambda_{n-1,\zeta}) P_\zeta^T, \tag{10.46}$$

where diag(·) denotes a diagonal matrix, P_ζ is an orthonormal matrix,

$$P_\zeta = \left[\delta_{ij} + p_{ij,\zeta}\right]_{(n-1)\times(n-1)},$$

and

$$\begin{cases} \lambda_{i,\zeta} = \lambda_i + o(1), \\ p_{ij,\zeta} = o(1). \end{cases} \tag{10.47}$$

By the matrix exponential, we have

$$\begin{aligned} e^{-\Lambda_\zeta t} &= P_\zeta \mathrm{diag}(e^{-t\lambda_{1,\zeta}}, ..., e^{-t\lambda_{n-1,\zeta}}) P_\zeta^T \\ &= P_\zeta \mathrm{diag}(e^{-t\lambda_{i,\zeta}}) \left[\delta_{ij} + o(1)\right]^T_{(n-1)\times(n-1)}. \end{aligned} \tag{10.48}$$

By (10.38), we have

$$\begin{aligned} V_{0,\zeta} &= \tilde{x}_0 - \Lambda_\zeta^{-1} B_\zeta x_n^0 - (1-\zeta)\Lambda_\zeta^{-2} B_\zeta \\ &= \left[x_j^0 - l_j x_n^0 + o(1)\right]_{(n-1)\times 1}. \end{aligned} \tag{10.49}$$

Left-multiplying $e^{-\Lambda_\zeta t} V_{0,\zeta}$ by P_ζ^T, we obtain

$$\begin{aligned} P_\zeta^T e^{-\Lambda_\zeta t} V_{0,\zeta} &= \mathrm{diag}(e^{-t\lambda_{i,\zeta}})\big[\delta_{ij} + o(1)\big]^T \Big[x_j^0 - l_j x_n^0 + o(1)\Big] \\ &= \mathrm{diag}(e^{-t\lambda_{i,\zeta}})\big[x_i^0 - l_i x_n^0 + o(1)\big]_{(n-1)\times 1}. \end{aligned} \tag{10.50}$$

Assume a negative eigenvalue λ_k of $\tilde{C}_\lambda$ with $1 \le k \le n-1$. By (10.42), we have

$$|\tilde{x}| = |P_\zeta^T \tilde{x}| \ge e^{-t\lambda_{k,\zeta}} |x_k^0 - l_k x_n^0 + o(1)|. \tag{10.51}$$

If $\lambda_k < 0$, then

$$\lambda_{k,\zeta} < \frac{1}{2}\lambda_k < 0.$$

If $x_k^0 - l_k x_n^0 \neq 0$, then as $t = \frac{x_n^0 - x_n}{1-\zeta} \to +\infty$, we have

$$|\tilde{x}| \ge e^{-t(\lambda_k + o(1))} |x_k^0 - l_k x_n^0 + o(1)| \ge e^{-\frac{1}{2}t\lambda_k} \frac{1}{2} |x_k^0 - l_k x_n^0| \to +\infty. \tag{10.52}$$

Thus, the present system leaves a neighborhood of the strict saddle $x^\star$. For the present system to converge to the strict saddle $x^\star$, the initial point $[x_1^0, x_2^0, \cdots, x_n^0]^T$ must satisfy

$$x_k^0 = l_k x_n^0, \tag{10.53}$$

with k as the index of the negative eigenvalue of the matrix $\tilde{C}_\lambda$. Equation (10.53) implies that the initial point x_0 must lie in an $(n-1)$–dimensional subset $\mathbb{E}_s \subset \mathbb{R}^n$. The subset $\mathbb{E}_s$ has measure zero in $\mathbb{R}^n$. If the initialization is uniformly random in a small neighborhood of $x^\star$, the probability of the initial point landing in $\mathbb{E}_s$ is zero. Obviously, the result holds if $\tilde{C}_\lambda$ has more than one negative eigenvalue, and thus the proof is complete. □

Remark 4. The local behavior of the present method shown in Theorem 11 closely resembles the behavior of the gradient descent method at a small neighborhood of a critical point as shown in section 6.4: the gradient flow avoids a strict saddle point $x^\star$ with a uniformly random initialization in a small neighborhood around $x^\star$.

CHAPTER 11

THE METHOD FOR MULTIPLE CONSTRAINTS

The present method is generalized to treat multiple inequality constrained optimization problem (7.1),

$$\begin{aligned} &\min_x && f(x), \\ &\text{subject to} && g_i(x) \leq 0, i = 1, .., m, \end{aligned}$$

where $f, g_1, ..., g_m : \mathbb{R}^n \to \mathbb{R}$ are twice continuously differentiable functions.

As presented in section 7.5, we use a logarithmic barrier function based formulation for multiple constraints. In this chapter, we formalize the method in the framework of the gradient descent akin method. We show that our theoretical results from the previous two chapters hold for this formulation.

11.1 Derivation of the formula

First, recall the logarithmic barrier function for the optimization problem (7.4),

$$\Phi(x) = -\sum_{i=1}^{m} \log(-g_i(x)).$$

As $g_i(x) \to 0^-$, $\Phi(x) \to +\infty$ and is not differentiable. In order to overcome this difficulty, we consider a level set of the barrier function,

$$L_M(\Phi(x)) \equiv \{x : \Phi(x) = M\}, \tag{11.1}$$

where $M \in \mathbb{R}^+$. The level set $L_M(\Phi(x))$ approximates and approaches the boundary of the original feasible set Ω as $M \uparrow +\infty$.

Consider a subset of the original feasible set,

$$\Omega_M = \{x : \Phi(x) \leq M\}. \tag{11.2}$$

Obviously, the boundary of Ω_M is the level set $L_M(\Phi(x))$. Denote

$$G(x) = \Phi(x) - M, \tag{11.3}$$

the optimization problem (7.1) can then be approximately formulated as

$$\begin{aligned} &\text{minimize} \quad f(x), \\ &\text{subject to} \;\; G(x) \leq 0, \end{aligned} \tag{11.4}$$

which is a single constrained optimization so that our previous analysis and results can be applied. Notice that in general $\nabla G(x) \neq 0, \forall x \in \Omega_M$ may not hold. To escape these critical points, we suggest using the gradient descent $-\nabla f(x)$. Thus, we propose a dynamical system for solving the multiple constrained optimization problem

$$\frac{dx}{dt} = \begin{cases} -\dfrac{\nabla f(x)}{|\nabla f(x)|} - \zeta \dfrac{\nabla \Phi(x)}{|\nabla \Phi(x)|}, & \text{if } \nabla \Phi(x) \neq 0; \\ -\nabla f(x), & \text{if } \nabla \Phi(x) = 0. \end{cases} \tag{11.5}$$

Note that in computational practice, reaching a critical point of $\Phi(x)$ is of rarity. The second ODE of the above system serves as a safeguard for the present method.

11.2 Theoretical results

Straightforwardly, we can obtain the following theoretical results.

Corollary 1. *Let $x_\zeta^\sharp$ be the first point where the resulting trajectory of the system (11.5) reaches the boundary of Ω_M, then*

$$x_\zeta^\sharp \in \{x : \Phi(x) = M, \cos\theta \le -\zeta\}.$$

This is to say that the point $x_\zeta^\sharp$ belongs to the closure of ζ-neighborhood of the central path. Especially, the limit of $x_\zeta^\sharp$ as $\zeta \to 1^-$ is at the intersection of the central path and the boundary of the subset Ω_M.

Proof. A similar method as used in Theorem 8 gives the proof. □

Let $\mathbf{x}\star \in \Omega_M$ be a point on the central path of the barrier function method. Set

$$\tilde{C}_\Phi = \frac{1}{|\nabla f|}\nabla_{\tilde{x}}^2 f(\mathbf{x}^\star) + \frac{1}{|\nabla \Phi|}\nabla_{\tilde{x}}^2 \Phi(\mathbf{x}^\star). \tag{11.6}$$

Definition 10. A point $\mathbf{x}^\star \in \Omega_M$ on the central path is *relative convex* if $\tilde{C}_\Phi(\mathbf{x}^\star)$ is a positive definite matrix.

Corollary 2. *Let $\zeta \to 1^-$, and the point $x^\star$ on the central path be relative convex. Then, the solution of the linearized system of (11.5) converges to the tangent line of the central path $\mathcal{L}$ at $x^\star$.*

Proof. In the subset Ω_M, $\nabla\Phi$ is Lipschitz continuous. Under the assumption (A2), we have $\nabla\Phi \neq 0$ along a central path. A uniformly continuous argument shows that $\nabla\Phi \neq 0$ in a close neighborhood of the central path. A similar method shown in section 10 gives a proof. □

To conclude, using the barrier function formulation for problem (7.1), the resulting trajectory achieves an approximated local solution that locates on the boundary of the subset Ω_M. Notice, too, that the system (11.5) does not depend on the choice of M, and Ω_M exhaust Ω, i.e., $\Omega = \cup_{M>0}\Omega_M$. This means that the resulting trajectory keeps approaching the boundary of the original feasible set Ω by crossing the boundary of any subset Ω_M

dependent on M. As a result, it eases the practical implementation of the method since no extra parameter M needs to be specified for the stopping criterion, but rather, a check for each constraint violation would be sufficient.

CHAPTER 12

COMPUTATIONAL EXAMPLES

12.1 Numerical implementation

In continuous-time, the canonical flow of the present method is defined by the differential equation

$$\frac{dx}{dt} = \mathbf{s}_\zeta(x), \tag{12.1}$$

where

$$\mathbf{s}_\zeta(x) = \begin{cases} -\dfrac{\nabla f(x)}{|\nabla f(x)|} - \zeta \dfrac{\nabla \Phi(x)}{|\nabla \Phi(x)|}, & \text{if } \nabla \Phi(x) \neq 0, \\ -\nabla f(x), & \text{if } \nabla \Phi(x) = 0. \end{cases}$$

Just as the gradient descent method can be seen as applying the Euler method for a gradient flow, the simplest numerical optimization implementation for the present flow (12.1) is the explicit forward method

$$x_{k+1} = x_k + \alpha \mathbf{s}_\zeta(x_k), \tag{12.2}$$

where $\alpha > 0$ is the step size and k is the iteration counter.

