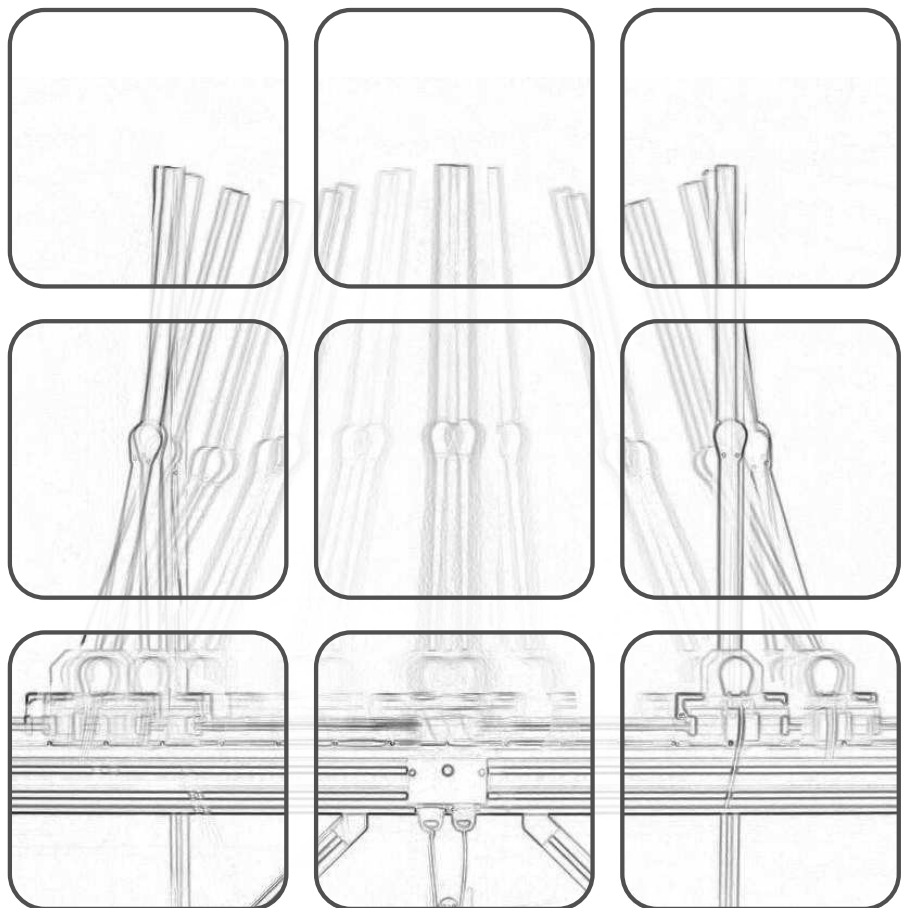


Vorlesung und Übung
WS 2026/2027

Andreas STEINBÖCK

OPTIMIERUNG



Optimierung

Vorlesung und Übung
WS 2026/2027

Andreas STEINBÖCK

TU Wien
Institut für Automatisierungs- und Regelungstechnik
Gruppe für komplexe dynamische Systeme

Gußhausstraße 27–29
1040 Wien
Telefon: +43 1 58801 – 37615
Internet: <https://www.acin.tuwien.ac.at>

© Institut für Automatisierungs- und Regelungstechnik, TU Wien

Inhaltsverzeichnis

1	Einleitung	2
1.1	Statische Optimierungsprobleme	2
1.1.1	Mathematische Formulierung	2
1.1.2	Beispiele	4
1.2	Dynamische Optimierungsprobleme	7
1.2.1	Mathematische Formulierung	7
1.2.2	Beispiele	8
1.3	Mathematische Grundlagen	12
1.3.1	Infimum und Minimum	13
1.3.2	Existenz von Minima	14
1.3.3	Gradient und Hessematrix	15
1.3.4	Berechnung von Ableitungen	17
1.3.5	Satz über implizite Funktionen	21
1.3.6	Konvexität	22
1.3.6.1	Konvexe Mengen	22
1.3.6.2	Konvexe Funktionen	23
1.4	Literatur	26
2	Statische Optimierung: Unbeschränkter Fall	28
2.1	Optimalitätsbedingungen	28
2.2	Rechnergestützte Minimierungsverfahren: Grundlagen	31
2.2.1	Iterative Abstiegsverfahren	31
2.2.2	Konvergenzverhalten	32
2.2.3	Abbruchkriterien	34
2.3	LinienSuchverfahren	35
2.3.1	Wahl der Schrittweite	35
2.3.1.1	Intervallschachtelungsverfahren („Goldener Schnitt“)	35
2.3.1.2	Quadratische Interpolation	38
2.3.1.3	Heuristische Wahl der Schrittweite	39
2.3.2	Wahl der Suchrichtung	40
2.3.2.1	Gradientenmethode	40
2.3.2.2	Konjugierte Gradientenmethode	45
2.3.2.3	Newton-Methode	51
2.3.2.4	Quasi-Newton-Methode	53
2.3.2.5	Gauss-Newton-Methode	57
2.4	Methode der Vertrauensbereiche	60
2.5	Direkte Suchverfahren	64
2.6	Beispiel: Rosenbrock’s „Bananenfunktion“	66

2.7	Literatur	73
-----	---------------------	----

Vorwort

Wesentliche Teile dieses Skriptums wurden von Prof. Dr.-Ing. Knut GRAICHEN und Univ.-Prof. Dr. techn. Andreas KUGI verfasst. Ihnen gebührt aufrichtiger Dank dafür. Fragen sowie Korrektur- und Verbesserungsvorschläge zu diesem Skriptum können Sie jederzeit an Andreas STEINBÖCK richten.

1 Einleitung

Unter *Optimierung* versteht man gemeinhin die Suche nach einem im Sinne einer bestimmten Zielsetzung bestmöglichen Punkt (optimale Lösung) in einem Entscheidungsraum, wobei bei dieser Suche meist Nebenbedingungen zu berücksichtigen sind. Zur Systematisierung solcher Entscheidungsfindungsprozesse können mathematische Formulierungen und Lösungen von Optimierungsaufgaben (Optimierungsproblemen) verwendet werden. Das vorliegende Skriptum gibt einen Überblick über die mathematische Formulierung und Lösung von Optimierungsaufgaben.

Es wird grundsätzlich zwischen *statischen* und *dynamischen* Optimierungsproblemen unterschieden:

- *Statisches Optimierungsproblem*: Minimierung einer Funktion mit Optimierungsvariablen, die Elemente eines finit-dimensionalen Raumes (z. B. dem Euklidischen Raum) sind
- *Dynamisches Optimierungsproblem*: Minimierung eines Funktional mit Optimierungsvariablen, die Elemente eines unendlich-dimensionalen Raumes sind (z. B. Zeitfunktionen)

In diesem Abschnitt soll anhand von Beispielen der prinzipielle Unterschied zwischen statischen und dynamischen Optimierungsaufgaben verdeutlicht werden.

1.1 Statische Optimierungsprobleme

Unter einem *statischen Optimierungsproblem* wird das Minimieren einer Funktion $f(\mathbf{x})$ unter Berücksichtigung gewisser Nebenbedingungen verstanden, wobei die Optimierungsvariablen $\mathbf{x}$ Elemente des Euklidischen Raumes $\mathbb{R}^n$ sind.

1.1.1 Mathematische Formulierung

Die Standardformulierung eines statischen Optimierungsproblems lautet

$$\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x}) \quad \text{Kostenfunktion} \quad (1.1a)$$

$$\text{u.B.v. } g_i(\mathbf{x}) = 0, \quad i = 1, \dots, p \quad \text{Gleichungsbeschränkungen} \quad (1.1b)$$

$$h_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, q \quad \text{Ungleichungsbeschränkungen.} \quad (1.1c)$$

Ist ein Optimierungsproblem ohne die Gleichungs- und Ungleichungsbeschränkungen (1.1b) und (1.1c) gegeben, spricht man von einem *unbeschränkten Optimierungsproblem*. Im allgemeinen Fall, d. h. unter Berücksichtigung der Nebenbedingungen (1.1b) und (1.1c), handelt es sich um ein *beschränktes Optimierungsproblem*.

Die Menge $\mathcal{X} \subset \mathbb{R}^n$, die die Gleichungs- und Ungleichungsbeschränkungen (1.1b) und (1.1c) erfüllt,

$$\mathcal{X} = \{ \mathbf{x} \in \mathbb{R}^n \mid g_i(\mathbf{x}) = 0, i = 1, \dots, p, \quad h_i(\mathbf{x}) \leq 0, i = 1, \dots, q \} \quad (1.2)$$

wird als *zulässiges Gebiet* oder *zulässige Menge* (englisch: *admissible region* oder *feasible region*) und jedes $\mathbf{x} \in \mathcal{X}$ als *zulässiger Punkt* bezeichnet. Damit lässt sich das statische Optimierungsproblem (1.1) auch in der äquivalenten Form

$$\min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) \quad (1.3)$$

angeben. Im Falle von unbeschränkten Problemen gilt $\mathcal{X} = \mathbb{R}^n$. $\mathcal{X}$ darf nicht die leere Menge sein, da sonst das Optimierungsproblem (1.3) keine Lösung besitzt. Eine weitere notwendige Bedingung für $\mathcal{X}$ kann aus den Gleichungsbeschränkungen (1.1b) abgeleitet werden, da sich durch die p algebraischen Restriktionen $g_i(\mathbf{x}) = 0$ die Anzahl der freien Optimierungsvariablen $\mathbf{x} \in \mathbb{R}^n$ auf $n - p$ reduziert. Somit darf die Anzahl p der Gleichungsbeschränkungen (1.1b) nicht größer als die Anzahl der Optimierungsvariablen $\mathbf{x} \in \mathbb{R}^n$ sein, da die zulässige Menge $\mathcal{X}$ ansonsten leer sein kann.

In der Literatur hat sich weitgehend die Formulierung als Minimierungsproblem (1.1) oder (1.3) durchgesetzt. Analog dazu kann ein Maximierungsproblem ebenfalls als Minimierungsproblem gemäß (1.3) geschrieben werden:

$$\max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) = - \left(\min_{\mathbf{x} \in \mathcal{X}} -f(\mathbf{x}) \right).$$

Neben der Bezeichnung statische Optimierung werden häufig auch die Begriffe *mathematische Programmierung* oder *endlich-dimensionale Optimierung* verwendet. Der Begriff *Programmierung* ist eher im Sinne von *Planung* zu verstehen als im Sinne der Erstellung eines Computerprogramms.

Bei statischen Optimierungsproblemen werden häufig folgende Klassen unterschieden:

- *Lineare Programmierung*: Die Kostenfunktion und die Beschränkungen sind linear (genauer affin).
- *Quadratische Programmierung*: Die Kostenfunktion ist quadratisch, während die Beschränkungen linear (genauer affin) sind.
- *Nichtlineare Programmierung*: Die Kostenfunktion oder mindestens eine Beschränkung ist nichtlinear.
- *Konvexe Programmierung*: Konvexität ist ein mathematischer Begriff, der im Hinblick auf die Optimierung eine besondere Rolle spielt. Ein Optimierungsproblem ist konvex, wenn die Kostenfunktion eine konvexe Funktion und das zulässige Gebiet eine konvexe Menge ist. Bei konvexen Optimierungsproblemen sind die notwendigen Optimalitätsbedingungen erster Ordnung gleichzeitig hinreichend für ein globales Optimum.
- *Integer-Programmierung*: Alle Optimierungsvariablen sind diskret.
- *Mixed-Integer-Programmierung*: Es treten kontinuierliche und diskrete Optimierungsvariablen auf.

1.1.2 Beispiele

Insbesondere die *lineare Programmierung* wird häufig bei ökonomischen Fragestellungen, wie Produktions-, Planungs- oder Investitionsproblemen, eingesetzt. Das folgende Beispiel zeigt eine einfache Portfolio-Optimierung.

Beispiel 1.1 (Portfolio-Optimierung). Ein Anleger möchte 10.000 Euro gewinnbringend investieren und hat die Auswahl zwischen drei Aktienfonds mit unterschiedlicher Gewinnerwartung und Risikoeinstufung:

Fonds	Erwarteter Gewinn/Jahr	Risikoeinstufung
A	10 %	4
B	7 %	2
C	4 %	1

Der Anleger möchte nach einem Jahr mindestens 600 Euro Gewinn erzielen. Andererseits möchte er sein Geld eher konservativ anlegen, d. h. er möchte mindestens 4.000 Euro in Fonds C investieren und das Risiko gemäß der oben gegebenen Risikoeinstufung minimieren. Wie muss der Anleger die 10.000 Euro verteilen, damit diese Kriterien erfüllt werden?

Zunächst werden die Optimierungsvariablen x_1 , x_2 , x_3 eingeführt, die den prozentualen Anteil der investierten 10.000 Euro an den jeweiligen Fonds A, B, C kennzeichnen. Dabei kann x_3 durch die Beziehung

$$x_3 = 1 - x_1 - x_2 \quad (1.4)$$

ersetzt werden. Der geforderte Mindestgewinn von 600 Euro lässt sich als die Beschränkung

$$10.000(0.1x_1 + 0.07x_2 + 0.04(1 - x_1 - x_2)) \geq 600 \quad \Rightarrow \quad 6x_1 + 3x_2 \geq 2 \quad (1.5)$$

ausdrücken. Die Mindestanlage von 4.000 Euro in Fonds C führt zu

$$10.000(1 - x_1 - x_2) \geq 4.000 \quad \Rightarrow \quad x_1 + x_2 \leq 0.6. \quad (1.6)$$

Des Weiteren müssen $x_1 \geq 0$, $x_2 \geq 0$ und $x_3 \geq 0$ erfüllt sein. Mit diesen Beschränkungen und (1.6) ist auch sichergestellt, dass $x_1 \leq 1$, $x_2 \leq 1$ und $x_3 \leq 1$. Das Ziel ist die Minimierung des Anlagerisikos, was sich durch die Funktion

$$f(\mathbf{x}) = 4x_1 + 2x_2 + (1 - x_1 - x_2) = 1 + 3x_1 + x_2 \quad (1.7)$$

ausdrücken lässt.

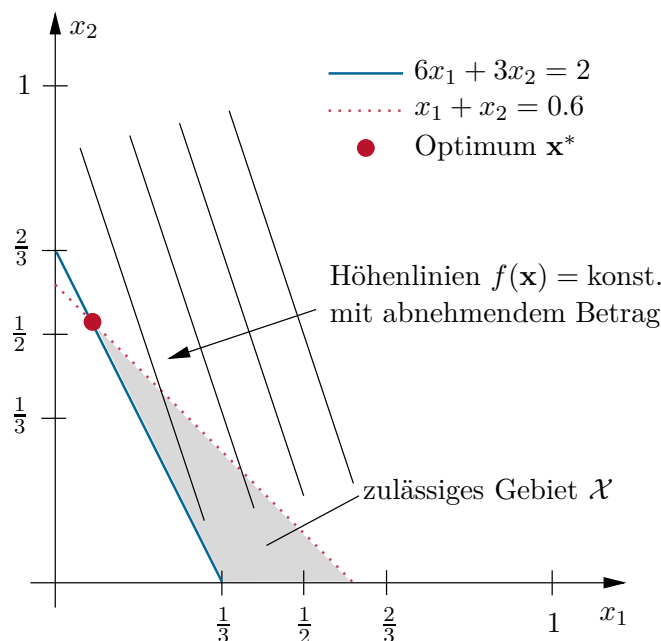


Abbildung 1.1: Grafische Lösung zur Portfolio-Optimierung.

Somit kann das statische Optimierungsproblem in der Form

$$\min_{\mathbf{x} \in \mathbb{R}^2} f(\mathbf{x}) = 1 + 3x_1 + x_2 \quad (1.8a)$$

$$\text{u.B.v. } 6x_1 + 3x_2 \geq 2 \quad (1.8b)$$

$$x_1 + x_2 \leq 0.6 \quad (1.8c)$$

$$x_1 \geq 0 \quad (1.8d)$$

$$x_2 \geq 0 \quad (1.8e)$$

geschrieben werden. Abbildung 1.1 stellt die einzelnen Beschränkungen sowie das zulässige Gebiet grafisch dar. Aus dem Verlauf der Höhenlinien $f(\mathbf{x}) = \text{konst.}$ der Kostenfunktion (1.8a) ist direkt ersichtlich, dass der Punkt $\mathbf{x}^*$ jene Ecke des zulässigen Gebiets $\mathcal{X}$ mit dem niedrigsten Wert von $f(\mathbf{x})$ ist. Somit ergibt sich für die optimale Verteilung der 10.000 Euro auf die einzelnen Fonds

$$x_1^* = \frac{1}{15}, \quad x_2^* = \frac{8}{15}, \quad x_3^* = \frac{6}{15}. \quad (1.9)$$

Das folgende Beispiel der quadratischen Programmierung soll den Einfluss von Beschränkungen auf eine optimale Lösung verdeutlichen.

Beispiel 1.2. Betrachtet wird das (zunächst) unbeschränkte Problem

$$\min_{\mathbf{x} \in \mathbb{R}^2} f(\mathbf{x}) = (x_1 - 2)^2 + (x_2 - 1)^2. \quad (1.10)$$

Die Höhenlinien $f(\mathbf{x}) = \text{konst.}$ der Funktion $f(\mathbf{x})$ sind in Abbildung 1.2 in Abhängigkeit der beiden Optimierungsvariablen $\mathbf{x} = [x_1 \ x_2]^T$ dargestellt. Es ist direkt ersichtlich, dass das Minimum $f(\mathbf{x}^*) = 0$ an der Stelle $\mathbf{x}^* = [2 \ 1]^T$ auftritt.

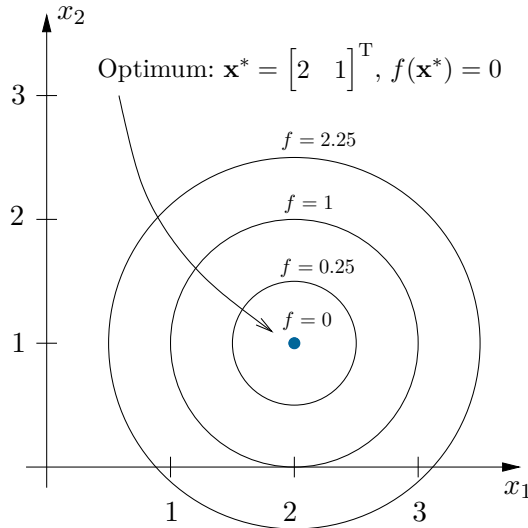


Abb. 1.2: Geometrische Darstellung des unbeschränkten Optimierungsproblems (1.10).

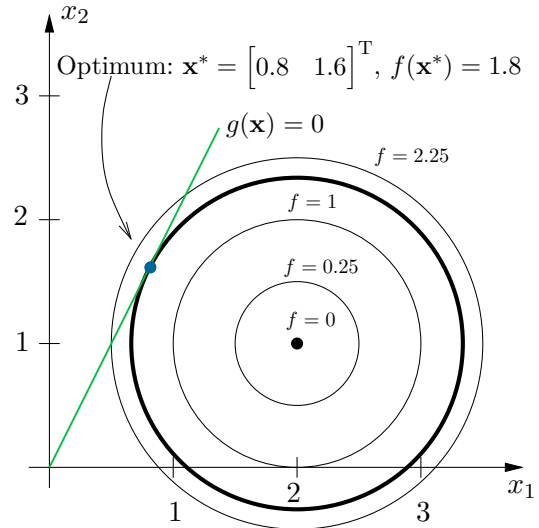


Abb. 1.3: Geometrische Darstellung des beschränkten Optimierungsproblems (1.10), (1.11).

Um den Einfluss verschiedener Beschränkungen zu untersuchen, wird zunächst eine Gleichungsbeschränkung der Form (1.1b) betrachtet

$$g(\mathbf{x}) = x_2 - 2x_1 = 0. \quad (1.11)$$

Die Gleichungsbeschränkung entspricht einer algebraischen Zwangsbedingung, wodurch lediglich noch eine Optimierungsvariable frei wählbar ist. Geometrisch interpretiert bedeutet dies, dass eine mögliche Lösung auf jener Geraden liegen muss, die durch (1.11) definiert wird (siehe Abbildung 1.3). Die optimale Lösung liegt dabei auf dem tangentialen Berührungspunkt der Geraden $g(\mathbf{x}) = 0$ mit der Höhenlinie $f(\mathbf{x}) = 1.8$.

Anstelle der Gleichungsbeschränkung (1.11) wird nun die Ungleichungsbeschränkung

$$h_1(\mathbf{x}) = x_1 + x_2 - 2 \leq 0 \quad (1.12)$$

betrachtet, wodurch sich die Menge der zulässigen Punkte $\mathbf{x} = [x_1 \ x_2]^T$ auf das Gebiet links unterhalb der Geraden $h_1(\mathbf{x}) = 0$ beschränkt (siehe Abbildung 1.4). Das Optimum $f(\mathbf{x}^*) = 0.5$ an der Stelle $\mathbf{x}^* = [1.5 \ 0.5]^T$ befindet sich an der Grenze des zulässigen Gebiets und liegt, wie im vorherigen Szenario, auf einer Höhenlinie, die die Gerade $h_1(\mathbf{x}) = 0$ tangential berührt.

Zusätzlich zur ersten Ungleichungsbeschränkung (1.12) soll eine weitere Ungleichung der Form

$$h_2(\mathbf{x}) = x_1^2 - x_2 \leq 0 \quad (1.13)$$

betrachtet werden, durch die sich die Menge der zulässigen Punkte weiter verkleinert (siehe Abbildung 1.5). Der optimale Punkt $\mathbf{x}^* = [1 \ 1]^T$ mit dem Minimum $f(\mathbf{x}^*) = 1$ liegt nun im Schnittpunkt der Kurven $h_1(\mathbf{x}) = 0$ und $h_2(\mathbf{x}) = 0$, d. h. beide Beschränkungen (1.12) und (1.13) sind aktiv.

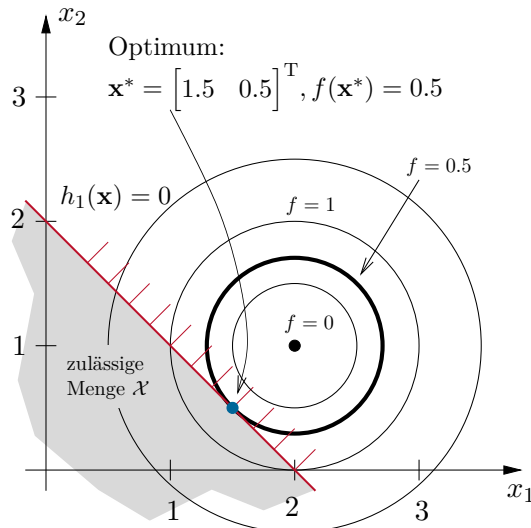


Abb. 1.4: Geometrische Darstellung des beschränkten Optimierungsproblems (1.10), (1.12).

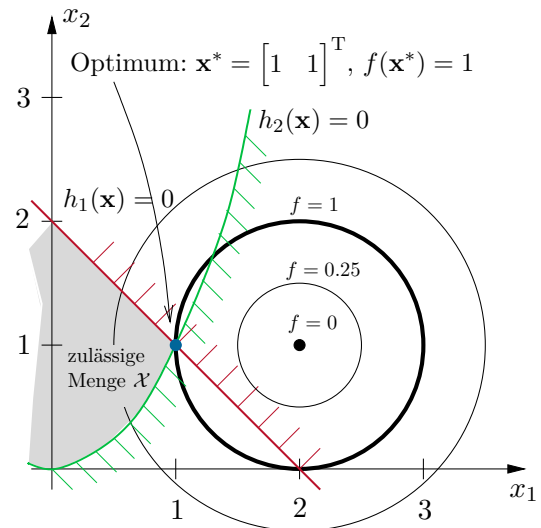


Abb. 1.5: Geometrische Darstellung des beschränkten Optimierungsproblems (1.10), (1.12), (1.13).

Das obige Beispiel 1.2 verdeutlicht den Einfluss von Gleichungs- und Ungleichungsbeschränkungen auf die Lösung (und Lösbarkeit) eines statischen Optimierungsproblems. Die systematische Untersuchung von statischen Optimierungsproblemen sowie die zugehörigen Verfahren zur numerischen Lösung werden in späteren Abschnitten behandelt.

1.2 Dynamische Optimierungsprobleme

Bei den Problemstellungen der statischen Optimierung im vorangegangenen Abschnitt stellen die Optimierungsvariablen $\mathbf{x}$ Elemente aus einem finit-dimensionalen Raum, meist dem Euklidischen Raum $\mathbb{R}^n$, dar. Bei der dynamischen Optimierung hingegen wird in einem Raum von Funktionen einer unabhängigen Variablen nach einem Optimum gesucht. Da es sich bei der unabhängigen Variablen meistens um die Zeit t handelt, wird in diesem Zusammenhang von *dynamischer Optimierung* gesprochen.

1.2.1 Mathematische Formulierung

Die generelle Struktur eines dynamischen Optimierungsproblems lautet

$$\min_{\mathbf{u}(\cdot)} J(\mathbf{u}) = \varphi(t_1, \mathbf{x}(t_1)) + \int_{t_0}^{t_1} l(t, \mathbf{x}(t), \mathbf{u}(t)) dt \quad \text{Kostenfunktional} \quad (1.14a)$$

u.B.v. $\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}, \mathbf{u})$	Systemdynamik	(1.14b)
$\mathbf{x}(t_0) = \mathbf{x}_0$	Anfangsbedingungen	(1.14c)
$\boldsymbol{\psi}(t_1, \mathbf{x}(t_1)) = \mathbf{0}$	Endbedingungen	(1.14d)
$h_i(\mathbf{x}, \mathbf{u}) \leq 0, \quad i = 1, \dots, q$	Ungleichungsbeschr.	(1.14e)

Dabei stellt $\mathbf{u} \in \mathbb{R}^m$ die Eingangsgröße des nichtlinearen Systems (1.14b) mit dem Zustand $\mathbf{x} \in \mathbb{R}^n$ dar. Zusätzlich zu den Anfangsbedingungen (1.14c) sind häufig Endbedingungen der Form (1.14d) gegeben, um z. B. einen gewünschten Zustand $\mathbf{x}_f$ zur Endzeit t_1 zu erreichen (also $\boldsymbol{\psi}(t_1, \mathbf{x}(t_1)) = \mathbf{x}(t_1) - \mathbf{x}_f$). In der Praxis treten häufig Ungleichungsbeschränkungen (1.14e) auf, die z. B. die Begrenzung einer Stellgröße oder Sicherheitsschranken eines Zustandes darstellen können.

Die Aufgabe der dynamischen Optimierung besteht nun darin, eine Eingangstrajektorie $\mathbf{u}(t)$, $t \in [t_0, t_1]$ derart zu finden, dass die Zustandstrajektorie $\mathbf{x}(t)$, $t \in [t_0, t_1]$ des dynamischen Systems (1.14b) mit den Anfangsbedingungen (1.14c) die Endbedingungen (1.14d) erfüllt, die Beschränkungen (1.14e) erfüllt werden und gleichzeitig das Kostenfunktional (1.14a) minimiert wird. Abhängig davon, ob t_1 vorgegeben oder unbekannt ist, spricht man von einer *festen* oder *freien Endzeit* t_1 .

Neben der Bezeichnung dynamische Optimierung werden häufig auch die Begriffe *unendlich-dimensionale Optimierung*, *Optimalsteuerungsproblem* oder *dynamische Programmierung* verwendet. Die folgenden Beispiele erläutern die Problem- und Aufgabenstellung der dynamischen Optimierung.