In practical applications, however, this procedure might result in poor performance. We first notice that, according to Theorem 8 and Remark 2, the accuracy of the solution increases as the parameter ζ increases. However, as $\zeta \to 1^{-}$, $|\mathbf{s}_\zeta| \downarrow 0$ at the central path. This means that the velocity of the dynamical system at a close neighborhood of the central path can be very small, if a large ζ is chosen. In cases where the optimal solution is still far away, this velocity may be unnecessarily small, and a larger step size may be chosen to accelerate the optimization progress.

Another difficulty results from the potential poor scaling of the logarithmic barrier function, which results in ill-conditioned Hessian near the boundary of the feasible set (Nocedal et al. [67, p. 500-502]). We will show test examples that suffer from this problem in the following.

To overcome these difficulties, a step size rule that considers and adapts the parameter ζ may be needed. A good practical self-adaptive ζ is expected to result in an optimization trajectory that behaves similar to the Mehrotra's practical implementation for the interior-point method (Mehrotra [59]), which is well-known for its very good performance and its difficulty in deriving a convergence theory.

Another way to improve the performance may be the implementation of a momentum method or the Nesterov Accelerated Gradient (Nesterov [66]) for the present system (12.1). Although our first implementations have shown promising performance on some test problems, we leave a systematical study together with the design of a step size rule (that considers self-adaptive ζ) for future work.

12.1.1 A time-reparameterized system

In this work, we propose a simple yet robust[1] framework that is a time-reparameterized system of (12.1):

$$\frac{dx}{dt} = \frac{\mathbf{s}_\zeta(x)}{|\mathbf{s}_\zeta(x)|}, \tag{12.3}$$

where a fixed parameter ζ is chosen. In this way, we can only obtain solutions that are $(1-\zeta)$-suboptimal solutions (refer to Theorem 8 and Remark

[1] Robustness is shown in our computational experiments.

2). Assume that the solutions exist only on the boundary of the feasible set, we can define the set of all $(1-\zeta)$-suboptimal solutions as

$$\mathcal{X}_\zeta = \{x^\sharp : x^\sharp \in \Theta_\zeta, \max(g_i(x^\sharp)) = 0\}, \tag{12.4}$$

where Θ_ζ is the ζ-neighborhood given in Definition 5. A different way of defining the suboptimal solutions is given in Skaf et al. [81] and some useful applications using suboptimal solutions are shown.

Referring to the work by Murray et al. [63], we can say that the systems (12.3) and (12.1) are topologically equivalent and solutions of (12.3) are merely arc-length reparameterizations of (12.1) solutions. As the system (12.1) may move slowly at the close neighborhood of a central path, the system (12.3) moves in the same orbit with constant speed. In Murray et al. [63], the authors show that the normalized gradient descent method escapes saddle points ''quickly''. This might be beneficial when solving nonconvex optimization problems, where the gradient of the function $\nabla f(x)$ vanishes at saddle points. In fact, this phenomenon may be similar to the present system (12.1), where $|\mathbf{s}_\zeta| \downarrow 0$ at the central path as $\zeta \to 1^-$. In the following, we show numerical experiments applying the system (12.3) with appropriate constant step sizes. Note, by using the logarithmic barrier function formulation for multiple constraints in the system (12.3), we may still suffer from potential poor scaling behavior near feasible set boundaries.

12.1.2 Numerical implementation

A sketch of the pseudocode representation for our numerical implementation is shown in Algorithm 12.1. The stopping criterion is set as any violation of the constraints since an optimal solution is found when the trajectory reaches the boundary of the feasible set (see Theorem 7(i), Theorem 8, and Remark 1). In our test experiments, all optimal solutions are nontrivially located at the boundary of the feasible set. Therefore, the proposed stopping criterion is sufficient. For problems that may have a strictly feasible solution, we suggest an additional stopping criterion akin to the gradient descent method for unconstrained optimizations. Based on our computational experience, we suggest choosing the parameter ζ in the range of $0.95 \sim 0.99$. The step size α is, however, problem-dependent, and parameter tuning is needed.

Algorithm 12.1: Constant step size and normalized $\mathbf{s}_\zeta$

```
Input: Step size $\alpha$, parameter $\zeta$, initial configuration $x_0$
$x \leftarrow x_0$
repeat
    $\nabla f \leftarrow \nabla f(x), \nabla\Phi \leftarrow \nabla\Phi(x)$
    if $\nabla\Phi \neq 0$ then
        $\mathbf{s}_\zeta \leftarrow -\frac{\nabla f}{|\nabla f|} - \zeta \frac{\nabla\Phi}{|\nabla\Phi|}$
        $x \leftarrow x + \alpha \frac{\mathbf{s}_\zeta}{|\mathbf{s}_\zeta|}$
    if $\nabla\Phi = 0$ then
        $\mathbf{s}_\zeta \leftarrow -\nabla f$
        $x \leftarrow x + \alpha \frac{\mathbf{s}_\zeta}{|\mathbf{s}_\zeta|}$
until stopping criterion is met;
Output: Optimized configuration $x$
```

12.2 Experiments with common benchmarks

We first show numerical experiments for the inequality constrained problems in the EA competition at the 2006 IEEE Congress on Evolutionary Computation (Liang et al. [52]). These benchmarks are widely used among the community of evolutionary algorithms. We choose them as test examples for three reasons. First, they are well defined constrained optimization problems and have different characteristics (Rao [72]). Second, they are nontrivial to solve with first-order methods. Third, the relatively simple formulation of the optimization problems allows us to gain deeper insight into the numerical behavior of the present method. We choose the inequality constrained optimization problems for the experimentation. The problem $G12$ is excluded because it has a feasible set consisting of 9^3 disjointed spheres. For the problem $G24$, which has a feasible set consisting of two disconnected sub-regions, we choose initial designs in the sub-region that contains the reported optimal solution.

12.2.1 Results with constant ζ and step size

We conduct numerical experiments with the parameter $\zeta = 0.98$ and tuned fixed-step sizes. All bound constraints are treated as inequalities. Random initializations, that are away from the reported optimal solution, are selected in the feasible sets. As shown in Table 12.1, apart from problem G19, we find solutions for the test problems that have an absolute error less than 2e-2 compared to the reported optima. More accurate results can be obtained when we choose shorter step sizes and larger parameter ζ.

For problem G19, the reported optimal solution $\mathbf{x}^\star = $ (1.6699e-17, 3.9637e-16, 3.9459, 1.060e-16, 3.2831, 9.9999, 1.1283e-17, 1.2026e-17, 2.5071 e-15, 2.2462e-15, 0.3708, 0.2785, 0.5238, 0.3886, 0.2982) is not achievable with a fixed-step size that has a Euclidean norm of $5e-2$.

For problems G01, G04, G06, G07, G08, G24, we find sub-optimal solutions that are close to the reported optimal designs. For problems G09, G10, G18, G19, sub-optimal solutions close to local minima are found. It appears that the ''no free lunch theorem'' (Wolpert et al. [89]) may apply to the present method implementation: while we solve some of the test problems in no more than a few hundred of iterations, a much larger number of iterations are needed for the remaining problems.

Table 12.1: Results with the parameter $\zeta = 0.98$ and tuned step sizes

Prob.	step size	iters.	f($x^\star$)	f(x_ζ)	abs. error
G01	0.002	2362	-15	-14.7215	1.86e-2
G04	0.2	136	-3.0665e+4	-3.0657e+4	2.85e-4
G06	0.002	4826	-6.9618e+3	-6.8371e+3	1.79e-2
G07	0.0027	3009	24.3062	24.7876	1.98e-2
G08	0.01	66	-9.5825e-2	-9.5063e-2	0.80e-2
G09	0.05	120	6.8063e+2	6.9238e+2	1.73e-2
G10	0.35	5319	7.0492e+3	7.1898e+3	1.99e-2
G18	0.01	257	-0.8660	-0.8546	1.32e-2
G19	0.05	294	32.6556	2.7120e+2	7.30
G24	0.02	268	-5.5080	-5.4147	1.69e-2

Still, to get more insight into the numerical behavior of the method implementation, we plot the centrality measure $\cos\theta$ over the optimization process for each test problem in figure 12.1. The dashed-lines indicate the

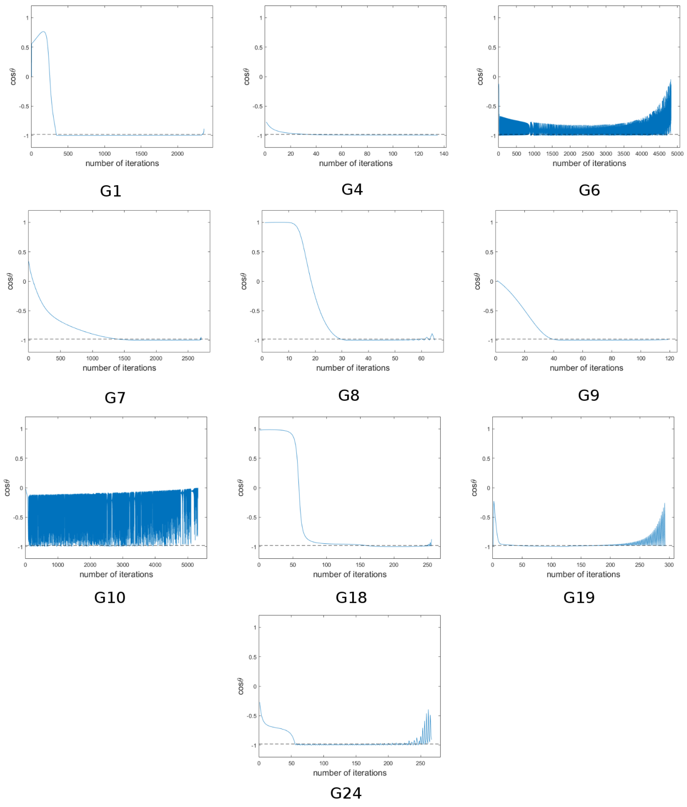

Figure 12.1: $\cos\theta$ plots.

ζ-neighborhood. The results may be summarized in three categories:

Category 1. G04, G09: optimization traverses within the ζ-neighborhood;

Category 2. G01, G07, G08, G19, G19, and G24: optimization traverses to the ζ-neighborhood but zigzags when it gets close to the solutions;

Category 3. G06 and G10: optimization zigzags around the ζ-neighborhood during the optimization process.

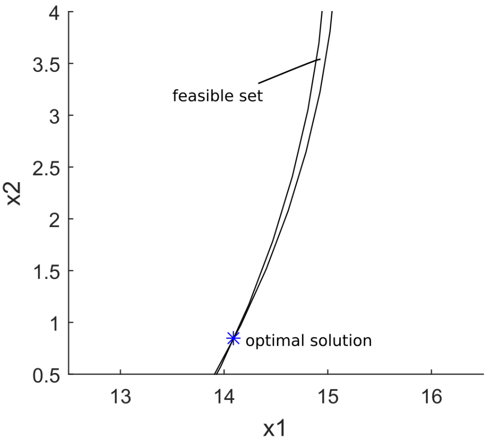

Figure 12.2: Feasible set for problem G06

The reasons for the zigzagging may be two-folds. First, it may be due to the poor scaling of the logarithmic barrier function near the boundary of the feasible set. The second reason may be the ‘‘overshooting’’: a large fixed-step size is unable to achieve a small ζ-neighborhood when close to an optimal solution. In problem G06 and G10, the central paths locate close to the boundaries of the respective feasible sets, thus resulting in zigzagging throughout the whole optimization process. In figure 12.2, we show the narrow feasible set of the test problem G06. The central path traverses close to the two boundaries of the feasible set. A similar phenomenon can be observed in problem G10. In problem G08, the zigzagging disappears when sufficiently short step sizes are chosen. It thus supports our argument of the ‘‘overshooting’’.

12.2.2 Convergence to second-order optimal solutions

In chapter 10.6, we have shown that the present method finds a second-order optimal solution asymptotically. Here, further numerical results are reported. To check the second-order optimality, the solution needs to be first-order stationary (or sufficiently accurate in terms of first-order optimality). Therefore, problem G04 and G09 of *Category 1* are chosen for the experiment, because both test cases converge smoothly and higher accuracy can be achieved by increasing the parameter ζ. We note that problem G09 is a difficult problem with a sextic polynomial objective function and a quartic polynomial constraint function. Therefore, the strict relative saddle condition may fail (see Definition 9 and compare Lee et al. [50] for unconstrained optimizations).

We conduct multi-start studies with random initializations. The second-order optimality is checked by the projected Hessian of the Lagrangian function. First, the Hessian of the Lagrangian function for the logarithmic barrier approximated problem at the solution $x^\star$ is computed,

$$\nabla_{xx}\mathcal{L}(x^\star, \lambda^\star) = \nabla_{xx} f(x^\star) + \lambda^\star \nabla_{xx}\Phi(x^\star). \tag{12.5}$$

Referred to Remark 3, the Lagrange multiplier can be computed as

$$\lambda^\star = \frac{|\nabla f(x^\star)|}{|\nabla \Phi(x^\star)|}.$$

The projected Hessian matrix can then be computed by (5.46),

$$Z^T \nabla_{xx}\mathcal{L}(x^\star, \lambda^\star) Z,$$

where the columns of the matrix Z span the null space of $\nabla\Phi(x^\star)$. The matrix Z can be computed by applying the QR factorization for $\nabla\Phi$ (Nocedal et al. [67, p. 349]). Once the projected Hessian is obtained, we can check its positive definiteness to justify the second-order optimality. We note that the feasibility of a solution is guaranteed due to our choice of the stopping criterion: the algorithm stops by any violation of the constraints.

For problem G04, we choose $\zeta = 0.99$ and a fixed-step size $\alpha = 0.1$. For problem G09, we choose $\zeta = 0.98$ and 0.99, and a fixed-step size $\alpha = 0.05$. All numerical experiments achieve second-order optimal solutions.

Table 12.2 shows results for problem G04 with random initializations. Table 12.3 and 12.4 show results for problem G09 with random initializations and different parameters ζ. Comparing Table 12.3 and 12.4, it can be observed that the method finds different local minima by varying the parameter ζ. Thus, it is evident that the parameter ζ shapes the optimization trajectories, which can result in different local solutions for a nonconvex problem.

For comparison, the reported solutions for problem G04 and G09 are presented below.

The reported solution for problem G04 is

$$x^{\sharp} = \left\{ \begin{array}{c} 78 \\ 33 \\ 29.9952569246815985 \\ 45 \\ 36.7758129057882073 \end{array} \right\}.$$

The reported optimal function value is

$$f(x^{\sharp}) = -3.066553867178332 \times 10^{4}.$$

For problem G09, the reported optimal solution is

$$x^{\sharp} = \left\{ \begin{array}{c} 2.33049935147405174 \\ 1.95137236847114592 \\ -0.477541399510615805 \\ 4.36572624923625874 \\ -0.624486959100388983 \\ 1.03813099410962173 \\ 1.5942266780671519 \end{array} \right\}.$$

The reported optimal function value is

$$f(x^{\sharp}) = 680.630057374402.$$

Table 12.2: Multi-start study for G04 with $\alpha = 0.1$ and $\zeta = 0.99$

initialization x_0	solution $x^\star$	eig(projected Hessian)	$f(x^\star)$
$\begin{Bmatrix} 101.0646 \\ 34.0158 \\ 30.9531 \\ 44.1759 \\ 29.0849 \end{Bmatrix}$	$\begin{Bmatrix} 78.1108 \\ 33.0649 \\ 30.0827 \\ 44.7616 \\ 36.6684 \end{Bmatrix}$	$\begin{Bmatrix} 2.5706 \\ 1.1772 \\ 0.4358 \\ 0.0798 \end{Bmatrix} \times 10^3$	-3.0637×10^4
$\begin{Bmatrix} 99.6638 \\ 33.2691 \\ 39.1329 \\ 41.8679 \\ 39.0573 \end{Bmatrix}$	$\begin{Bmatrix} 78.1319 \\ 33.1040 \\ 30.0972 \\ 44.6048 \\ 36.7110 \end{Bmatrix}$	$\begin{Bmatrix} 5.8769 \\ 0.7153 \\ 0.3345 \\ 0.0413 \end{Bmatrix} \times 10^3$	-3.0628×10^4
$\begin{Bmatrix} 80.0771 \\ 34.1917 \\ 35.2778 \\ 29.5124 \\ 31.4357 \end{Bmatrix}$	$\begin{Bmatrix} 78.0997 \\ 33.1122 \\ 30.0747 \\ 44.6436 \\ 36.7204 \end{Bmatrix}$	$\begin{Bmatrix} 1.3193 \\ 0.0433 \\ 0.0128 \\ 0.0020 \end{Bmatrix} \times 10^4$	-3.0637×10^4
$\begin{Bmatrix} 78.9900 \\ 44.0100 \\ 44.0100 \\ 27.9900 \\ 27.9900 \end{Bmatrix}$	$\begin{Bmatrix} 78.1017 \\ 33.1193 \\ 30.0709 \\ 44.6499 \\ 36.7175 \end{Bmatrix}$	$\begin{Bmatrix} 2.3245 \\ 0.0255 \\ 0.0071 \\ 0.0013 \end{Bmatrix} \times 10^4$	-3.0638×10^4
$\begin{Bmatrix} 100.1555 \\ 39.8080 \\ 44.0140 \\ 36.4692 \\ 33.3848 \end{Bmatrix}$	$\begin{Bmatrix} 78.1808 \\ 33.1729 \\ 30.1102 \\ 44.5612 \\ 36.6763 \end{Bmatrix}$	$\begin{Bmatrix} 8.0849 \\ 0.0971 \\ 0.0387 \\ 0.0085 \end{Bmatrix} \times 10^3$	-3.0623×10^4

Table 12.3: Multi-start study for G09 with $\alpha = 0.05$ and $\zeta = 0.98$

initialization x_0	solution $x^\star$	eig(projected Hessian)	$f(x^\star)$
$\begin{Bmatrix} -2.2608 \\ -2.7430 \\ -4.9203 \\ 0.3922 \\ -9.7739 \\ -6.8723 \\ 9.4189 \end{Bmatrix}$	$\begin{Bmatrix} 1.7553 \\ 1.9862 \\ -0.1713 \\ 4.3803 \\ -0.6154 \\ 0.8642 \\ 1.6093 \end{Bmatrix}$	$\begin{Bmatrix} 43.9185 \\ 36.2822 \\ 32.7452 \\ 13.3501 \\ 8.9740 \\ 0.8150 \end{Bmatrix}$	686.2192
$\begin{Bmatrix} 1.2675 \\ 2.7139 \\ 3.8369 \\ 0.1900 \\ -8.7219 \\ 5.8168 \\ 6.5422 \end{Bmatrix}$	$\begin{Bmatrix} 1.7751 \\ 1.9867 \\ -0.1737 \\ 4.3863 \\ -0.6154 \\ 0.8736 \\ 1.5991 \end{Bmatrix}$	$\begin{Bmatrix} 43.8858 \\ 33.5187 \\ 32.5849 \\ 13.1799 \\ 7.6776 \\ 0.5527 \end{Bmatrix}$	685.5084
$\begin{Bmatrix} 1.4067 \\ 2.7577 \\ -4.3237 \\ -0.5279 \\ -9.4532 \\ -5.4089 \\ 7.9848 \end{Bmatrix}$	$\begin{Bmatrix} 1.7518 \\ 1.9861 \\ -0.1709 \\ 4.3793 \\ -0.6154 \\ 0.8625 \\ 1.6111 \end{Bmatrix}$	$\begin{Bmatrix} 43.9256 \\ 36.7740 \\ 32.7568 \\ 13.4356 \\ 9.1647 \\ 0.8602 \end{Bmatrix}$	686.3474
$\begin{Bmatrix} 0.8577 \\ -2.7257 \\ 4.1926 \\ -0.1845 \\ -9.5364 \\ 5.4456 \\ 9.4927 \end{Bmatrix}$	$\begin{Bmatrix} 1.7655 \\ 1.9865 \\ -0.1725 \\ 4.3834 \\ -0.6154 \\ 0.8691 \\ 1.6041 \end{Bmatrix}$	$\begin{Bmatrix} 43.8995 \\ 34.8436 \\ 32.6988 \\ 13.2603 \\ 8.3590 \\ 0.6815 \end{Bmatrix}$	685.8503