1.2.2 Beispiele

Beispiel 1.3 (Inverses Pendel). Ein klassisches Problem in der Regelungstechnik ist das inverse Pendel, das an einem Wagen drehbar befestigt ist. Als Beispielproblem soll das seitliche Versetzen des Pendels betrachtet werden

$$\min_{u(\cdot), t_1} J(u) = \int_0^{t_1} 1 + c u^2 \, dt, \quad (1.15a)$$

$$\text{u.B.v.} \quad \begin{bmatrix} 1 & \varepsilon \cos \theta \\ \cos \theta & 1 \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \varepsilon \dot{\theta}^2 \sin \theta + u \\ -\sin \theta \end{bmatrix}, \quad \varepsilon = m/(M + m) \quad (1.15b)$$

$$\mathbf{x}(0) = \begin{bmatrix} 0 & 0 & \pi & 0 \end{bmatrix}^T, \quad \mathbf{x}(t_1) = \begin{bmatrix} 1 & 0 & \pi & 0 \end{bmatrix}^T, \quad (1.15c)$$

$$-1 \leq u \leq 1. \quad (1.15d)$$

Die vereinfachten Bewegungsgleichungen (1.15b) für die Zustände $\mathbf{x} = [x \quad \dot{x} \quad \theta \quad \dot{\theta}]^T$ sind normiert. Der Eingang u stellt die am Wagen angreifende Kraft dar und ist durch (1.15d) beschränkt. Die Masse des Pendels wird mit m , diejenige des Wagens mit M bezeichnet. Abbildung 1.6 zeigt exemplarisch das seitliche Versetzen des Pendels, um die Bewegung zu verdeutlichen.

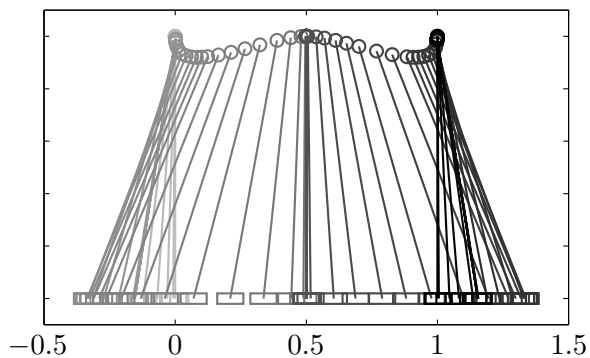


Abbildung 1.6: Momentaufnahmen beim Versetzen des inversen Pendels.

Das Kostenfunktional (1.15a) und somit der Charakter des Optimierungsproblems hängt von dem Parameter c ab. Für $c = 0$ ergibt sich die Aufgabe, die Endzeit t_1 zu minimieren

$$J(u) = \int_0^{t_1} 1 \, dt = t_1. \quad (1.16)$$

Für $c > 0$ wird der Eingang u im Kostenfunktional (und somit der Stellgrößenaufwand) mitberücksichtigt.

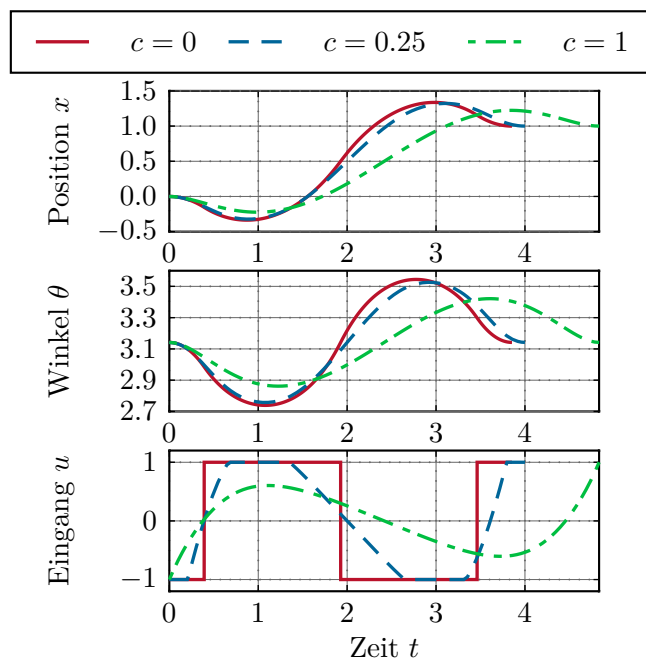


Abbildung 1.7: Optimale Trajektorien beim Versetzen des inversen Pendels.

Abbildung 1.7 zeigt die optimalen Trajektorien für den Parameterwert $\varepsilon = 0.5$ sowie

für die Werte $c = 0$, $c = 0.25$ und $c = 1$. Für $c = 0$ weist der Eingang u sogenanntes Bang-Bang-Verhalten auf, während für $c > 0$ die Steueramplituden kleiner werden und die benötigte Zeit t_1 zunimmt.

Dieses Beispiel verdeutlicht, dass nicht zu jedem Optimierungsproblem eine Lösung existiert, insbesondere wenn die Endzeit t_1 nicht festgelegt ist. Wie aus Abbildung 1.7 ersichtlich, vergrößert sich die Endzeit t_1 bei zunehmender Gewichtung von u^2 im Vergleich zum zeitoptimalen Anteil in dem Kostenfunktional (1.15a). Wenn nur die Stellgröße u gewichtet würde, d. h.

$$J(u) = \int_0^{t_1} u^2 \, dt, \quad (1.17)$$

hätte das Optimierungsproblem keine Lösung, da das Versetzen des Pendels dann unendlich langsam, d. h. mit $t_1 \rightarrow \infty$, ablaufen würde.

Beispiel 1.4 (Goddard-Rakete [1.1, 1.2]). Ein klassisches Optimierungsproblem aus der Raumfahrt ist die Maximierung der Flughöhe einer Rakete unter dem Einfluss von Luftreibung und Erdbeschleunigung. Dieses Problem wurde von Robert H. Goddard im Jahr 1919 aufgestellt und kann in der normierten Form

$$\min_{u(\cdot)} \quad -h(t_1) \quad (1.18a)$$

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

$$h(0) = 1, \quad v(0) = 0, \quad m(0) = 1, \quad m(t_1) = 0.6, \quad (1.18c)$$

$$0 \leq u \leq 3.5 \quad (1.18d)$$

geschrieben werden.

Die Zustandsgrößen sind die Flughöhe h , die Geschwindigkeit v und die Masse m der Rakete. Die Luftreibung $D(h, v)$ hängt über die Funktion

$$D(h, v) = D_0 v^2 e^{[\beta(1-h)]} \quad (1.19)$$

von den Zuständen h und v ab. Die Randbedingungen in (1.18c) umfassen die normierten Anfangsbedingungen sowie die Endbedingung für $m(t_1)$, die dem Leergewicht der Rakete ohne Treibstoff entspricht. Der Eingang des Systems ist der Schub u , der innerhalb der Beschränkungen (1.18d) liegen muss.

In Abbildung 1.8 sind die optimalen Trajektorien für die Goddard-Rakete dargestellt. Die verwendeten Parameterwerte lauten $c = 0.5$, $D_0 = 310$ und $\beta = 500$. Der Schub $u(t)$ ist am Anfang maximal und weist dann einen parabelförmigen Verlauf auf, bevor der Treibstoff verbraucht ist. Dieses Verhalten wird durch den Luftwiderstand $D(h, v)$ hervorgerufen, der mit zunehmender Höhe abnimmt. Es ist somit im Falle eines hohen Luftwiderstandes *optimaler* nicht permanent mit vollem Schub zu fliegen.

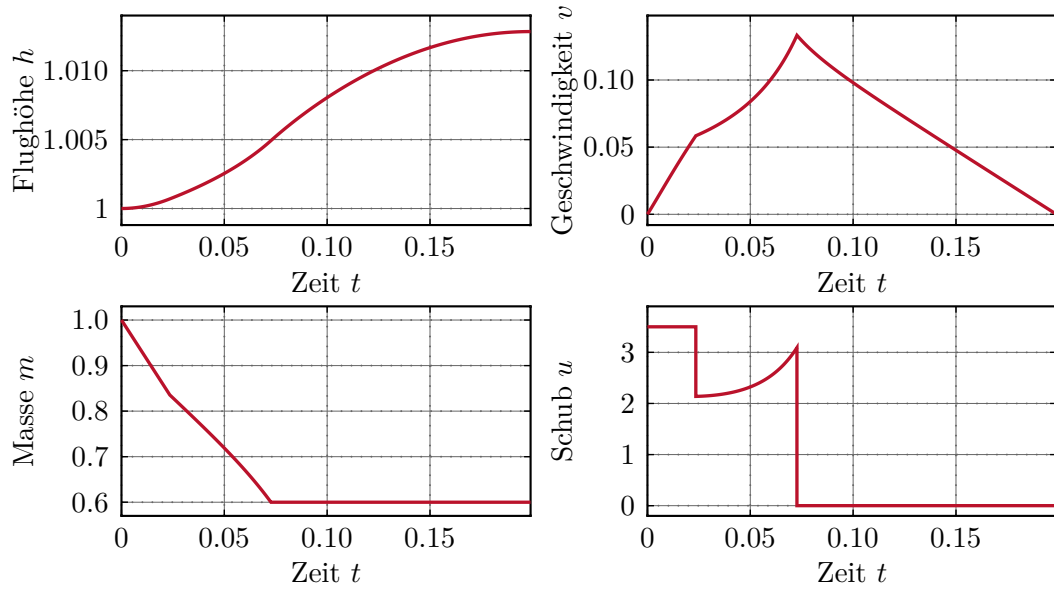


Abbildung 1.8: Trajektorien für die Goddard-Rakete.

Beispiel 1.5 (Ökonomisches Modell [1.3, 1.4]). Ein weiterer Anwendungszweig der dynamischen Optimierung sind wirtschaftliche Prozesse. Das folgende Beispiel beschreibt das Verhalten eines typischen Konsumenten, der Konsum, Freizeit und Bildung über die Lebensdauer optimieren will. Der Bildungsgrad B und das Kapital K eines durchschnittlichen Konsumenten werden durch folgendes Modell beschrieben.

$$\dot{B} = \overbrace{B^\varepsilon u_2 u_3}^{\text{Weiterbildung}} - \overbrace{\delta B}^{\text{Vergessen}}, \quad B(0) = B_0 \quad (1.20a)$$

$$\dot{K} = \underbrace{iK}_{\text{Verzinsung}} + \underbrace{Bu_2 g(u_3)}_{\text{Einkommen}} - \underbrace{u_1}_{\text{Konsum}}, \quad K(0) = K_0 \quad (1.20b)$$

Die Eingangsgrößen sind der Konsum u_1 , der Anteil der Arbeitszeit an der Gesamtzeit u_2 sowie der Anteil der Fortbildungszeit an der Arbeitszeit u_3 . Die Eingänge unterliegen den Beschränkungen

$$u_1 > 0, \quad 0 \leq u_2 \leq 1, \quad 0 \leq u_3 < 1. \quad (1.21)$$

Das Optimierungsziel des Konsumenten ist die Maximierung von Konsum, Freizeit und Bildung über die Lebensdauer von $t_1 = 75$ Jahren, was in dem (zu minimierenden) Kostenfunktional

$$J(\mathbf{u}) = -K^\kappa(t_1) - \int_{t_0}^{t_1} U(t, u_1, u_2, B) e^{-\rho t} dt. \quad (1.22)$$

ausgedrückt ist. Die Nutzenfunktion

$$U(t, u_1, u_2, B) = \alpha_0 u_1^\alpha + \beta_0 (1 - u_2)^\beta + \gamma_0 t B^\gamma \quad (1.23)$$

gewichtet dabei den Konsum u_1 , die Freizeit $1 - u_2$ und den Bildungsgrad B , während der Endwert $-K^\kappa(t_1)$ in (1.22) zusätzlich das Vererbungskapital berücksichtigt.

Die Funktion $g(u_3)$ in (1.20b) ist durch die Parabel $g(u_3) = 1 - (1 - a)u_3 - au_3^2$ gegeben. Die verwendeten Parameterwerte lauten $a = 0.3$, $\alpha = -1$, $\alpha_0 = -1$, $\beta = -0.5$, $\beta_0 = -1$, $\gamma = 0.2$, $\gamma_0 = 5$, $\kappa = 0.2$, $\rho = 0.01$, $\varepsilon = 0.35$, $\delta = 0.01$, $i = 0.04$, $B_0 = 1$ und $K_0 = 30$.

Die optimalen Zeitverläufe des Bildungsgrades $B(t)$ und des Kapitals $K(t)$ sind in Abbildung 1.9 dargestellt. Die ersten 17 Jahre stellen die Lernphase dar (d. h. $u_3 = 1$). Daraufhin folgt eine lange Arbeitsphase von 34 Jahren mit einem hohen Maß an Weiterbildung, bevor in den nächsten 10 Jahren (52.–61. Lebensjahr) eine reine Arbeitsphase mit zusätzlich reduzierter Arbeitszeit u_2 stattfindet. Ab dem 62. Lebensjahr setzt der Ruhestand ein. Der Bildungsgrad B ist besonders hoch im Alter von 30–60 Jahren. Das Kapital K ist negativ während der ersten Lebenshälfte, was der Aufnahme eines Kredites entspricht. Im Laufe des Lebens wird dies aber durch das steigende Einkommen kompensiert.

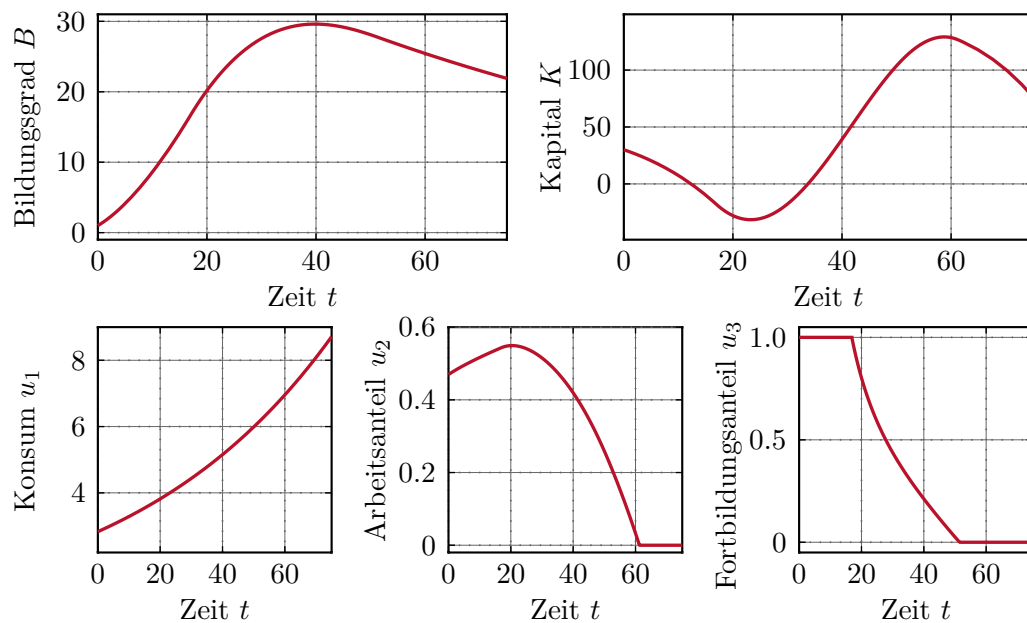


Abbildung 1.9: Optimale Trajektorien für das Konsumentenverhalten.