Table 12.4: Multi-start study for G09 with $\alpha = 0.05$ and $\zeta = 0.99$

initialization x_0	solution $x^\star$	eig(projected Hessian)	$f(x^\star)$
$\begin{pmatrix} -2.2608 \\ -2.7430 \\ -4.9203 \\ 0.3922 \\ -9.7739 \\ -6.8723 \\ 9.4189 \end{pmatrix}$	$\begin{pmatrix} 1.9616 \\ 1.9671 \\ -0.2268 \\ 4.3939 \\ -0.6195 \\ 0.9537 \\ 1.5827 \end{pmatrix}$	$\begin{pmatrix} 45.2769 \\ 39.8254 \\ 33.8867 \\ 13.5914 \\ 10.9004 \\ 1.1475 \end{pmatrix}$	683.8082
$\begin{pmatrix} 1.2675 \\ 2.7139 \\ 3.8369 \\ 0.1900 \\ -8.7219 \\ 5.8168 \\ 6.5422 \end{pmatrix}$	$\begin{pmatrix} 1.9681 \\ 1.9670 \\ -0.2278 \\ 4.3948 \\ -0.6196 \\ 0.9562 \\ 1.5798 \end{pmatrix}$	$\begin{pmatrix} 45.2378 \\ 38.6200 \\ 33.8408 \\ 13.4789 \\ 10.5630 \\ 1.0790 \end{pmatrix}$	683.6521
$\begin{pmatrix} 1.4067 \\ 2.7577 \\ -4.3237 \\ -0.5279 \\ -9.4532 \\ -5.4089 \\ 7.9848 \end{pmatrix}$	$\begin{pmatrix} 1.9991 \\ 1.9667 \\ -0.2325 \\ 4.3991 \\ -0.6197 \\ 0.9680 \\ 1.5658 \end{pmatrix}$	$\begin{pmatrix} 45.1643 \\ 34.3464 \\ 31.2763 \\ 13.0640 \\ 7.5025 \\ 0.7178 \end{pmatrix}$	682.9375
$\begin{pmatrix} 0.8577 \\ -2.7257 \\ 4.1926 \\ -0.1845 \\ -9.5364 \\ 5.4456 \\ 9.4927 \end{pmatrix}$	$\begin{pmatrix} 1.9987 \\ 1.9667 \\ -0.2324 \\ 4.3991 \\ -0.6197 \\ 0.9678 \\ 1.5660 \end{pmatrix}$	$\begin{pmatrix} 45.1646 \\ 34.3548 \\ 31.3419 \\ 13.0673 \\ 7.5514 \\ 0.7219 \end{pmatrix}$	682.9445

12.3 Academic examples in shape optimization

The presented method has been tested by various constrained shape optimization problems. For comparison, the gradient projection method (Rosen [74]) is used. The gradient projection method uses the steepest descent direction, as there is no constraint active. After it has identified the active constraints, the gradient projection method uses a search direction obtained by solving a subproblem with quadratic cost (Bertsekas [14]).

12.3.1 Hook with strain energy constraint

The first shape optimization example is a mass minimization of a hook under static loading as shown in figure 12.3. A linear elastic analysis is carried out. The Young's Modulus $E = 2.068e11Pa$ and the Poisson's ratio $\mu = 0.29$ are assigned as material properties of the hook. The strain energy $\leq 5Nm$ is set to be the inequality constraint.

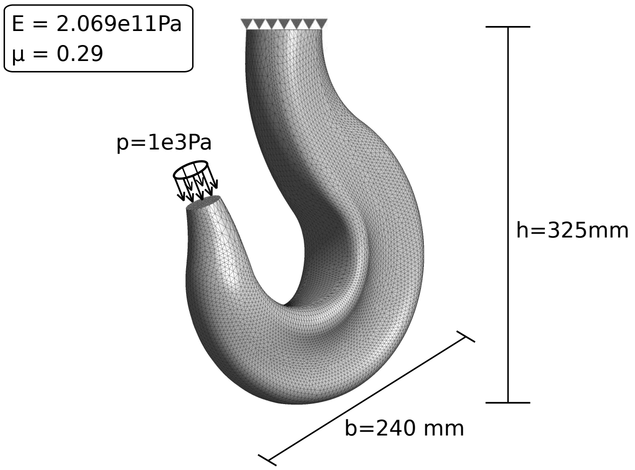

Figure 12.3: Hook under static pressure loading with fixed support at the top.

A node-based shape optimization for the hook is carried out, where we choose the surface node coordinates of the discretized hook as design

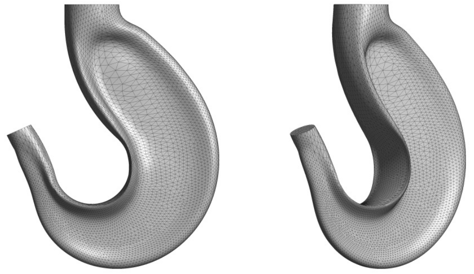

Figure 12.4: Optimal hook shape computed by the present method.

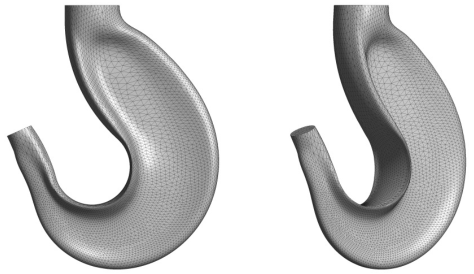

Figure 12.5: Optimal hook shape computed by the gradient projection method.

controls (number of design variables $n \approx 25000$). The nodal coordinates of the inner hook side, as well as the loading and the support area, are fixed.

As shown in figure 12.4, we obtain an optimized shape after 28 steps with the present method. The mass of the hook is reduced to 43.64% of the initial design. For comparison, the gradient projection method is used and the optimized shape is illustrated in figure 12.5. Here, the mass is reduced to 45.56% of the initial design after 29 steps. By applying the two methods, we obtain almost the same local solution. The present method achieves a slightly better design concerning objective value.

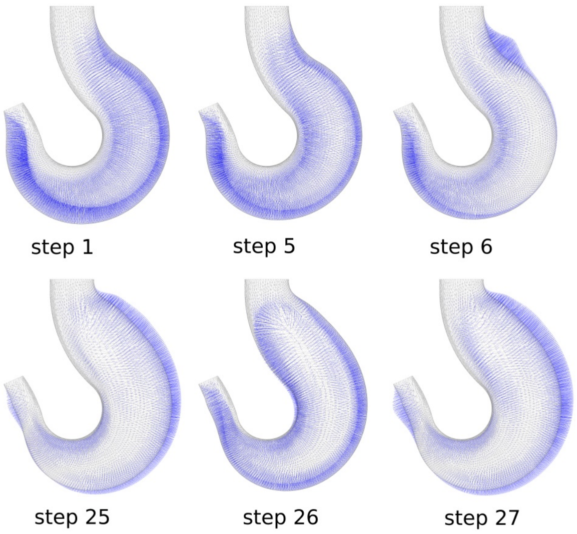

Figure 12.6: Shape updates using the gradient projection method for the hook.

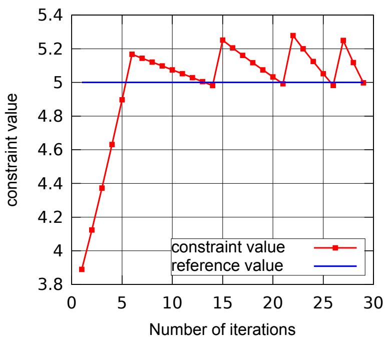

Figure 12.7: Constraint evolution plot using the gradient projection method for the hook.

Although the two methods achieve a similar local minimum, the optimization paths that are taken are different. To illustrate this difference, we depict the shape update of specific optimization steps from both methods. In figure 12.6, the shape updates of 6 individual optimization steps when using the gradient projection method are shown. The blue lines indicate the shape update direction of the design surface nodes. From the first to the fifth step, the shape update direction is pointing towards the inside of the hook. This shows that the gradient projection algorithm uses the steepest descent direction for the shape updates. At step 6, the inequality constraint for the strain energy is active, and the new search direction is computed by taking the constraint information into account. We observe there is a change in the shape update direction from step 5 to step 6. After the inequality constraint is active, the gradient projection method travels along the constraint to find a local minimum. The zig-zagging effect can be observed, for example, from step 25 to step 27 as illustrated in figure 12.6, where the direction of the shape update changes from pointing outside to pointing inside and again vice versa. This zig-zagging effect can also be observed in figure 12.7, where the constraint values are plotted.

In figure 12.8, the individual shape updates of 6 optimization steps when using the present method are shown. By taking the constraint gradient information throughout the optimization process into account, sudden changes in the direction of shape updates from subsequent optimization iterations are effectively avoided. The shape is updated significantly more smoothly compared to using the gradient projection method. The evolution of the constraint value is plotted in figure 12.9. The designs of each iteration remain in the feasible domain, and the optimization stops when the constraint becomes active at the 28-th step.

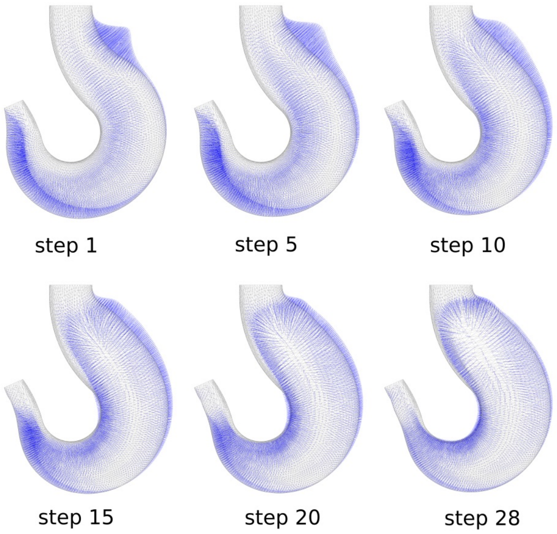

Figure 12.8: Shape updates using the present method for the hook.

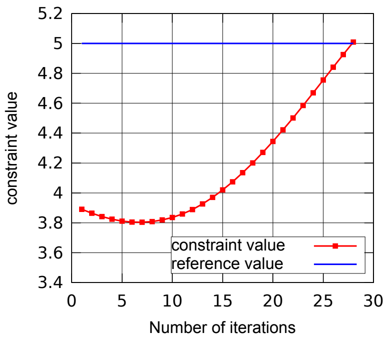

Figure 12.9: Constraint evolution plot using the present method for the hook.

12.3.2 Sphere with geometric constraints

Although it is shown that the potential poor scaling of the logarithmic barrier function may result in increasing computational effort, we want to point out that, for problems that fall on *categories 1 and 2* described in section 12.2, the logarithmic barrier function formulation can be very efficient in treating a large number of constraints. We support our argument with a shape optimization problem.

Here, we show an academic convex problem. The objective is to maximize the volume of a small design sphere, which is located inside a bigger sphere that acts as the geometric constraint. The shape geometry of both spheres are represented with finite element meshes. The optimization problem writes

$$\begin{aligned} &\text{minimize} && -V(x), \\ &\text{subject to} && g_i(x) \leq 0, \; i = 1, \dots, m, \end{aligned} \tag{12.6}$$

where $V(x)$ is the volume function, $g_i(x)$ is a point-wise defined geometric constraint for the i-th design node, m is the number of nodes of the design mesh, and $x \in \mathbb{R}^{3m}$ is the field of nodal coordinates of the design sphere mesh. The number of nodes of the small sphere (design sphere) is 19897. Thus, the total number of design variables is 59691 and the total number of constraints is 19897.

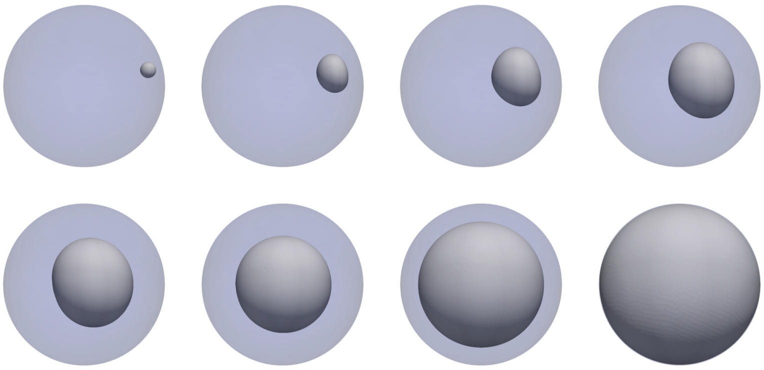

Figure 12.10: Shape evolution using the present method.

We use the logarithmic barrier function for multiple constraints and choose the parameter $\zeta = 0.95$. In figure 12.10, the shape variation process with depicted iterations is shown. Initially, the design sphere is located close to the boundary of the constraint sphere. During the shape variation process, it moves towards the center of the constraint sphere, while adapting its shape at each iteration. One could recognize easily that the central path of this optimization problem is being approached and followed until the solution is found when constraints become active.

12.4 Real-world application to shape optimization

We consider a real-world application to shape optimization, which is an obstacle problem. The present method is implemented in ShapeModule, which is a flexible solver agnostic optimization platform and provides optimization algorithms as well as shape control methods, such as Vertex Morphing (Hojjat et al. [37]). The optimization problem is to minimize the mass of a frame structure under load-displacement constraint (i.e., the displacement of every surface node is bounded). The optimization problem writes

$$\begin{aligned} &\text{minimize} && M(x), \\ &\text{subject to} && g_i(x,u) \leq 0,\ i = 1, \ldots, m, \end{aligned} \tag{12.7}$$

where $M(x)$ is the function for the mass, $g_i(x,u)$ is a point-wise formulated displacement constraint for the i-th node, m is the number of nodes of the design surface mesh, $x \in \mathbb{R}^{3m}$ is the field of nodal coordinates of the design surface mesh, and $u \in \mathbb{R}^{3m}$ is the nodal displacement field. The number of design variables is 144423, and the number of constraints is 48141. Note that for multiple constraints, we can use the logarithmic barrier function formulation as in the previous test examples. Each single displacement constraint gradient can be efficiently computed using the adjoint sensitivity analysis. In this application, we use the load-displacement sensitivity provided by the software OptiStruct to conform with a standard industrial design chain. We choose the parameter $\zeta = 0.95$.

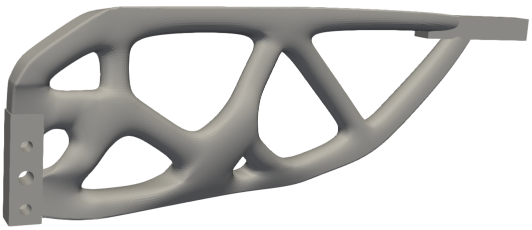

Figure 12.11: The initial frame design.

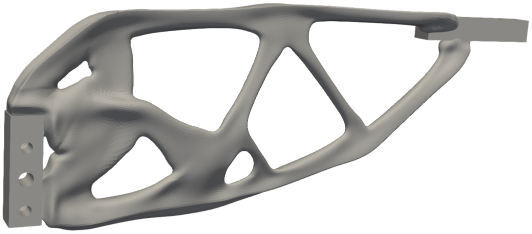

Figure 12.12: The optimized frame design.

Figure 12.11 shows the initial frame design and figure 12.12 shows the shape optimized design after 194 iterations. The mass of the structure is reduced by 41% as shown in figure 12.13. In figure 12.14, we plot the maximum constraint value $g = \max\{g_i\}$ at each iteration. In figure 12.15, we plot the centrality measure $\cos\theta$. It is shown that the optimization is able to approach and follow a central path within the ζ-neighborhood.

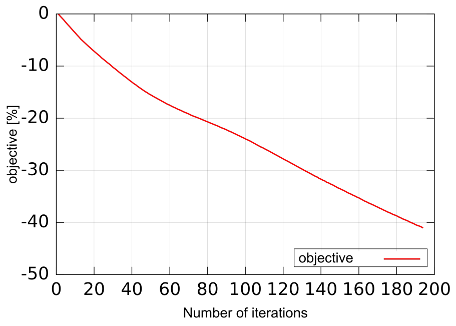

Figure 12.13: Plot of the objective function.

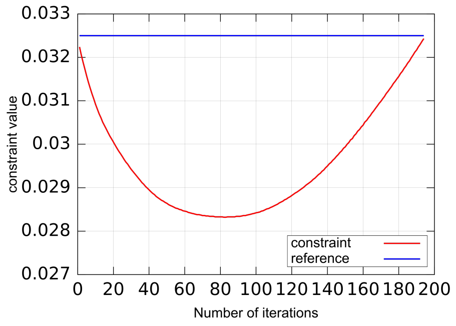

Figure 12.14: Plot of the constraint function.

Remark 5. By following a central path, an intermediate design improves not only the objective function but also the constraint function. Take the design of iteration 80 as an example: the mass is reduced by 20.5%, and the displacement is reduced by 12.1%. These designs may enrich the design options if the original problem is reformulated as a bi-objective optimization problem, in which both mass and the maximum displacement are set as objectives. The resulting intermediate designs alongside a central path are approximated Pareto solutions.

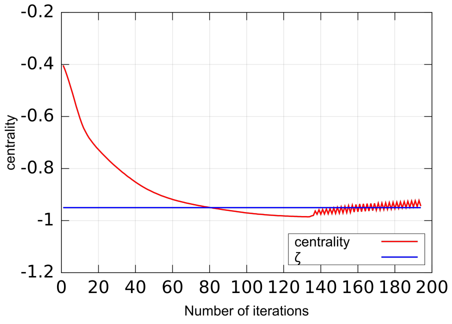

Figure 12.15: Plot of the centrality measure $\cos\theta$.

CHAPTER 13

CONCLUSIONS

In this work, a gradient descent akin method for inequality constrained optimization is presented. At each iteration, we compute a search direction using a linear combination of the negative and normalized gradient of the objective and constraint function,

$$\mathbf{s}_\zeta = -\frac{\nabla f(x)}{|\nabla f(x)|} - \zeta \frac{\nabla g(x)}{|\nabla g(x)|}, \quad \zeta \in [0,1).$$

The design of the method is inspired by the singular value decomposition. A generalization of the method to multiple constraints is presented using the logarithmic barrier function.

We use a dynamical systems approach to study the theory of the method. We show that the method

1. is globally convergent to KKT solutions;
2. is locally convergent to local minimizers, provided that all saddles are strict;

3. exhibits a convergence rate and behavior akin to that of the gradient descent method.

Various computational experiments were conducted, including common test examples and nontrivial large-scale shape optimizations, demonstrating that the present method is robust.

Finally, let me conclude by returning to the very first question of this thesis:
What makes a shape elegant?
Can I say that an optimal shape, in engineering design, is elegant?

APPENDIX A

APPENDIX A

A.1 Einstein summation convention

Einstein summation convention states: whenever an index variable appears twice in a product term, once as a subscript and once as a superscript, it implies that the summation is carried out over all the values of the index (Müller [62]). In the context of continuum mechanics, a common convention is as follows:

- Latin indices (e.g., i, k) are chosen in the 3-dimensional case (they can take values 1,2, and 3).
- Greek indices (e.e., α, β) are chosen in the 2-dimensional case (they can take the values 1,2).

Examples for Einstein summation convention are as follows:

$$a_{\alpha}{}^{\alpha} = a_{1}{}^{1} + a_{2}{}^{2},$$
$$a_{i}{}^{j} b_{j}{}^{k} = a_{i}{}^{1} b_{1}{}^{k} + a_{i}{}^{2} b_{2}{}^{k} + a_{i}{}^{3} b_{3}{}^{k}.$$

A.2 Differential operators in tensor calculus

Following (Basar et al. [10]), some of the most important differential operators in tensor calculus are presented in this section.

The *Nabla-Operator* is defined as

$$\nabla = \mathbf{g}^k \frac{\partial}{\partial \Theta^k}. \tag{A.1}$$

The *gradient* of a scalar-valued function Φ is a vector defined by

$$\operatorname{grad} \Phi = \frac{\partial \Phi}{\partial \Theta^k} \mathbf{g}^k = \Phi_{,k} \mathbf{g}^k = \nabla \Phi. \tag{A.2}$$

When applied to scalar valued functions, grad and ∇ are identical operations.