1.3 Mathematische Grundlagen

In diesem Abschnitt werden kurz einige mathematische Begriffe und Grundkonzepte erläutert, die das Verständnis der weiteren Kapitel erleichtern.

1.3.1 Infimum und Minimum

Definition 1.1 (Infimum). Es sei $\mathcal{Y} \subset \mathbb{R}$ eine nichtleere Menge. Das Infimum von $\mathcal{Y}$, kurz $\inf \mathcal{Y}$ geschrieben, bezeichnet die größte untere Schranke von $\mathcal{Y}$, d. h. es existiert eine Zahl $\alpha = \inf \mathcal{Y}$ so, dass gilt

- (a) $x \geq \alpha$ für alle $x \in \mathcal{Y}$
- (b) für alle $\bar{\alpha} > \alpha$ existiert ein $x \in \mathcal{Y}$ so, dass $x < \bar{\alpha}$.

Existiert für eine nichtleere Menge $\mathcal{Y}$ ein Infimum, so muss dieses nicht automatisch in $\mathcal{Y}$ enthalten sein. Als Beispiel dazu betrachte man die Menge $\mathcal{Y} = (0, +\infty)$. In diesem Fall gilt offensichtlich $\inf \mathcal{Y} = 0 \notin \mathcal{Y}$.

Für die folgende Definition wird angenommen, dass $\mathcal{X} \subset \mathbb{R}^n$ die zulässige Menge des betrachteten Optimierungsproblems gemäß (1.2) bezeichnet.

Definition 1.2 (Globale und lokale Minima). Die Funktion $f(\mathbf{x})$ besitzt in $\mathcal{X}$ an der Stelle $\mathbf{x}^*$

- (a) ein *lokales Minimum*, falls für eine Norm $\|\cdot\|$ ein $\varepsilon > 0$ so existiert, dass gilt $f(\mathbf{x}^*) \leq f(\mathbf{x})$ für alle $\mathbf{x} \in \{\mathbf{x} \in \mathcal{X} \mid \|\mathbf{x} - \mathbf{x}^*\| < \varepsilon\}$,
- (b) ein *strikt lokales Minimum*, falls für eine Norm $\|\cdot\|$ ein $\varepsilon > 0$ so existiert, dass gilt $f(\mathbf{x}^*) < f(\mathbf{x})$ für alle $\mathbf{x} \in \{\mathbf{x} \in \mathcal{X} \mid \|\mathbf{x} - \mathbf{x}^*\| < \varepsilon, \mathbf{x} \neq \mathbf{x}^*\}$,
- (c) ein *globales Minimum*, falls $f(\mathbf{x}^*) \leq f(\mathbf{x})$ für alle $\mathbf{x} \in \mathcal{X}$, und
- (d) ein *strikt globales Minimum*, falls $f(\mathbf{x}^*) < f(\mathbf{x})$ für alle $\mathbf{x} \in \mathcal{X} \setminus \{\mathbf{x}^*\}$.

Statt dem Begriff *strikt* wird gelegentlich auch der Begriff *eindeutig* verwendet. Sprachlich etwas unscharf werden zumeist sowohl $\mathbf{x}^*$ als auch $f(\mathbf{x}^*)$ als Minimum bezeichnet. Abbildung 1.10 zeigt unterschiedliche Arten von Minima für eine Funktion in einer Optimierungsvariablen.

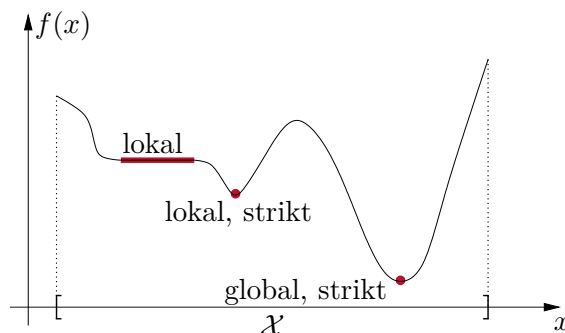


Abbildung 1.10: Verschiedene Minima einer Funktion $f(x)$ mit $x \in \mathcal{X} \subset \mathbb{R}$.

An dieser Stelle sei betont, dass ein Punkt $\mathbf{x}^*$, der die Funktion $f(\mathbf{x})$ in der Menge $\mathcal{X}$ minimiert, in $\mathcal{X}$ enthalten sein muss. Ein Punkt $\mathbf{x}$, dessen Funktionswert $f(\mathbf{x})$ gerade dem Infimum $\inf\{f(\mathbf{x}) \mid \mathbf{x} \in \mathcal{X}\}$ entspricht, muss jedoch nicht existieren, auch nicht außerhalb von $\mathcal{X}$.

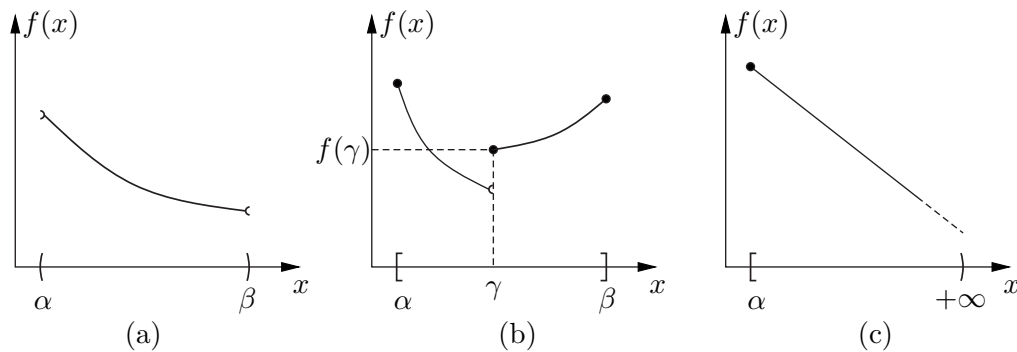


Abbildung 1.11: Nichtexistenz von Minima.

Die Menge aller Minima (strenggenommen die Menge aller Punkte an denen $f(\mathbf{x})$ ein Minimum aufweist) wird oftmals in der Form

$$\mathcal{G} = \arg \min \{f(\mathbf{x}) \mid \mathbf{x} \in \mathcal{X}\} \quad (1.24)$$

angeschrieben, wobei natürlich $\mathcal{G} \subseteq \mathcal{X}$ gilt und $\mathcal{G}$ sowohl leer sein kann als auch aus endlich oder unendlich vielen Punkten $\mathbf{x} \in \mathcal{X}$ bestehen kann. Im Falle eines strikten globalen Minimums in $\mathcal{X}$ versteht man unter dem Ausdruck $\bar{\mathbf{x}} = \arg \min \{f(\mathbf{x}) \mid \mathbf{x} \in \mathcal{X}\}$ meist jene Funktion, die gerade den Punkt $\bar{\mathbf{x}} \in \mathcal{X}$ zurückgibt, der die Funktion $f(\mathbf{x})$ global minimiert.

1.3.2 Existenz von Minima

Abbildung 1.11 zeigt drei Fälle, bei denen kein Minimum existiert. In Abbildung 1.11(a) ist das Infimum von $f(x)$ in der Menge $\mathcal{X} = (\alpha, \beta)$ durch $f(\beta)$ gegeben. Da aber $\mathcal{X}$ nicht abgeschlossen ist und somit $\beta \notin \mathcal{X}$, existiert in diesem Fall kein Minimum. In Abbildung 1.11(b) ist der linksseitige Grenzwert $\lim_{x \rightarrow \gamma^-} f(x)$ das Infimum von $f(x)$ in der Menge $\mathcal{X} = [\alpha, \beta]$. Auch in diesem Fall existiert auf Grund der Unstetigkeit von $f(x)$ das Minimum nicht. Im letzten Fall, Abbildung 1.11(c), existiert das Minimum ebenfalls nicht, da $f(x)$ in der unbeschränkten Menge $\mathcal{X} = \{x \in \mathbb{R} \mid x \geq \alpha\}$ nach unten hin nicht beschränkt ist.

Der nachfolgende Satz gibt nun hinreichende Bedingungen für die Existenz einer Lösung von Optimierungsproblemen an.

Satz 1.1 (Satz von Weierstrass). *Es sei $\mathcal{X}$ eine nichtleere und kompakte (abgeschlossene und beschränkte) Menge und $f : \mathcal{X} \rightarrow \mathbb{R}$ stetig auf $\mathcal{X}$. Dann ist die Menge aller Minima $\mathcal{G} = \arg \min \{f(\mathbf{x}) \mid \mathbf{x} \in \mathcal{X}\}$ nichtleer und kompakt.*

Der Beweis dieses Satzes ist beispielsweise in [1.5, 1.6] zu finden. Satz 1.1 beschreibt eine hinreichende jedoch nicht notwendige Bedingung für die Existenz einer optimalen Lösung. Als Beispiel dazu betrachte man die Minimierungsaufgabe $\min_{x \in (-1,1)} x^2$, die zeigt, dass mit $x = 0$ ein Minimum gegeben ist, obwohl die Menge $\mathcal{X} = (-1, 1)$ offen und damit nicht kompakt ist.

1.3.3 Gradient und Hessematrix

Die Berechnung von Ableitungen erster und zweiter Ordnung einer Kostenfunktion $f(\mathbf{x})$ ist von fundamentaler Bedeutung in der Optimierung. Da im Falle von unstetigen Funktionen oder unstetigen Ableitungen Probleme auftreten können (sowohl numerischer als auch theoretischer Natur), wird oft angenommen, dass alle Funktionen eines Optimierungsproblems stetig und hinreichend oft differenzierbar sind. So nicht anders erwähnt, gilt dies auch für diese Vorlesung. Im Rahmen der Optimierungsalgorithmen spielen der Gradient und die Hessematrix eine bedeutende Rolle.

Definition 1.3 (Gradient). Es sei $f : \mathcal{X} \rightarrow \mathbb{R}$ eine stetig differenzierbare Funktion, d. h. $f \in C^1$. Dann bezeichnet

$$(\nabla f)(\mathbf{x}) = \left(\frac{\partial f}{\partial \mathbf{x}} \right)^T = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} \quad (1.25)$$

den Gradienten (also die 1. partielle Ableitung) von $f(\mathbf{x})$ an der Stelle $\mathbf{x} = [x_1 \ \dots \ x_n]^T$.

Definition 1.4 (Hessematrix). Es sei $f : \mathcal{X} \rightarrow \mathbb{R}$ eine zweifach stetig differenzierbare Funktion, d. h. $f \in C^2$. Dann bezeichnet

$$(\nabla^2 f)(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix} \quad (1.26)$$

die Hessematrix (also die 2. partielle Ableitung) von $f(\mathbf{x})$ an der Stelle $\mathbf{x} = [x_1 \ \dots \ x_n]^T$.

Im Falle von Funktionen $f(x)$ mit nur einem skalaren Argument wird die ∇ -Notation häufig durch $f'(x)$ und $f''(x)$ ersetzt.

Aus der Stetigkeit der 2. partiellen Ableitungen folgt Kommutativität, d. h.

$$\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial^2 f}{\partial x_j \partial x_i}. \quad (1.27)$$

Daraus ergibt sich $(\nabla^2 f)(\mathbf{x}) = (\nabla^2 f)^T(\mathbf{x})$, d. h. die Hessematrix ist symmetrisch. Folglich hat sie stets rein reelle Eigenwerte. In der Optimierung ist oft von Bedeutung, ob Hessematrizen positiv (semi-)definit sind. Diese Eigenschaft kann wie folgt untersucht werden.

Satz 1.2 (Definitheit von Matrizen). Die Definitheit einer symmetrischen Matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ lässt sich durch folgende Bedingungen charakterisieren:

Matrix $\mathbf{A}$ ist	(a) für alle $\mathbf{p} \in \mathbb{R}^n$ mit $\mathbf{p} \neq \mathbf{0}$ gilt	(b) alle n Eigen- werte λ_i sind	(c) für alle n Haupt- minoren D_i gilt
positiv semi-definit:	$\mathbf{p}^T \mathbf{A} \mathbf{p} \geq 0$	≥ 0	-
positiv definit:	$\mathbf{p}^T \mathbf{A} \mathbf{p} > 0$	> 0	$D_i > 0$
negativ semi-definit:	$\mathbf{p}^T \mathbf{A} \mathbf{p} \leq 0$	≤ 0	-
negativ definit:	$\mathbf{p}^T \mathbf{A} \mathbf{p} < 0$	< 0	$(-1)^{i+1} D_i < 0$

Die Eigenwerte λ_i , $i = 1, \dots, n$ der Matrix $\mathbf{A}$ sind die Lösungen der Gleichung

$$\det(\lambda \mathbf{E} - \mathbf{A}) = 0, \quad (1.28)$$

wobei $\mathbf{E}$ die Einheitsmatrix der Dimension n darstellt. Die Hauptminoren D_i sind die Determinanten der linken oberen Untermatrizen von $\mathbf{A}$,

$$D_1 = a_{11}, \quad D_2 = \det \begin{pmatrix} a_{11} & a_{12} \\ a_{12} & a_{22} \end{pmatrix}, \quad \dots, \quad D_n = \det \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{1n} & \dots & a_{nn} \end{pmatrix}, \quad (1.29)$$

wobei a_{ij} das Element in der i -ten Zeile und j -ten Spalte von $\mathbf{A}$ bezeichnet, d. h. $\mathbf{A} = [a_{ij}]_{i,j=1,\dots,n}$. Um die Definitheit einer symmetrischen Matrix $\mathbf{A}$ zu bestimmen, muss lediglich eine der drei Bedingungen (a)–(c) in Satz 1.2 ausgewertet werden, da jede für sich notwendig und hinreichend ist. Das Kriterium (c) wird auch *Sylvester-Kriterium* genannt und kann nicht für semi-definite Matrizen verwendet werden.