The gradient of a vector $\mathbf{u}$ is a second-order tensor

$$\operatorname{grad} \mathbf{u} = \frac{\partial \mathbf{u}}{\partial \Theta^k} \otimes \mathbf{g}^k = \mathbf{u}_{,k} \otimes \mathbf{g}^k = \mathbf{u}_{i|k} \mathbf{g}^i \otimes \mathbf{g}^k, \tag{A.3}$$

where $(\cdot)|_k$ denotes covariant derivatives with respect to Θ^k. $\mathbf{u}_{,k}$ transforms according to the covariant rule, therefore, grad $\mathbf{u}$ is invariant. The gradient increases the order of the initial tensor by one.

The product rule for gradient of $\Phi \mathbf{u}$ reads

$$\operatorname{grad} (\Phi \mathbf{u}) = (\Phi \mathbf{u})_{,k} \otimes \mathbf{g}^k = \mathbf{u} \otimes \operatorname{grad} \Phi + \Phi \operatorname{grad} \mathbf{u}. \tag{A.4}$$

The *divergence* of a vector $\mathbf{u}$ is a scalar-valued invariant defined by the rule

$$\operatorname{div} \mathbf{u} = \operatorname{grad} \mathbf{u} : \mathbf{I} = \nabla \cdot \mathbf{u}. \tag{A.5}$$

The divergence for a second-order tensor $\mathbf{A}$ reads

$$\operatorname{div} \mathbf{A} = \operatorname{grad} \mathbf{A} : \mathbf{I} = \operatorname{grad} \mathbf{A} : \mathbf{g}_k \otimes \mathbf{g}^k = A^{ij}|_j \mathbf{g}_i = \mathbf{A}_{,k} \mathbf{g}^k. \tag{A.6}$$

The divergence decreases the order of a tensor by one.

The product rule of divergence for $(\mathbf{uA})$ reads

$$\operatorname{div} (\mathbf{uA}) = \mathbf{A} : \operatorname{grad} \mathbf{u} + \mathbf{u} \cdot \operatorname{div} \mathbf{A}. \tag{A.7}$$

A.3 Differentiable function and smoothness

A function $f : V \subset \mathbb{R} \to \mathbb{R}$, defined on an open set V, is said to be differentiable at $x_0 \in V$ if the following condition is satisfied:
The derivative $f'(x_0) = \lim_{h\to 0} \frac{f(x_0+h)-f(x_0)}{h}$ exists.
A function f is said to be *continuously differentiable* or C^1 *continuous* if the derivative $f'(x)$ exists and is itself a continuous function (i.e., is of class C^0). Recursively, a C^k function has derivative of C^{k-1}.

A.4 Lipschitz continuity

A function $f : X \to Y$ is called *Lipschitz continuous* if there exists a real constant $K \geq 0$ such that, for all x_1 and x_2 in X,

$$|f(x_1) - f(x_2)|_Y \leq K|x_1 - x_2|_X,$$

where $|\cdot|_Y$ is the metric on set Y, and $|\cdot|_X$ is the metric on set X.

A.5 Implicit function theorem

Theorem A.5.1 (The implicit function theorem for $\mathbb{R}^n$)**.** *Consider a continuously differentiable function $F : \mathbb{R}^n \to \mathbb{R}$ and a point $\mathbf{x}^0 = (x_1^0, ..., x_{n-1}^0, x_n^0) \in \mathbb{R}^n$ so that $F(\mathbf{x}^0) = c$. If*

$$\frac{\partial F}{\partial x_n} \neq 0,$$

then there is a neighborhood of $\mathbf{x}^0$ so that whenever $\tilde{\mathbf{x}} = (x_1, ..., x_{n-1})$ is sufficiently close to $\tilde{\mathbf{x}}^0 = (x_1^0, ..., x_{n-1}^0)$ there is a unique x_n so that $F(\mathbf{x}) = c$. Moreover, this assignment makes x_n a continuous function of $\tilde{\mathbf{x}}$.

By the implicit function theorem, differentiating both side of $F(\tilde{\mathbf{x}}, x_n)$ with respect to $x_i, i = 1, ..., n-1$ gives

$$\frac{\partial F}{\partial x_i} + \frac{\partial F}{\partial x_n}\frac{\partial x_n}{\partial x_i} = 0. \tag{A.8}$$

Therefore,

$$\frac{\partial x_n}{\partial x_i} = -\frac{\partial F}{\partial x_i} / \frac{\partial F}{\partial x_n}. \tag{A.9}$$

A.6 Element of ordinary differential equations

The solution of the ODE

$$\frac{dx}{dt} = -px + A + Bt, \tag{A.10}$$

has the form

$$x = u(t)e^{-pt}. \tag{A.11}$$

Differentiate x we get

$$\frac{dx}{dt} = \left(u(t)e^{-pt}\right)' = u'(t)e^{-pt} - pue^{-pt}. \tag{A.12}$$

Insert (A.12) into (A.10), we get

$$\begin{aligned} u'(t)e^{-pt} - pue^{-pt} &= -pue^{-pt} + A + Bt \\ u'(t)e^{-pt} &= A + Bt. \end{aligned} \tag{A.13}$$

Or

$$u'(t) = (A + Bt)e^{pt}. \tag{A.14}$$

The function $u(t)$ can be calculated by integrating (A.14):

$$\begin{aligned} u(t) &= \int (A + Bt)e^{pt} dt \\ &= A \int e^{pt} dt + B \int t e^{pt} dt \\ &= A \int e^{pt} dt + B \int t \left(\frac{1}{p} e^{pt}\right)' dt \\ &= \frac{A}{p} e^{pt} + B \left[\frac{t}{p} e^{pt} - \int \frac{1}{p} e^{pt} dt\right] \\ &= \frac{A}{p} e^{pt} + B \left[\frac{t}{p} e^{pt} - \frac{1}{p^2} e^{pt}\right] + C, \end{aligned} \tag{A.15}$$

where C is an arbitrary constant number. Thus, the solution of the ordinary differential equation (A.10) is

$$x(t) = Ce^{-pt} + \frac{B}{p}t + \frac{A}{p} - \frac{B}{p^2}. \tag{A.16}$$

A.7 Matrix exponential

The matrix exponential is a matrix function on square matrices that is analogous to the ordinary exponential function for a single variable, i.e.,

$$f(x) = e^x. \tag{A.17}$$

Let A be an $n \times n$ real matrix. The exponential of A is defined by the convergent power series

$$e^A = \sum_{k=0}^{\infty} \frac{1}{k!} A^k = I + A + \frac{AA}{2!} + \frac{AAA}{3!} + ..., \tag{A.18}$$

which is an $n \times n$ matrix and I is the identity matrix of dimension n.

The matrix exponential has important applications in solving systems of linear, constant coefficient ordinary differential equations that mathematically model many physical, biological, and economic systems (Moler et al. [61]). Consider the following systems of ordinary differential equations

$$\frac{dx(t)}{dt} = Ax(t), \tag{A.19}$$

where $x(t)$ is a vector. Specifying the initial condition

$$x(0) = x_0, \tag{A.20}$$

the solution vector $x(t)$ of (A.19) is given by

$$x(t) = e^{tA} x_0, \tag{A.21}$$

and

$$e^{tA} = I + tA + \frac{t^2 A^2}{2!} + \cdots. \tag{A.22}$$

BIBLIOGRAPHY

[1] T. A. Albring, M. Sagebaum, and N. R. Gauger. “Efficient aerodynamic design using the discrete adjoint method in SU2.” In: *17th AIAA/ISSMO multidisciplinary analysis and optimization conference*. 2016, p. 3518.

[2] M. M. Ali and T.-L. Oliphant. “A Trajectory-Based Method for Constrained Nonlinear Optimization Problems.” In: *Journal of Optimization Theory and Applications* 177.2 (2018), pp. 479–497.

[3] G. Allaire. *Shape optimization by the homogenization method*. Vol. 146. Springer Science & Business Media, 2012.

[4] G. Allaire, F. Jouve, and A.-M. Toader. “Structural optimization using sensitivity analysis and a level-set method.” In: *Journal of computational physics* 194.1 (2004), pp. 363–393.

[5] F. Alvarez, H. Attouch, J. Bolte, and P. Redont. “A second-order gradient-like dissipative dynamical system with hessian-driven damping.: Application to optimization and mechanics.” In: *Journal de mathématiques pures et appliquées* 81.8 (2002), pp. 747–779.

[6] H. Antil, R. H. Hoppe, and C. Linsenmann. “Path-following primal-dual interior-point methods for shape optimization of stationary flow problems.” In: *Journal of Numerical Mathematics* 15.2 (2007), pp. 81–100.

[7] E. Arian and S. Ta’asan. “Analysis of the Hessian for aerodynamic optimization: Inviscid flow.” In: *Computers & fluids* 28.7 (1999), pp. 853–877.

[8] S. Arora, N. Cohen, and E. Hazan. “On the optimization of deep networks: Implicit acceleration by overparameterization.” In: *35th International Conference on Machine Learning, ICML 2018*. International Machine Learning Society (IMLS). 2018, pp. 372–389.

[9] Y. Basar and W. B. Krätzig. *Mechanik der Flächentragwerke: Theorie, Berechnungsmethoden, Anwendungsbeispiele*. Springer Fachmedien Wiesbaden, 1985.

[10] Y. Basar and D. Weichert. *Nonlinear continuum mechanics of solids: fundamental mathematical and physical concepts*. Springer Science & Business Media, 2013.

[11] D. A. Bayer and J. C. Lagarias. “The nonlinear geometry of linear programming. I. Affine and projective scaling trajectories.” In: *Transactions of the American Mathematical Society* 314.2 (1989), pp. 499–526.

[12] W. Behrman. *An efficient gradient flow method for unconstrained optimization*. stanford university PhD thesis, 1998.

[13] M. P. Bendsoe and O. Sigmund. *Topology optimization: theory, methods, and applications*. Springer Science & Business Media, 2013.

[14] D. P. Bertsekas. *Nonlinear Programming*. Athena scientific Belmont, 1999.