Die positive Definitheit einer symmetrischen Matrix $\mathbf{A}$ kann alternativ zu den Bedingungen (a)–(c) aus Satz 1.2 auch mit Hilfe der *Cholesky-Faktorisierung* überprüft werden. Gemäß dieser Methode gilt, dass eine symmetrische Matrix $\mathbf{A}$ genau dann positiv definit ist, wenn sie sich in der Form $\mathbf{A} = \mathbf{G}\mathbf{G}^T$ faktorisieren lässt, wobei $\mathbf{G}$ eine untere Dreiecksmatrix mit positiven Diagonaleinträgen ist.

Bei der Abschätzung von Funktionen werden häufig der Gradient und die Hessematrix im Rahmen des *Mittelwertsatzes* (Satz von Taylor) verwendet.

Satz 1.3 (Mittelwertsatz, Satz von Taylor). Es sei $f(\mathbf{x})$ eine stetig differenzierbare Funktion, d. h. $f \in C^1$, in einer Menge $\mathcal{X}$, die das Liniensegment $[\mathbf{x}_1, \mathbf{x}_2]$ beinhaltet, dann existiert eine reelle Zahl α , $0 \leq \alpha \leq 1$ so, dass gilt

$$f(\mathbf{x}_2) = f(\mathbf{x}_1) + (\mathbf{x}_2 - \mathbf{x}_1)^T (\nabla f)(\mathbf{x}_1 + \alpha(\mathbf{x}_2 - \mathbf{x}_1)) . \quad (1.30)$$

Ist die Funktion $f(\mathbf{x})$ zweifach stetig differenzierbar, d. h. $f \in C^2$, dann existiert eine reelle Zahl α , $0 \leq \alpha \leq 1$ so, dass die Beziehung

$$f(\mathbf{x}_2) = f(\mathbf{x}_1) + (\mathbf{x}_2 - \mathbf{x}_1)^T (\nabla f)(\mathbf{x}_1) + \frac{1}{2} (\mathbf{x}_2 - \mathbf{x}_1)^T \left(\nabla^2 f \right) (\mathbf{x}_1 + \alpha(\mathbf{x}_2 - \mathbf{x}_1)) (\mathbf{x}_2 - \mathbf{x}_1) \quad (1.31)$$

gilt.

1.3.4 Berechnung von Ableitungen

Zur konkreten Berechnung von Ableitungen können verschiedene Verfahren verwendet werden.

Analytisches Differenzieren

Ist $f(\mathbf{x})$ als analytischer Ausdruck gegeben, so können Ableitungen direkt mittels analytischer Differenzierung berechnet werden.

Algorithmisches Differenzieren

In der Optimierung steht die Kostenfunktion $f(\mathbf{x})$ häufig nicht als geschlossener analytischer Ausdruck zur Verfügung sondern in Form von Funktionen (Algorithmen), die in einem Computerprogramm realisiert sind. In diesem Fall bietet das *algorithmische Differenzieren*, gelegentlich auch *automatisches Differenzieren* genannt, eine komfortable Möglichkeit Ableitungen zu berechnen. Das Verfahren nutzt die Regeln der analytischen Differentiation (Differentiationsregeln für elementare Funktionen, Kettenregel, Produktregel, Summenregel, Quotientenregel, Ableitungsregel für inverse Funktionen, etc.) um ein neues Computerprogramm zu erstellen, das die gewünschten Ableitungen von $f(\mathbf{x})$ berechnet. Die Programmschritte werden dabei so organisiert, dass eine effiziente und zugleich möglichst genaue Berechnung der Ableitungen erreicht wird. Mehr Informationen über das algorithmische Differenzieren ist z. B. unter <http://www.autodiff.org> oder in [1.7] zu finden.

Ableitungsberechnung mit Differenzenquotienten

Eine näherungsweise Ableitungsberechnung ist auch numerisch durch Bildung von Differenzenquotienten möglich [1.8]. Für eine allgemeine, hinreichend oft differenzierbare Funktion $f(\mathbf{x})$ sind in Tabelle 1.1 Beispiele für Differenzenquotienten gegeben. Hierbei ist h die Schrittweite und $\mathbf{e}_i$ der Einheitsvektor mit dem Eintrag 1 an der Stelle i . Die Tabelle enthält auch die Ordnungen der Fehler, welche bei dieser näherungsweisen Ableitungsberechnung entstehen können. Das sind *Abschneidefehler* und *Rundungsfehler*, wobei e_r der maximale relative Fehler zufolge von Rundungsoperationen bei Gleitkommaarithmetik ist.

Die Herleitung von Differenzenquotienten und die Berechnung der zugehörigen Abschneidefehler können mittels Taylorreihenentwicklung am Punkt $\mathbf{x}$ bzw. dem Mittelwertsatz 1.3 erfolgen [1.9]. Der Abschneidefehler ergibt sich, weil die Taylorreihenentwicklung bei einer bestimmten Ordnung abgebrochen wird und eine finite Schrittweite $h > 0$ verwendet

Ableitung, Richtung	Formel	Abschneide- fehler	Rundungs- fehler
1. Ableitung, vorwärts	$(\nabla f)(\mathbf{x}) \approx \left[\frac{f(\mathbf{x} + h\mathbf{e}_i) - f(\mathbf{x})}{h} \right]_{i=1,\dots,n}$	$\mathcal{O}(h)$	$\mathcal{O}(e_r h^{-1})$
1. Ableitung, rückwärts	$(\nabla f)(\mathbf{x}) \approx \left[\frac{f(\mathbf{x}) - f(\mathbf{x} - h\mathbf{e}_i)}{h} \right]_{i=1,\dots,n}$	$\mathcal{O}(h)$	$\mathcal{O}(e_r h^{-1})$
1. Ableitung, zentral	$(\nabla f)(\mathbf{x}) \approx \left[\frac{f(\mathbf{x} + h\mathbf{e}_i) - f(\mathbf{x} - h\mathbf{e}_i)}{2h} \right]_{i=1,\dots,n}$	$\mathcal{O}(h^2)$	$\mathcal{O}(e_r h^{-1})$
2. Ableitung, vorwärts	$(\nabla^2 f)(\mathbf{x}) \approx [f(\mathbf{x} + h\mathbf{e}_i + h\mathbf{e}_j) - f(\mathbf{x} + h\mathbf{e}_i) - f(\mathbf{x} + h\mathbf{e}_j) + f(\mathbf{x})]_{i,j=1,\dots,n} / h^2$	$\mathcal{O}(h)$	$\mathcal{O}(e_r h^{-2})$
2. Ableitung, zentral	$(\nabla^2 f)(\mathbf{x}) \approx [f(\mathbf{x} + h\mathbf{e}_i + h\mathbf{e}_j) - f(\mathbf{x} + h\mathbf{e}_i - h\mathbf{e}_j) - f(\mathbf{x} - h\mathbf{e}_i + h\mathbf{e}_j) + f(\mathbf{x} - h\mathbf{e}_i - h\mathbf{e}_j)]_{i,j=1,\dots,n} / (4h^2)$	$\mathcal{O}(h^2)$	$\mathcal{O}(e_r h^{-2})$

Tabelle 1.1: Differenzenquotienten.

werden muss. Der Abschneidefehler ist also der Methode geschuldet und entsteht selbst bei exakter Berechnung des Funktionswertes $f(\mathbf{x})$. Der Rundungsfehler ist auf die praktisch nicht exakte numerische Berechnung von $f(\mathbf{x})$, $f(\mathbf{x} \pm h\mathbf{e}_i)$ und $f(\mathbf{x} \pm h\mathbf{e}_i \pm h\mathbf{e}_j)$ bei Verwendung von Gleitkommaarithmetik zurückzuführen. Wie auch nachfolgendes Beispiel zeigt, kann der Rundungsfehler für $h \rightarrow 0$ unbeschränkt anwachsen.

Beispiel 1.6. Beispielhaft soll nun die Herleitung und Fehlerberechnung für den Vorwärtsdifferenzenquotient zur Approximation des Gradienten (erste Zeile in Tabelle 1.1) durchgeführt werden. Eine Taylorreihenentwicklung von $f(\mathbf{x})$ liefert unter Berücksichtigung des Mittelwertsatzes 1.3

$$f(\mathbf{x} + h\mathbf{e}_i) = f(\mathbf{x}) + h \left. \frac{\partial f}{\partial x_i} \right|_{\mathbf{x}} + \frac{h^2}{2} \left. \frac{\partial^2 f}{\partial x_i^2} \right|_{\mathbf{x} + \alpha h\mathbf{e}_i} \quad (1.32)$$

mit $\alpha \in (0, 1)$. Daraus folgt

$$\left. \frac{\partial f}{\partial x_i} \right|_{\mathbf{x}} = \frac{f(\mathbf{x} + h\mathbf{e}_i) - f(\mathbf{x})}{h} - \underbrace{\frac{h}{2} \left. \frac{\partial^2 f}{\partial x_i^2} \right|_{\mathbf{x} + \alpha h\mathbf{e}_i}}_{\text{Abschneidefehler, } \mathcal{O}(h)}. \quad (1.33)$$

Geht man nun davon aus, dass e_r der maximale relative Fehler durch die in der Gleitkommaarithmetik notwendigen Rundungsoperationen bei der Auswertung der

Funktionen $f(\mathbf{x} + h\mathbf{e}_i)$ und $f(\mathbf{x})$ ist, so ergibt sich im schlechtesten Fall der berechnete Wert

$$\begin{aligned} & \frac{f(\mathbf{x} + h\mathbf{e}_i)(1 + e_r) - f(\mathbf{x})(1 - e_r)}{h} \\ &= \frac{f(\mathbf{x} + h\mathbf{e}_i) - f(\mathbf{x})}{h} + \underbrace{\frac{(f(\mathbf{x} + h\mathbf{e}_i) + f(\mathbf{x}))e_r}{h}}_{\text{Rundungsfehler, } \mathcal{O}(e_r h^{-1})} \end{aligned} \quad (1.34)$$

als Approximation von $\partial f / \partial x_i|_{\mathbf{x}}$. Der Gesamtfehler folgt daher in der Form

$$\begin{aligned} & \underbrace{\frac{f(\mathbf{x} + h\mathbf{e}_i) - f(\mathbf{x})}{h}}_{\text{Berechneter Wert}} + \underbrace{\frac{(f(\mathbf{x} + h\mathbf{e}_i) + f(\mathbf{x}))e_r}{h}}_{\text{Rundungsfehler}} - \underbrace{\frac{\partial f}{\partial x_i}|_{\mathbf{x}}}_{\text{Exakter Wert}} \\ &= \frac{h}{2} \frac{\partial^2 f}{\partial x_i^2} \bigg|_{\mathbf{x} + \alpha h \mathbf{e}_i} + \frac{(f(\mathbf{x} + h\mathbf{e}_i) + f(\mathbf{x}))e_r}{h} . \end{aligned} \quad (1.35)$$

Abbildung 1.12 zeigt wie in diesem Fall die Absolutwerte von Abschneide-, Rundungs- und Gesamtfehler von h abhängen. Es existiert also eine optimale Schrittweite h , um den Gesamtfehler zu minimieren.

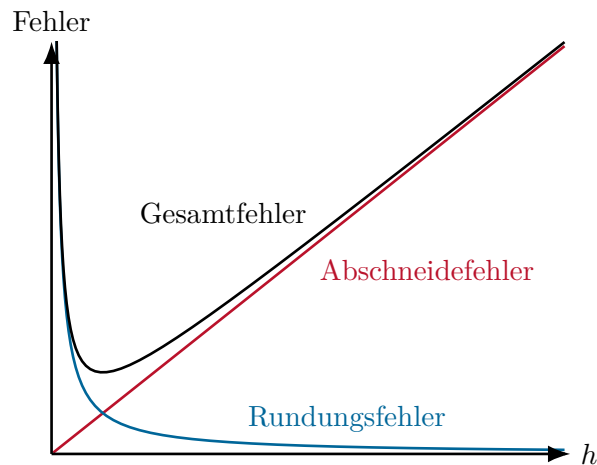


Abbildung 1.12: Absolutwerte der Fehler bei der Approximation des Gradienten durch den Vorwärtsdifferenzenquotienten in Abhängigkeit der Schrittweite h .

In Tabelle 1.1 und Beispiel 1.6 wurde nur der Rundungsfehler durch Gleitkommaarithmetik bei der Auswertung der Funktionen f berücksichtigt. Tatsächlich treten solche Rundungsfehler aber auch bei der Berechnung von Differenzenquotienten selbst auf, also z. B. bei der Auswertung von

$$\frac{f(\mathbf{x} + h\mathbf{e}_i) - f(\mathbf{x})}{h} . \quad (1.36)$$

Bei der Differenzbildung im Zähler dieses Ausdrucks kann es zu erheblichen Genauigkeits-

verlusten durch *Auslöschungsfehler* kommen [1.10]. Beim Rechnen mit Gleitkommaarithmetik müssen alle Eingangsgrößen, Zwischenergebnisse der auftretenden Elementaroperationen und Endergebnisse auf die Zahl der Mantissenstellen gerundet werden. Als *Auslöschung* bezeichnet man das Annullieren führender Mantissenstellen bei der Subtraktion zweier (ähnlicher) Zahlen [1.11]. Geht man davon aus, dass mit M Mantissenstellen gerechnet wird und zwei zu subtrahierende Zahlen sich in m führenden Mantissenstellen gleichen, so bleiben im Ergebnis nur noch $M - m$ gültige von Null verschiedene Mantissenstellen übrig, was direkt die erzielbare Genauigkeit einschränkt. Je ähnlicher sich daher zwei zu subtrahierende Zahlen sind, desto kleiner wird $M - m$, was bei der Berechnung von Differenzenquotienten im Falle $h \rightarrow 0$ zur Unbrauchbarkeit des Ergebnisses führt.