[15] M Bischoff and E Ramm. “Shear deformable shell elements for large strains and rotations.” In: *International Journal for Numerical Methods in Engineering* 40.23 (1997), pp. 4427–4449.

[16] C. A. Botsaris. “Differential gradient methods.” In: *Journal of Mathematical Analysis and Applications* 63.1 (1978), pp. 177–198.

[17] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[18] A. E. Bryson. *Applied optimal control: optimization, estimation and control*. CRC Press, 1975.

[19] L. Chen, K.-U. Bletzinger, A. Geiser, and R. Wüchner. “A modified search direction method for inequality constrained optimization problems using the singular-value decomposition of normalized response gradients.” In: *Structural and Multidisciplinary Optimization* (2019). DOI: 10.1007/s00158-019-02320-9.

[20] L. Chen, W. Chen, and K.-U. Bletzinger. “A gradient descent akin method for inequality constrained optimization.” In: (2019). arXiv: `1902.04040 [math.OC]`.

[21] L. Chizat and F. Bach. “On the global convergence of gradient descent for over-parameterized models using optimal transport.” In: *Advances in neural information processing systems*. 2018, pp. 3036–3046.

[22] J. Cortés. “Finite-time convergent gradient flows with applications to network consensus.” In: *Automatica* 42.11 (2006), pp. 1993–2000.

[23] A. Dener, G. K. Kenway, J. E. Hicken, and J. Martins. “Comparison of inexact-and quasi-newton algorithms for aerodynamic shape optimization.” In: *53rd AIAA Aerospace Sciences Meeting*. 2015, p. 1945.

[24] I. Diener. “Trajectory methods in global optimization.” In: *Handbook of Global optimization*. Springer, 1995, pp. 649–668.

[25] H.-B. Dörr, E. Saka, and C. Ebenbauer. “A smooth vector field for quadratic programming.” In: *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. IEEE. 2012, pp. 2515–2520.

[26] F. Feppon, G. Allaire, and C. Dapogny. “Null space gradient flows for constrained optimization with applications to shape optimization.” In: *ESAIM: Control, Optimisation and Calculus of Variations* (2020).

[27] B. Fröhlich, J. Gade, F. Geiger, M. Bischoff, and P. Eberhard. “Geometric element parameterization and parametric model order reduction in finite element based shape optimization.” In: *Computational Mechanics* 63.5 (2019), pp. 853–868.

[28] G. H. Golub and C. F. Van Loan. *Matrix computations*. Vol. 3. JHU press, 2012.

[29] C. P. Grant. “Theory of ordinary differential equations.” In: *Brigham Young University* (2014).

[30] A. O. Griewank. “Generalized descent for global optimization.” In: *Journal of optimization theory and applications* 34.1 (1981), pp. 11–39.

[31] R. T. Haftka and R. V. Grandhi. “Structural shape optimization—a survey.” In: *Computer methods in applied mechanics and engineering* 57.1 (1986), pp. 91–106.

[32] R. T. Haftka, J. Sobieszczanski-Sobieski, and S. L. Padula. “On options for interdisciplinary analysis and design optimization.” In: *Structural optimization* 4.2 (1992), pp. 65–74.

[33] J. Haslinger and R. A. Mäkinen. *Introduction to shape optimization: theory, approximation, and computation.* SIAM, 2003.

[34] U. Helmke and J. B. Moore. *Optimization and dynamical systems.* Springer Science & Business Media, 1996.

[35] J. Herskovits, G Dias, G. Santos, and C. M. Soares. “Shape structural optimization with an interior point nonlinear programming algorithm.” In: *Structural and Multidisciplinary Optimization* 20.2 (2000), pp. 107–115.

[36] J. E. Hicken and D. W. Zingg. “Aerodynamic optimization algorithm with integrated geometry parameterization and mesh movement.” In: *AIAA journal* 48.2 (2010), pp. 400–413.

[37] M. Hojjat, E. Stavropoulou, and K.-U. Bletzinger. “The vertex morphing method for node-based shape optimization.” In: *Computer Methods in Applied Mechanics and Engineering* 268 (2014), pp. 494–513. DOI: `10.1016/j.cma.2013.10.015`.

[38] R. H. Hoppe, C. Linsenmann, and H. Antil. “Adaptive path following primal dual interior point methods for shape optimization of linear and nonlinear Stokes flow problems.” In: *International Conference on Large-Scale Scientific Computing.* Springer. 2007, pp. 259–266.

[39] T. J. Hughes, M. Cohen, and M. Haroun. “Reduced and selective integration techniques in the finite element analysis of plates.” In: *Nuclear Engineering and design* 46.1 (1978), pp. 203–222.

[40] J. T. Hwang. “A modular approach to large-scale design optimization of aerospace systems.” In: *PhDT* (2015).

[41] F. Jarre, M. Kocvara, and J. Zowe. “Optimal truss design by interior-point methods.” In: *SIAM Journal on Optimization* 8.4 (1998), pp. 1084–1107. DOI: `10.1137/S1052623496297097`.

[42] M. I. Jordan. “Dynamical, symplectic and stochastic perspectives on gradient-based optimization.” In: *Proceedings of the International Congress of Mathematicians.* Vol. 1. World Scientific, 2018, pp. 525–550.

[43] G. Kennedy. “Large-scale multi-material topology optimization for additive manufacturing.” In: *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. 2015, p. 1799.

[44] G. K. Kenway, G. J. Kennedy, and J. R. Martins. “Scalable parallel approach for high-fidelity steady-state aeroelastic analysis and adjoint derivative computations.” In: *AIAA journal* 52.5 (2014), pp. 935–951.

[45] J Kiendl, R Schmidt, R Wüchner, and K.-U. Bletzinger. “Isogeometric shape optimization of shells using semi-analytical sensitivity analysis and sensitivity weighting.” In: *Computer Methods in Applied Mechanics and Engineering* 274 (2014), pp. 148–167.

[46] J. Kiendl. “Isogeometric analysis and shape optimal design of shell structures.” PhD thesis. Technische Universität München, 2011.

[47] J. Kiendl, K.-U. Bletzinger, J. Linhard, and R. Wüchner. “Isogeometric shell analysis with Kirchhoff--Love elements.” In: *Computer Methods in Applied Mechanics and Engineering* 198.49-52 (2009), pp. 3902–3914.

[48] M. Kocvara and S. Mohammed. “Primal-dual interior point multigrid method for topology optimization.” In: *SIAM Journal on Scientific Computing* 38.5 (2016), B685–B709.

[49] J. Korelc. “Automation of primal and sensitivity analysis of transient coupled problems.” In: *Computational mechanics* 44.5 (2009), pp. 631–649.

[50] J. D. Lee, M. Simchowitz, M. I. Jordan, and B. Recht. “Gradient descent only converges to minimizers.” In: *Conference on learning theory*. 2016, pp. 1246–1257.

[51] L. Lessard, B. Recht, and A. Packard. “Analysis and design of optimization algorithms via integral quadratic constraints.” In: *SIAM Journal on Optimization* 26.1 (2016), pp. 57–95.

[52] J. Liang, T. P. Runarsson, E. Mezura-Montes, M. Clerc, P. N. Suganthan, C. C. Coello, and K. Deb. “Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization.” In: *Journal of Applied Mechanics* 41.8 (2006), pp. 8–31.

[53] J. Liedmann, S. Gerke, F.-J. Barthold, and M. Brünig. “Shape optimization of the X0-specimen: theory, numerical simulation and experimental verification.” In: *Computational Mechanics* (2020), pp. 1–17.

[54] K. Linkwitz and H.-J. Schek. “Einige Bemerkungen zur Berechnung von vorgespannten Seilnetzkonstruktionen.” In: *Ingenieur-Archiv* 40.3 (1971), pp. 145–158.

[55] A. Logg, K.-A. Mardal, and G. Wells. *Automated solution of differential equations by the finite element method: The FEniCS book*. Vol. 84. Springer Science & Business Media, 2012.

[56] D. Luft, V. H. Schulz, and K. Welker. “Efficient techniques for shape optimization with variational inequalities using adjoints.” In: *SIAM Journal on Optimization* 30.3 (2020), pp. 1922–1953.

[57] B. Maar and V. Schulz. “Interior point multigrid methods for topology optimization.” In: *Structural and Multidisciplinary Optimization* 19.3 (2000), pp. 214–224.

[58] J. R. Martins and A. B. Lambe. “Multidisciplinary design optimization: a survey of architectures.” In: *AIAA journal* 51.9 (2013), pp. 2049–2075.

[59] S. Mehrotra. “On the implementation of a primal-dual interior point method.” In: *SIAM Journal on optimization* 2.4 (1992), pp. 575–601. DOI: 10.1137/0802028.

[60] J. Meiss. “Dynamical systems.” In: *Scholarpedia* (2007).

[61] C. Moler and C. Van Loan. “Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later.” In: *SIAM review* 45.1 (2003), pp. 3–49.

[62] G. Müller. *Lecture script at the Technical University of Munich: Continuum mechanics and tensor analysis*. 2013.

[63] R. Murray, B. Swenson, and S. Kar. “Revisiting normalized gradient descent: Fast evasion of saddle points.” In: *IEEE Transactions on Automatic Control* (2019).

[64] S. Nadarajah and A. Jameson. “A comparison of the continuous and discrete adjoint approach to automatic aerodynamic optimization.” In: *38th Aerospace Sciences Meeting and Exhibit*. 2000, p. 667.

[65] R. Najian Asl, I. Antonau, A. Ghantasala, W. G. Dettmer, R. Wüchner, and K.-U. Bletzinger. "A partitioned scheme for adjoint shape sensitivity analysis of fluid--structure interactions involving non-matching meshes." In: *Optimization Methods and Software* (2020), pp. 1–31.

[66] Y. E. Nesterov. "A method for solving the convex programming problem with convergence rate O (1/kˆ 2)." In: *Dokl. akad. nauk Sssr*. Vol. 269. 1983, pp. 543–547.

[67] J. Nocedal and S. Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[68] M. Nouiehed and M. Razaviyayn. "A Trust Region Method for Finding Second-Order Stationarity in Linearly Constrained Nonconvex Optimization." In: *SIAM Journal on Optimization* 30.3 (2020), pp. 2501–2529.