Beispiel 1.7 (Auslöschungsfehler). Man betrachte die Rechnung

$$0.123\,456 - 0.123\,455 = 0.000\,001 . \quad (1.37)$$

Ist in den beiden Eingangsgrößen auch nur die letzte Nachkommastelle zufolge von früheren Rundungsfehlern unsicher, so kann das Ergebnis 0.000 001 völlig falsch sein, d. h. keine einzige gültige von Null verschiedene Nachkommastelle besitzen.

Neben den Fehlerordnungen, Rundungsfehlern und Auslöschungsfehlern spielt auch der jeweilige Rechenaufwand bei der Wahl eines Differenzenquotienten eine Rolle. Bei der näherungsweisen Berechnung von $(\nabla f)(\mathbf{x})$ erfordern einseitige Differenzenquotienten $n + 1$ Auswertungen der Funktion f und der zentrale Differenzenquotient $2n$ solche Auswertungen. D. h. bei der näherungsweisen Berechnung von $(\nabla f)(\mathbf{x})$ steigt der Berechnungsaufwand mit der Ordnung $\mathcal{O}(n)$. Bei der näherungsweisen Berechnung von $(\nabla^2 f)(\mathbf{x})$ steigt der Berechnungsaufwand mit der Ordnung $\mathcal{O}(n^2)$.

Ableitungsberechnung mittels komplexer Funktionsauswertung

Im Englischen ist die nachfolgend beschriebene Methode zur näherungsweisen Ableitungsberechnung als *complex step derivative* bekannt [1.12, 1.13]. Diese numerische Methode ist nur auf *holomorphe* Funktionen anwendbar. Holomorphe Funktionen werden auch als *komplex differenzierbar* bezeichnet.

Definition 1.5 (Holomorphe Funktion). Eine Funktion $f : \mathcal{X} \rightarrow \mathbb{C}$ mit $\mathcal{X} \subseteq \mathbb{C}$ nennt man *holomorph*, falls der Grenzwert

$$\lim_{h \rightarrow 0} \frac{f(z + h) - f(z)}{h} \quad \forall z \in \mathcal{X} \quad (1.38)$$

mit $h \in \mathbb{C}$ existiert.

Eine Funktion $f : \mathcal{X} \rightarrow \mathbb{C}$ in mehreren Variablen, d. h. $\mathcal{X} \subseteq \mathbb{C}^n$, ist genau dann holomorph, wenn sie holomorph ist bezüglich jeder einzelnen Variable bei festgehaltenen übrigen Variablen.

Die erste Ableitung einer holomorphen reellen Funktion $f(\mathbf{x})$, d. h. $f : \mathbb{R}^n \rightarrow \mathbb{R}$, kann näherungsweise mit der Formel

$$(\nabla f)(\mathbf{x}) \approx \left[\frac{\operatorname{Im}(f(\mathbf{x} + h\mathbf{Ie}_i))}{h} \right]_{i=1,\dots,n} \quad (1.39)$$

berechnet werden, wobei $h > 0$ eine kleine reelle Schrittweite darstellt. Zur Herleitung von (1.39), wird $f(\mathbf{x})$ zunächst in eine Taylorreihe

$$f(\mathbf{x} + h\mathbf{Ie}_i) = f(\mathbf{x}) + h\mathbf{I} \left. \frac{\partial f}{\partial x_i} \right|_{\mathbf{x}} - \frac{h^2}{2} \left. \frac{\partial^2 f}{\partial x_i^2} \right|_{\mathbf{x}} - \frac{h^3}{6} \left. \frac{\partial^3 f}{\partial x_i^3} \right|_{\mathbf{x} + \alpha h\mathbf{Ie}_i} \quad (1.40)$$

mit $\alpha \in (0, 1)$ entwickelt. Dividiert man den Imaginärteil von (1.40) durch h , so folgen daraus

$$\left. \frac{\partial f}{\partial x_i} \right|_{\mathbf{x}} = \frac{\operatorname{Im}(f(\mathbf{x} + h\mathbf{Ie}_i))}{h} + \underbrace{\frac{h^2}{6} \left. \frac{\partial^3 f}{\partial x_i^3} \right|_{\mathbf{x} + \alpha h\mathbf{Ie}_i}}_{\text{Abschneidefehler, } \mathcal{O}(h^2)} \quad (1.41)$$

und somit direkt die Komponenten von (1.39). In ähnlicher Weise werden in [1.14] Formeln zur näherungsweisen Berechnung von zweiten Ableitungen mittels komplexer Funktionsauswertung hergeleitet.

$(\nabla f)(\mathbf{x})$ kann also mit nur n komplexen Auswertungen der Funktion f näherungsweise berechnet werden, wobei ein Abschneidefehler der Ordnung $\mathcal{O}(h^2)$ erzielt wird. Der Rundungsfehler besitzt die Ordnung $\mathcal{O}(e_r h^{-1})$. Ein zentraler Vorteil dieser Methode gegenüber der Ableitungsberechnung mit Differenzenquotienten ist, dass bei der Berechnung der ersten Ableitung kein Auslöschungsfehler auftritt (keine Subtraktion ähnlicher Zahlen nötig), weshalb h sehr klein gewählt werden kann. Dies gilt im Allgemeinen nicht mehr für höhere Ableitungen. Der Nachteil dieser Methode liegt im geringfügig höheren numerischen Aufwand, da mit komplexen Zahlen gerechnet werden muss. In [1.13] wird der Zusammenhang zwischen dieser Methode und dem algorithmischen Differenzieren diskutiert.

Aus dem Realteil von (1.40) folgt noch

$$f(\mathbf{x}) = \operatorname{Re}(f(\mathbf{x} + h\mathbf{Ie}_i)) + \mathcal{O}(h^2) . \quad (1.42)$$

Wird also eine Ableitung $\partial f / \partial x_i$ berechnet, so erhält man ohne zusätzliche Funktionsauswertungen auch eine Näherung für den Wert $f(\mathbf{x})$.

1.3.5 Satz über implizite Funktionen

Satz 1.4 (Satz über implizite Funktionen). Es sei $\mathbf{f} : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^m$ eine stetig differenzierbare Funktion. Sind an einem Punkt $(\bar{\mathbf{x}}, \bar{\mathbf{y}}) \in \mathbb{R}^n \times \mathbb{R}^m$

$$\mathbf{f}(\bar{\mathbf{x}}, \bar{\mathbf{y}}) = \mathbf{0} \quad (1.43)$$

und

$$\operatorname{rang} \left(\left. \frac{\partial \mathbf{f}(\bar{\mathbf{x}}, \mathbf{y})}{\partial \mathbf{y}} \right|_{\mathbf{y}=\bar{\mathbf{y}}} \right) = m \quad (1.44)$$

erfüllt, so existieren eine offene Umgebung $\mathcal{X} \subseteq \mathbb{R}^n$ von $\bar{\mathbf{x}}$, eine offene Umgebung $\mathcal{Y} \subseteq \mathbb{R}^m$ von $\bar{\mathbf{y}}$ und eine stetig differenzierbare Funktion $\mathbf{g} : \mathcal{X} \rightarrow \mathcal{Y}$ genau so, dass

$$\mathbf{f}(\mathbf{x}, \mathbf{y}) = \mathbf{0} \quad \Leftrightarrow \quad \mathbf{y} = \mathbf{g}(\mathbf{x}) \quad \forall (\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y} \quad (1.45)$$

gilt.

Folglich gilt auch $\mathbf{f}(\mathbf{x}, \mathbf{g}(\mathbf{x})) = \mathbf{0} \quad \forall \mathbf{x} \in \mathcal{X}$. Für gegebenes $\mathbf{x}$ kann der eindeutige Wert $\mathbf{g}(\mathbf{x})$ (ohne explizite Kenntnis der Funktion $\mathbf{g}(\mathbf{x})$) durch numerisches Lösen der Gleichung

$$\mathbf{f}(\mathbf{x}, \mathbf{y}) = \mathbf{0} \quad (1.46)$$

nach $\mathbf{y}$ berechnet werden. Die Jacobi-Matrix $(\nabla \mathbf{g})(\mathbf{x}) \in \mathbb{R}^{n \times m}$ kann (ohne explizite Kenntnis der Funktion $\mathbf{g}(\mathbf{x})$) durch *implizites Ableiten* in der Form

$$(\nabla \mathbf{g})(\mathbf{x}) = - \left(\frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{g}(\mathbf{x}))}{\partial \mathbf{x}} \right)^T \left(\frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{y})}{\partial \mathbf{y}} \Big|_{\mathbf{y}=\mathbf{g}(\mathbf{x})} \right)^{-T} \quad (1.47)$$

berechnet werden.

1.3.6 Konvexität

Die Eigenschaft der Konvexität ist von großer Bedeutung in der Optimierung und erlaubt häufig eine einfache (numerische) Lösung einer Optimierungsaufgabe. Der Begriff *konvex* kann sowohl auf Mengen als auch auf Funktionen angewandt werden.

1.3.6.1 Konvexe Mengen

Definition 1.6 (Konvexe Menge). Eine Menge $\mathcal{X} \subseteq \mathbb{R}^n$ nennt man *konvex*, falls für alle $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ und alle reellen Zahlen α mit $0 < \alpha < 1$ gilt

$$(\alpha \mathbf{x} + (1 - \alpha) \mathbf{y}) \in \mathcal{X}. \quad (1.48)$$

Eine geometrische Interpretation dieser Definition ist, dass eine Menge $\mathcal{X} \subseteq \mathbb{R}^n$ genau dann konvex ist, falls die Verbindungsline zwischen zwei beliebigen Punkten $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ komplett in $\mathcal{X}$ enthalten ist. Abbildung 1.13 zeigt für den Raum $\mathbb{R}^2$ einige Beispiele konvexer und nicht-konvexer Mengen.

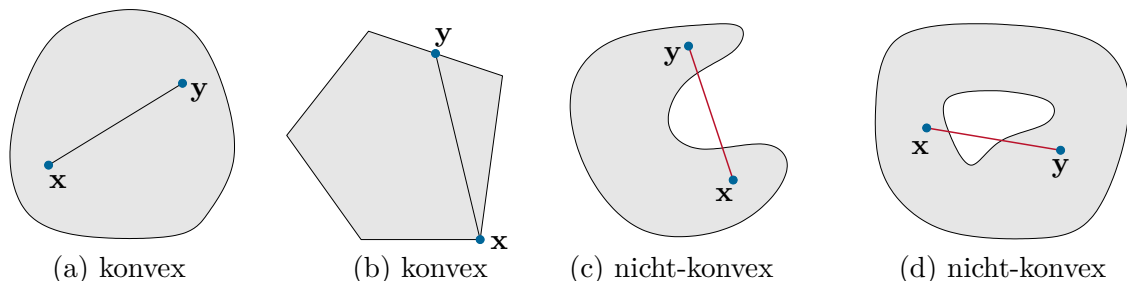


Abbildung 1.13: Beispiele von konvexen und nicht-konvexen Mengen im $\mathbb{R}^2$.

Konvexe Mengen besitzen folgende Eigenschaften:

- (a) Die *Schnittmenge* von konvexen Mengen ist wiederum konvex.
- (b) Wenn $\mathcal{X} \subseteq \mathbb{R}^n$ eine konvexe Menge, $\mathbf{A} \in \mathbb{R}^{m \times n}$ eine feste Matrix und $\mathbf{b} \in \mathbb{R}^m$ ein fester Vektor ist, dann ist die Menge

$$\{\mathbf{Ax} + \mathbf{b} \mid \mathbf{x} \in \mathcal{X}\} \subseteq \mathbb{R}^m \quad (1.49)$$

ebenfalls konvex. D. h. das Bild einer konvexen Menge unter einer *affinen Transformation* ist konvex.

- (c) Wenn $\mathcal{X}$ und $\mathcal{Y}$ konvexe Mengen sind, dann ist die Menge

$$\{\mathbf{x} + \mathbf{y} \mid \mathbf{x} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}\} \quad (1.50)$$

ebenfalls konvex.

Diese Eigenschaften sind u. a. bei der Charakterisierung der Konvexität der zulässigen Menge $\mathcal{X}$ einer Optimierungsaufgabe von Bedeutung.

1.3.6.2 Konvexe Funktionen

Definition 1.7 (Konvexe und konkave Funktionen). Es sei $\mathcal{X} \subseteq \mathbb{R}^n$ eine konvexe Menge. Man nennt die Funktion $f : \mathcal{X} \rightarrow \mathbb{R}$ *konvex* auf $\mathcal{X}$, falls für alle $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ und alle reellen Zahlen α mit $0 \leq \alpha \leq 1$ gilt

$$f(\alpha \mathbf{x} + (1 - \alpha) \mathbf{y}) \leq \alpha f(\mathbf{x}) + (1 - \alpha) f(\mathbf{y}). \quad (1.51)$$

Die Funktion f nennt man *strikt konvex*, falls für alle α mit $0 < \alpha < 1$ und $\mathbf{x} \neq \mathbf{y}$ gilt

$$f(\alpha \mathbf{x} + (1 - \alpha) \mathbf{y}) < \alpha f(\mathbf{x}) + (1 - \alpha) f(\mathbf{y}). \quad (1.52)$$

Man nennt die Funktion f (*strikt*) *konkav*, falls $-f$ (strikt) konvex ist.

Die Definition 1.7 kann wie folgt geometrisch interpretiert werden: Eine Funktion f ist konvex (konkav), falls für alle $\mathbf{x} \in \mathcal{X}$, $\mathbf{y} \in \mathcal{X}$ und $0 < \alpha < 1$ alle Funktionswerte $f(\alpha \mathbf{x} + (1 - \alpha) \mathbf{y})$ entlang der Achse f unterhalb (oberhalb) oder auf der Verbindungslinie zwischen $(\mathbf{x}, f(\mathbf{x}))$ und $(\mathbf{y}, f(\mathbf{y}))$ liegen. Abbildung 1.14 zeigt für den skalaren Fall einige Beispiele konvexer und konkaver Funktionen. Es ist direkt ersichtlich, dass affine Funktionen sowohl konkav als auch konvex sind.

Konvexe Funktionen besitzen folgende Eigenschaften:

- (a) Die Summenfunktion

$$f(\mathbf{x}) = \sum_{i=1}^k a_i f_i(\mathbf{x}) \quad (1.53)$$

von auf der konvexen Menge $\mathcal{X}$ konvexen Funktionen $f_i(\mathbf{x})$, $i = 1, \dots, k$ mit den reellen Koeffizienten $a_i \geq 0$, $i = 1, \dots, k$ ist auf $\mathcal{X}$ ebenfalls konvex.

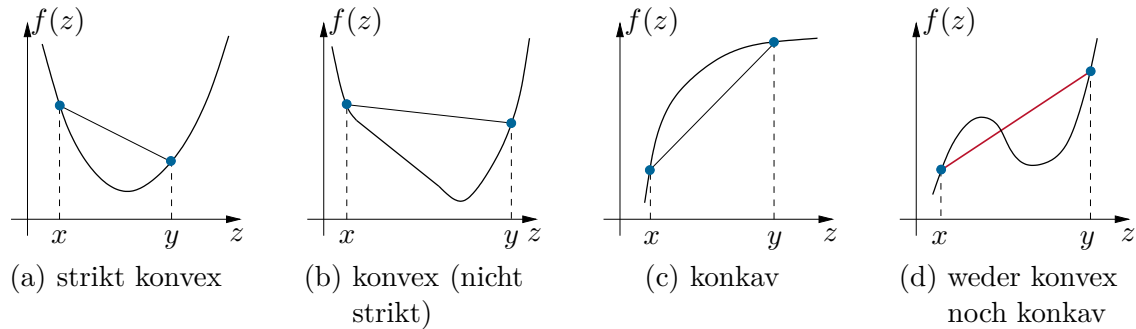


Abbildung 1.14: Beispiele von konvexen und konkaven Funktionen.

- (b) Ist die Funktion $f(\mathbf{y})$ auf der konvexen Menge $\mathcal{Y} \subseteq \mathbb{R}^m$ konvex und existieren eine konvexe Menge $\mathcal{X} \subseteq \mathbb{R}^n$, eine feste Matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ und ein fester Vektor $\mathbf{b} \in \mathbb{R}^m$ so, dass

$$\{\mathbf{Ax} + \mathbf{b} \mid \mathbf{x} \in \mathcal{X}\} \subseteq \mathcal{Y} \quad (1.54)$$

gilt, dann ist die Funktion

$$\tilde{f}(\mathbf{x}) = f(\mathbf{Ax} + \mathbf{b}) \quad (1.55)$$

konvex auf $\mathcal{X}$.

- (c) Ist die Funktion $f(\mathbf{x})$ auf der konvexen Menge $\mathcal{X}$ konvex, so ist die Menge

$$\{\mathbf{x} \in \mathcal{X} \mid f(\mathbf{x}) \leq c\} \quad (1.56)$$

für alle Werte $c \in \mathbb{R}$ ebenfalls konvex, siehe Abbildung 1.15.

- (d) Eine stetig differenzierbare Funktion $f \in C^1$ ist genau dann konvex auf der konvexen Menge $\mathcal{X}$, wenn für alle $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ die Ungleichung

$$f(\mathbf{y}) \geq f(\mathbf{x}) + (\mathbf{y} - \mathbf{x})^T (\nabla f)(\mathbf{x}) \quad (1.57)$$

erfüllt ist. Die geometrische Interpretation der Ungleichung (1.57) ist, dass an jedem Punkt $\mathbf{x}$ einer konvexen Funktion $f(\mathbf{x})$ eine sogenannte *stützende Hyperebene* (skalärer Fall: *stützende Tangente*) existieren muss, oberhalb oder auf der $f(\mathbf{x})$ verläuft. Dies ist in Abbildung 1.16 veranschaulicht.

- (e) Eine zweifach stetig differenzierbare Funktion $f \in C^2$ ist genau dann konvex auf der konvexen Menge $\mathcal{X}$, wenn die Hessematrix $(\nabla^2 f)(\mathbf{x})$ positiv semi-definit für alle $\mathbf{x} \in \mathcal{X}$ ist. Falls die Hessematrix $(\nabla^2 f)(\mathbf{x})$ positiv definit ist, so folgt daraus die strikte Konvexität der Funktion $f(\mathbf{x})$. Die Umkehrung dieser Aussage ist jedoch nicht gültig, wie man sich anhand der Funktion $f(x) = x^4$ überzeugen kann. Diese Funktion ist strikt konvex, aber die zugehörige Hessematrix an der Stelle $x = 0$ ist identisch Null.

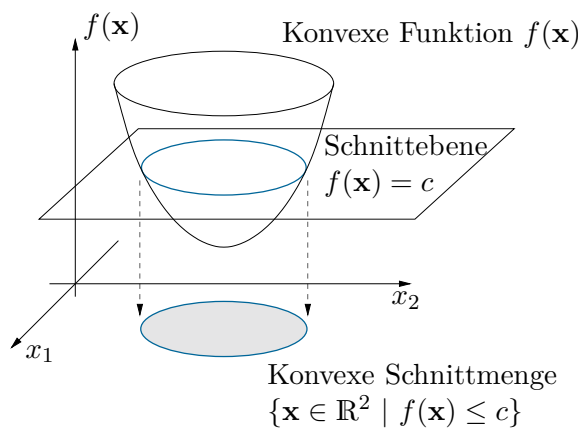


Abb. 1.15: Konvexe Menge, die durch den Schnitt einer konvexen Funktion $f(\mathbf{x})$ mit der Ebene $f(\mathbf{x}) = \text{konst.}$ entsteht.

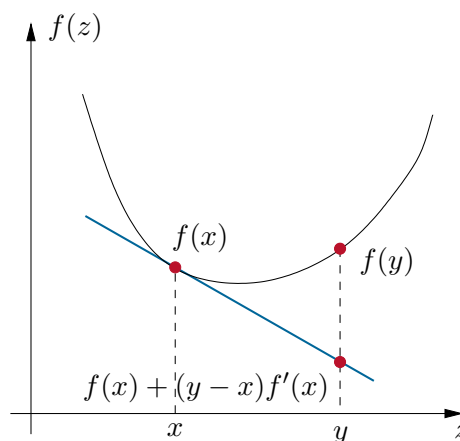


Abb. 1.16: Stützende Tangente einer konvexen Funktion $f(z)$.

Aufgabe 1.1. Beweisen Sie die Eigenschaften (a)–(e) von konvexen Funktionen. Nutzen Sie für den Beweis der Eigenschaft (e) den Mittelwertsatz, siehe Satz 1.3, im Speziellen (1.31).

Aufgabe 1.2. Zeigen Sie, dass die Funktion $f(\mathbf{x}) = x_1^4 + x_1^2 - 2x_1x_2 + x_2^2$ mit $\mathbf{x} = [x_1 \ x_2]^T \in \mathbb{R}^2$ über ihrem gesamten Definitionsbereich $\mathbb{R}^2$ konvex ist.

1.4 Literatur

- [1.1] A. E. Bryson, Jr., *Dynamic Optimization*. Addison-Wesley, 1999.
- [1.2] R. H. Goddard, „A method of reaching extreme altitudes,“ *Smithsonian Miscellaneous Collections*, Jg. 71, Nr. 2, 1919.
- [1.3] K. Pohmer, *Mikroökonomische Theorie der personellen Einkommens- und Vermögensverteilung* (Studies in Contemporary Economics). Springer, 1985, Bd. 16.
- [1.4] H. J. Oberle und R. Rosendahl, „Numerical computation of a singular-state subarc in an economic optimal control model,“ *Optimal Control Applications and Methods*, Jg. 27, Nr. 4, S. 211–235, 2006.
- [1.5] M. Bazaraa, H. Sherali und C. Shetty, *Nonlinear Programming: Theory and Algorithms*, 3. Aufl. John Wiley & Sons, 2006.
- [1.6] I. Griva, S. Nash und A. Sofer, *Linear and Nonlinear Optimization*, 2. Aufl. Society for Industrial und Applied Mathematics, 2009.
- [1.7] A. Griewank und A. Walther, *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation* (Other Titles in Applied Mathematics), 2. Aufl. Society for Industrial und Applied Mathematics, 2008.
- [1.8] D. Lynch, *Numerical Partial Differential Equations for Environmental Scientists and Engineers - A First Practical Course*. New York: Springer, 2005.
- [1.9] M. Hanke-Bourgeois, *Grundlagen der Numerischen Mathematik und des Wissenschaftlichen Rechnens*, 3. Aufl. Vieweg+Teubner, 2009.
- [1.10] P. E. Gill, W. Murray und M. H. Wright, *Practical Optimization*. Academic Press, 1981.
- [1.11] H.R. Schwarz und N. Köckler, *Numerische Mathematik*, 8. Aufl. Wiesbaden: Vieweg+Teubner, 2011.
- [1.12] J. Lyness und C. Moler, „Numerical differentiation of analytic functions,“ *SIAM Journal on Numerical Analysis*, Jg. 4, Nr. 2, S. 202–210, 1967.
- [1.13] J. Martins, P. Sturdza und J. Alonso, „The complex-step derivative approximation,“ *ACM Transactions on Mathematical Software*, Jg. 29, Nr. 3, S. 245–262, 2003.
- [1.14] R. Abreu, „Complex steps finite differences with applications to seismic problems,“ Diss., University of Granada, Granada, Spanien, 2013.
- [1.15] J. Nocedal und S. J. Wright, *Numerical Optimization* (Springer Series in Operations Research and Financial Engineering), 2. Aufl. Springer, 2006.
- [1.16] S. Boyd und L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.
- [1.17] M. Papageorgiou, M. Leibold und M. Buss, *Optimierung: Statische, dynamische, stochastische Verfahren für die Anwendung*, 4. Aufl. Springer, 2015.
- [1.18] D. P. Bertsekas, *Nonlinear Programming*, 2. Aufl. Athena Scientific, 1999.

- [1.19] D. G. Luenberger und Y. Ye, *Linear and Nonlinear Programming* (International Series in Operations Research & Management Science), 3. Aufl. Springer, 2008, Bd. 116.
- [1.20] B. C. Chachuat, „Nonlinear and Dynamic Optimization: From Theory to Practice,“ abrufbar unter <http://infoscience.epfl.ch/record/111939>, Laboratoire d'Automatique, École Polytechnique Fédérale de Lausanne, 2007. besucht am 30. Sep. 2020.

2 Statische Optimierung: Unbeschränkter Fall

In diesem Kapitel werden unbeschränkte statische Optimierungsprobleme der Art

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

betrachtet. Die Abschnitte 2.2 bis 2.6 behandeln numerische Verfahren zur Lösung solcher Optimierungsprobleme. Im nachfolgenden Abschnitt werden Optimalitätsbedingungen für das Problem (2.1) formuliert.

2.1 Optimalitätsbedingungen

Satz 2.1 (Notwendige Optimalitätsbedingung erster Ordnung). *Es sei $f \in C^1$ eine Funktion definiert auf $\mathbb{R}^n$. Wenn $\mathbf{x}^*$ ein lokales Minimum von f auf $\mathbb{R}^n$ ist, dann gilt*

$$(\nabla f)(\mathbf{x}^*) = \mathbf{0}. \quad (2.2)$$

Beweis. Man betrachte die Funktion $g(\alpha) = f(\mathbf{x}^* + \alpha \mathbf{d})$ mit einer beliebigen Richtung $\mathbf{d} \in \mathbb{R}^n$ und $\alpha \geq 0$. Für beliebige Werte $\alpha \geq 0$ gilt $\mathbf{x}^* + \alpha \mathbf{d} \in \mathbb{R}^n$, so dass die Funktion $g(\alpha)$ wohldefiniert ist. Diese Funktion muss am Punkt $\alpha = 0$ ein lokales Minimum besitzen. Entwickelt man $g(\alpha)$ um den Punkt $\alpha = 0$ in eine Taylorreihe und bricht diese nach dem linearen Glied ab, so erhält man

$$g(\alpha) = g(0) + g'(0)\alpha + \mathcal{O}(\alpha^2) \quad (2.3)$$

mit $g'(0) = \mathbf{d}^T (\nabla f)(\mathbf{x}^*)$. Der Restterm $\mathcal{O}(\alpha^2)$ klingt quadratisch nach Null ab, d. h. schneller als der lineare Term $g'(0)\alpha$. Wäre nun $g'(0) < 0$, dann würde für ein hinreichend kleines $\alpha > 0$ gelten $g(\alpha) - g(0) < 0$, was im Widerspruch zu der Annahme steht, dass $\alpha = 0$ bzw. $\mathbf{x}^*$ ein Minimum ist. Daher muss gelten $g'(0) = \mathbf{d}^T (\nabla f)(\mathbf{x}^*) \geq 0$. Dies kann aber nur dann für beliebige $\mathbf{d} \in \mathbb{R}^n$ erfüllt sein, wenn $(\nabla f)(\mathbf{x}^*) = \mathbf{0}$ gilt. $\square$

Beispiel 2.1. Man betrachte das Optimierungsproblem

$$\min_{\mathbf{x} \in \mathbb{R}^2} f(x_1, x_2) = x_1^2 - x_1 x_2 + x_2^2 - 3x_2. \quad (2.4)$$

Aus den notwendigen Optimalitätsbedingungen erster Ordnung gemäß (2.2)

$$2x_1 - x_2 = 0 \quad (2.5a)$$

$$-x_1 + 2x_2 = 3, \quad (2.5b)$$

folgt die Lösung $\mathbf{x}^* = [1 \ 2]^T$. Es kann mit Hilfe des nachfolgenden Satzes 2.3 gezeigt werden, dass $\mathbf{x}^*$ in diesem Fall sogar ein striktes globales Minimum ist.