[69] F. Otto Film. *Frei Otto - Modeling with soap films*. Youtube. 2015. URL: `https://www.youtube.com/watch?v=-IW7o25NmeA`.

[70] R. E. Perez, P. W. Jansen, and J. R. R. A. Martins. "pyOpt: A Python-Based Object-Oriented Framework for Nonlinear Constrained Optimization." In: *Structures and Multidisciplinary Optimization* 45.1 (2012), pp. 101–118. DOI: `10.1007/s00158-011-0666-3`.

[71] A. N. Pressley. *Elementary differential geometry*. Springer Science & Business Media, 2010.

[72] R Rao. "Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems." In: *International Journal of Industrial Engineering Computations* 7.1 (2016), pp. 19–34.

[73] J. J. Reuther, A. Jameson, J. J. Alonso, M. J. Rimlinger, and D. Saunders. "Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers, part 1." In: *Journal of aircraft* 36.1 (), p. 51.

[74] J. B. Rosen. "The gradient projection method for nonlinear programming. Part I. Linear constraints." In: *Journal of the society for industrial and applied mathematics* 8.1 (1960), pp. 181–217.

[75] A. H. Sakka. *Lecture notes on maximal interval of existence*. 2010.

[76] C. Schillings, S. Schmidt, and V. Schulz. “Efficient shape optimization for certain and uncertain aerodynamic design.” In: *Computers & Fluids* 46.1 (2011), pp. 78–87.

[77] S. Schmidt. “Efficient large scale aerodynamic design based on shape calculus.” In: (2010).

[78] S. Schmidt, C. Ilic, V. Schulz, and N. R. Gauger. “Three-dimensional large-scale aerodynamic shape optimization based on shape calculus.” In: *AIAA journal* 51.11 (2013), pp. 2615–2627.

[79] V. H. Schulz, M. Siebenborn, and K. Welker. “Towards a Lagrange--Newton approach for PDE constrained shape optimization.” In: *New Trends in Shape Optimization*. Springer, 2015, pp. 229–249.

[80] J. A. Sethian. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*. Vol. 3. Cambridge university press, 1999.

[81] J. Skaf and S. Boyd. “Techniques for exploring the suboptimal set.” In: *Optimization and Engineering* 11.2 (2010), pp. 319–337.

[82] J. Snyman and L. Fatti. “A multi-start global minimization algorithm with dynamic search trajectories.” In: *Journal of Optimization Theory and Applications* 54.1 (1987), pp. 121–141.

[83] J. Sokolowski and J.-P. Zolésio. “Introduction to shape optimization.” In: *Introduction to Shape Optimization*. Springer, 1992, pp. 5–12.

[84] W. Su, S. Boyd, and E. Candes. “A differential equation for modeling Nesterov’s accelerated gradient method: Theory and insights.” In: *Advances in Neural Information Processing Systems*. 2014, pp. 2510–2518.

[85] K. Svanberg. “The method of moving asymptotes—a new method for structural optimization.” In: *International journal for numerical methods in engineering* 24.2 (1987), pp. 359–373. DOI: 10.1002/nme.1620240207.

[86] J. Tomlow. “Designing and constructing the Olympic roof (Munich 1972).” In: *International Journal of Space Structures* 31.1 (2016), pp. 62–73.

[87] S. Wang, X. Yang, and K. L. Teo. "A unified gradient flow approach to constrained nonlinear optimization problems." In: *Computational Optimization and Applications* 25.1-3 (2003), pp. 251–268.

[88] Z. Wang. "Isogeometric shape optimization for quasi-static and transient problems." In: (2016).

[89] D. H. Wolpert, W. G. Macready, et al. "No free lunch theorems for optimization." In: *IEEE transactions on evolutionary computation* 1.1 (1997), pp. 67–82.

[90] O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu. *The finite element method: its basis and fundamentals.* Elsevier, 2005.

Bisherige Titel der Schriftenreihe

Band	Titel
1	Frank Koschnick, *Geometrische Lockingeffekte bei Finiten Elementen und ein allgemeines Konzept zu ihrer Vermeidung,* 2004.
2	Natalia Camprubi, *Design and Analysis in Shape Optimization of Shells*, 2004.
3	Bernhard Thomee, *Physikalisch nichtlineare Berechnung von Stahlfaserbetonkonstruktionen*, 2005.
4	Fernaß Daoud, *Formoptimierung von Freiformschalen - Mathematische Algorithmen und Filtertechniken*, 2005.
5	Manfred Bischoff, *Models and Finite Elements for Thin-walled Structures,* 2005.
6	Alexander Hörmann, *Ermittlung optimierter Stabwerkmodelle auf Basis des Kraftflusses als Anwendung plattformunabhängiger Prozesskopplung,* 2006.
7	Roland Wüchner, *Mechanik und Numerik der Formfindung und Fluid-Struktur-Interaktion von Membrantragwerken,* 2006.
8	Florian Jurecka, *Robust Design Optimization Based on Metamodeling Techniques*, 2007.
9	Johannes Linhard, *Numerisch-mechanische Betrachtung des Entwurfsprozesses von Membrantragwerken,* 2009.
10	Alexander Kupzok, *Modeling the Interaction of Wind and Membrane Structures by Numerical Simulation,* 2009.
11	Bin Yang, *Modified Particle Swarm Optimizers and their Application to Robust Design and Structural Optimization,* 2009.

Band	**Titel**
12	Michael Fleischer, *Absicherung der virtuellen Prozesskette für Folgeoperationen in der Umformtechnik*, 2009.
13	Amphon Jrusjrungkiat, *Nonlinear Analysis of Pneumatic Membranes - From Subgrid to Interface*, 2009.
14	Alexander Michalski, *Simulation leichter Flächentragwerke in einer numerisch generierten atmosphärischen Grenzschicht*, 2010.
15	Matthias Firl, *Optimal Shape Design of Shell Structures*, 2010.
16	Thomas Gallinger, *Effiziente Algorithmen zur partitionierten Lösung stark gekoppelter Probleme der Fluid-Struktur-Wechselwirkung*, 2011.
17	Josef Kiendl, *Isogeometric Analysis and Shape Optimal Design of Shell Structures*, 2011.
18	Joseph Jordan, *Effiziente Simulation großer Mauerwerksstrukturen mit diskreten Rissmodellen*, 2011.
19	Albrecht von Boetticher, *Flexible Hangmurenbarrieren: Eine numerische Modellierung des Tragwerks, der Hangmure und der Fluid-Struktur-Interaktion*, 2012.
20	Robert Schmidt, *Trimming, Mapping, and Optimization in Isogeometric Analysis of Shell Structures*, 2013.
21	Michael Fischer, *Finite Element Based Simulation, Design and Control of Piezoelectric and Lightweight Smart Structures*, 2013.
22	Falko Hartmut Dieringer, *Numerical Methods for the Design and Analysis for Tensile Structures*, 2014.
23	Rupert Fisch, *Code Verification of Partitioned FSI Environments for Lightweight Structures*, 2014.
24	Stefan Sicklinger, *Stabilized Co-Simulation of Coupled Problems Including Fields and Signals*, 2014.

Band	**Titel**
25	Madjid Hojjat, *Node-based parametrization for shape optimal design*, 2015.
26	Ute Israel, *Optimierung in der Fluid-Struktur-Interaktion - Sensitivitätsanalyse für die Formoptimierung auf Grundlage des partitionierten Verfahrens*, 2015.
27	Electra Stavropoulou, *Sensitivity analysis and regularization for shape optimization of coupled problems*, 2015.
28	Daniel Markus, *Numerical and Experimental Modeling for Shape Optimization of Offshore Structures*, 2015.
29	Pablo Suárez, *Design Process for the Shape Optimization of Pressurized Bulkheads as Components of Aircraft Structures*, 2015.
30	Armin Widhammer, *Variation of Reference Strategy - Generation of Optimized Cutting Patterns for Textile Fabrics*, 2015.
31	Helmut Masching, *Parameter Free Optimization of Shape Adaptive Shell Structures*, 2016.
32	Hao Zhang, *A General Approach for Solving Inverse Problems in Geophysical Systems by Applying Finite Element Method and Metamodel Techniques*, 2016.
33	Tianyang Wang, *Development of Co-Simulation Environment and Mapping Algorithms*, 2016.
34	Michael Breitenberger, *CAD-integrated Design and Analysis of Shell Structures*, 2016.
35	Önay Can, *Functional Adaptation with Hyperkinematics using Natural Element Method: Application for Articular Cartilage*, 2016.
36	Benedikt Philipp, *Methodological Treatment of Non-linear Structural Behavior in the Design, Analysis and Verification of Lightweight Structures*, 2017.
37	Michael Andre, *Aeroelastic Modeling and Simulation for the Assessment of Wind Effects on a Parabolic Trough Solar Collector*, 2018.

Band	Titel
38	Andreas Apostolatos, *Isogeometric Analysis of Thin-Walled Structures on Multipatch Surfaces in Fluid-Structure Interaction*, 2018.
39	Altuğ Emiroğlu, *Multiphysics Simulation and CAD-Integrated Shape Optimization in Fluid-Structure Interaction*, 2019.
40	Mehran Saeedi, *Multi-Fidelity Aeroelastic Analysis of Flexible Membrane Wind Turbine Blades*, 2017.
41	Reza Najian Asl, *Shape Optimization and Sensitivity Analysis of Fluids, Structures, and their Interaction Using Vertex Morphing Parametrization*, 2019
42	Ahmed Abodonya, *Verification Methodology for Computational Wind Engineering Presidction of Wind Loads on Structures*, 2020.
43	Anna Maria Bauer, *CAD-Integrated Isogeometric Analysis and Design of Lightweight Structures*, 2020.
44	Andreas Winterstein, *Modeling and Simulation of Wind-Structure Interaction of Slender Civil Engineering Structures Including Vibration Systems*, 2020.
45	Fran-Josef Ertl, *Vertex Morphing for Constrained Shape Optimization of Three-dimensional Solid Structures*, 2020.
46	Daniel Baumgärtner, *On the Grid-based Shape Optimization of Structures with Internal Flow and the Feedback of Shape Changes into a CAD Model*, 2020.
47	Mohamed Khalil, *Combining Physiscs-based Models and Machine Learning for an Enhanced Structural Health Monitoring*