Die Optimalitätsbedingung gemäß Satz 2.1 ist notwendig aber nicht hinreichend. Die Bedingung gibt lediglich an, dass es sich bei dem betreffenden Punkt um einen *Extremalpunkt* (auch als *stationärer Punkt* bezeichnet) handelt, und wird von einem Minimum, Maximum oder Sattelpunkt gleichermaßen erfüllt, siehe Abbildung 2.1.

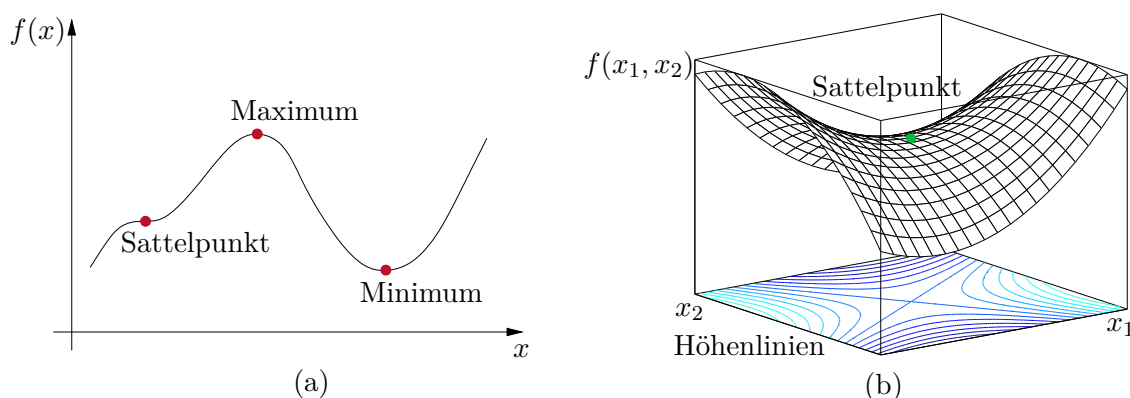


Abbildung 2.1: Beispiele von stationären Punkten im ein- und zweidimensionalen Fall.

Satz 2.2 (Notwendige Optimalitätsbedingungen zweiter Ordnung). Es sei $f \in C^2$ eine Funktion definiert auf $\mathbb{R}^n$. Wenn $\mathbf{x}^*$ ein lokales Minimum von f auf $\mathbb{R}^n$ ist, dann gelten die Bedingungen

$$(a) \quad (\nabla f)(\mathbf{x}^*) = \mathbf{0} \quad (2.6a)$$

$$(b) \quad (\nabla^2 f)(\mathbf{x}^*) \text{ ist positiv semi-definit.} \quad (2.6b)$$

Aufgabe 2.1. Beweisen Sie Satz 2.2. **Hinweis:** Orientieren Sie sich dabei am Beweis von Satz 2.1.

Auch Satz 2.2 beschreibt lediglich notwendige Optimalitätsbedingungen, wie man sich einfach anhand der Funktion $f(x) = x^3$ überzeugen kann. Diese Funktion besitzt an der Stelle $x^* = 0$ einen Extremalpunkt ($f'(x^*) = 3(x^*)^2 = 0$) und obwohl die zweite Ableitung $f''(x^*) = 6x^* = 0$ positiv semi-definit ist, ist $x^* = 0$ kein Minimum. Die Funktion hat an der Stelle $x^* = 0$ einen Sattelpunkt.

Es können nun folgende hinreichende Optimalitätsbedingungen angegeben werden.

Satz 2.3 (Hinreichende Optimalitätsbedingungen zweiter Ordnung). *Es sei $f \in C^2$ eine Funktion definiert auf $\mathbb{R}^n$. Wenn die Bedingungen*

$$(a) \quad (\nabla f)(\mathbf{x}^*) = \mathbf{0} \quad (2.7a)$$

$$(b) \quad (\nabla^2 f)(\mathbf{x}^*) \text{ ist positiv definit} \quad (2.7b)$$

erfüllt sind, dann ist $\mathbf{x}^*$ ein striktes lokales Minimum von f auf $\mathbb{R}^n$.

Aufgabe 2.2. Beweisen Sie Satz 2.3.

Beispiel 2.2. Für das Optimierungsproblem

$$\min_{\mathbf{x} \in \mathbb{R}^2} f(\mathbf{x}) = x_1^2 + ax_2^2 - x_1x_2 \quad (2.8)$$

sollen die stationären Werte $\mathbf{x}^*$ in Abhängigkeit des Parameters $a \neq \frac{1}{4}$ charakterisiert werden. Der Gradient und die Hessematrix von $f(\mathbf{x})$ ergeben sich zu

$$(\nabla f)(\mathbf{x}) = \begin{bmatrix} 2x_1 - x_2 \\ 2ax_2 - x_1 \end{bmatrix}, \quad (\nabla^2 f)(\mathbf{x}) = \begin{bmatrix} 2 & -1 \\ -1 & 2a \end{bmatrix}. \quad (2.9)$$

Aus $(\nabla f)(\mathbf{x}^*) = \mathbf{0}$ folgt $\mathbf{x}^* = [0 \ 0]^T$ als einziger stationärer Punkt. Die Definitheit der Hessematrix $(\nabla^2 f)(\mathbf{x})$ an der Stelle $\mathbf{x}^*$ lässt sich mit Hilfe der Hauptminoren (Sylvesterkriterium, siehe (c) in Satz 1.2) untersuchen

$$D_1 = 2, \quad D_2 = 4a - 1. \quad (2.10)$$

Somit ist $(\nabla^2 f)(\mathbf{x}^*)$ positiv definit für $a > \frac{1}{4}$ und $\mathbf{x}^* = [0 \ 0]^T$ stellt ein striktes Minimum dar. Für $a < \frac{1}{4}$ ist $D_1 > 0$ und $D_2 < 0$ und $(\nabla^2 f)(\mathbf{x})$ somit *indefinit*. In diesem Fall ist $\mathbf{x}^* = [0 \ 0]^T$ ein *Sattelpunkt*, ähnlich wie er in Abbildung 2.1(b) für $a = -1$ dargestellt ist.

Die Sätze 2.1 bis 2.3 liefern nur Aussagen zu lokalen Minima. Wenn die Funktion $f(\mathbf{x})$ konvex oder strikt konvex ist, dann können nachfolgende vereinfachte Optimalitätsbedingungen angegeben werden.

Satz 2.4 (Hinreichende Optimalitätsbedingung für konvexe Funktion). *Es sei $f \in C^1$ eine konvexe Funktion definiert auf $\mathbb{R}^n$. Wenn die Bedingungen*

$$(\nabla f)(\mathbf{x}^*) = \mathbf{0} \quad (2.11)$$

erfüllt ist, dann ist $\mathbf{x}^*$ ein globales Minimum von f auf $\mathbb{R}^n$.

Beweis. Für beliebiges $\mathbf{y} \in \mathbb{R}^n$ gilt wegen der Konvexität von $f(\mathbf{x})$ mit dem Minimum

$\mathbf{x}^*$ die Ungleichung

$$f(\mathbf{y}) \geq f(\mathbf{x}^*) + (\mathbf{y} - \mathbf{x}^*)^T \underbrace{(\nabla f)(\mathbf{x}^*)}_{=0} = f(\mathbf{x}^*) . \quad (2.12)$$

□

Folglich ist bei konvexen Funktionen $f(\mathbf{x})$ die notwendige Optimalitätsbedingung erster Ordnung gemäß Satz 2.1 auch *hinreichend*.

Satz 2.5 (Globale Minima einer konvexen Funktion). *Es sei $f(\mathbf{x})$ eine konvexe Funktion auf $\mathbb{R}^n$. Die Menge aller Minima $\mathcal{G} = \arg \min \{f(\mathbf{x}) \mid \mathbf{x} \in \mathbb{R}^n\}$ ist konvex. Jedes lokale Minimum $\mathbf{x}^* \in \mathcal{G}$ von f ist auch ein globales Minimum. Ist $f(\mathbf{x})$ strikt konvex, so ist $\mathbf{x}^*$ ein striktes globales Minimum.*

Beweis. Angenommen c beschreibt den minimalen Wert von f . Dann ist die Menge $\mathcal{G} = \{\mathbf{x} \mid \mathbf{x} \in \mathbb{R}^n, f(\mathbf{x}) \leq c\}$ gemäß (1.56) konvex, womit der erste Teil des Satzes gezeigt ist.

Der zweite Teil des Satzes kann mittels Beweis durch Widerspruch gezeigt werden. Angenommen $\mathbf{x}^*$ ist ein lokales Minimum aber kein globales Minimum von f auf $\mathbb{R}^n$. Dann existiert ein Punkt $\mathbf{y} \in \mathbb{R}^n$, der die Ungleichung $f(\mathbf{y}) < f(\mathbf{x}^*)$ erfüllt. Gemäß der Definition 1.7 für konvexe Funktionen gilt dann für alle $\alpha \in [0, 1]$

$$f(\alpha \mathbf{x}^* + (1 - \alpha)\mathbf{y}) \leq \alpha f(\mathbf{x}^*) + (1 - \alpha)f(\mathbf{y}) < \alpha f(\mathbf{x}^*) + (1 - \alpha)f(\mathbf{x}^*) = f(\mathbf{x}^*) . \quad (2.13)$$

Dies zeigt, dass mit $\alpha \rightarrow 1$ Punkte $\alpha \mathbf{x}^* + (1 - \alpha)\mathbf{y}$ gefunden werden können, die beliebig nahe bei $\mathbf{x}^*$ liegen, deren Funktionswerte $f(\alpha \mathbf{x}^* + (1 - \alpha)\mathbf{y})$ aber strikt kleiner sind als $f(\mathbf{x}^*)$. Dies verletzt die Definition eines lokalen Minimums an der Stelle $\mathbf{x}^*$ und steht daher im Widerspruch zu Annahme. Folglich kann kein Punkt $\mathbf{y} \in \mathbb{R}^n$ existieren, der die Ungleichung $f(\mathbf{y}) < f(\mathbf{x}^*)$ erfüllt und $\mathbf{x}^*$ ist ein globales Minimum.

In ähnlicher Weise kann der dritte Teil des Satzes gezeigt werden. Die Annahme, dass $\mathbf{x}^*$ kein striktes globales Minimum ist, also ein Punkt $\mathbf{y} \in \mathbb{R}^n$ mit $\mathbf{y} \neq \mathbf{x}^*$ existiert, der die Gleichung $f(\mathbf{y}) = f(\mathbf{x}^*)$ erfüllt, führt für eine strikt konvexe Funktion $f(\mathbf{x})$ auf einen Widerspruch. □

2.2 Rechnergestützte Minimierungsverfahren: Grundlagen

2.2.1 Iterative Abstiegsverfahren

Da die Bestimmung eines (lokal) optimalen Punktes $\mathbf{x}^*$ von (2.1) durch analytische Lösung der Stationaritätsbedingung $(\nabla f)(\mathbf{x}^*) = \mathbf{0}$ von (2.7a) (n nichtlineare Gleichungen in $\mathbf{x}^*$) nur in seltenen Fällen möglich ist, ist man im Allgemeinen auf *numerische Verfahren* zur Suche von $\mathbf{x}^*$ angewiesen. Viele der in dieser Vorlesung besprochenen Algorithmen finden den exakten Punkt $\mathbf{x}^*$ nicht direkt oder in einer endlichen Anzahl von Rechenschritten, sondern generieren eine Folge $\{\mathbf{x}_k\}$, entlang welcher die zu optimierende Funktion $f(\mathbf{x})$ abnimmt, d. h.

$$f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k), \quad k = 0, 1, 2, \dots, \quad (2.14)$$

und die zumindest für $k \rightarrow \infty$ gegen $\mathbf{x}^*$ konvergieren soll, d. h.

$$\lim_{k \rightarrow \infty} \mathbf{x}_k = \mathbf{x}^*. \quad (2.15)$$

Solche Algorithmen werden auch als *iterative Abstiegsverfahren* (englisch: iterative descent methods) bezeichnet und (2.14) als *Abstiegsbedingung*. Neben der anhand von (2.15) zu beantwortenden Frage, ob ein Algorithmus prinzipiell gegen die richtige Lösung $\mathbf{x}^*$ konvergiert, interessiert, wie rasch er dies tut. Diese Frage wird im nachfolgenden Abschnitt behandelt.

2.2.2 Konvergenzverhalten

Es wird das Konvergenzverhalten von iterativen Abstiegsverfahren analysiert, welche eine Folge $\{\mathbf{x}_k\}$ liefern. Erreicht diese Folge einen optimalen Punkt $\mathbf{x}^*$ bereits nach einer endlichen Anzahl an Rechenschritten, so besitzt das Verfahren *endliche Konvergenz*. Erreicht diese Folge einen optimalen Punkt $\mathbf{x}^*$ erst für $k \rightarrow \infty$, so besitzt das Verfahren *asymptotische Konvergenz*, welche nachfolgend vorrangig betrachtet wird. Selbst bei Verfahren mit formal endlicher Konvergenz kann eine Untersuchung des asymptotischen Konvergenzverhaltens von Interesse sein, da Rechenungenauigkeiten eine exakte Erreichung des optimalen Punktes $\mathbf{x}^*$ nach einer endlichen Anzahl an Rechenschritten verhindern können.

Zumeist basiert die Analyse des Konvergenzverhaltens auf einer *Fehlerfunktion* $e : \mathbb{R}^n \rightarrow \mathbb{R}$, welche $e(\mathbf{x}) \geq 0$ für alle $\mathbf{x} \in \mathbb{R}^n$ und $e(\mathbf{x}^*) = 0$ erfüllt. Als Fehlerfunktion kann z. B. der Abstand

$$e(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}^*\| \geq 0 \quad (2.16a)$$

im Sinne einer Norm $\|\cdot\|$ oder die Kostendifferenz

$$e(\mathbf{x}) = f(\mathbf{x}) - f(\mathbf{x}^*) \geq 0 \quad (2.16b)$$

verwendet werden [2.1]. Das asymptotische Konvergenzverhalten eines Algorithmus kann anhand der zu $\{\mathbf{x}_k\}$ gehörenden Folge $\{e_k\}$ mit $e_k = e(\mathbf{x}_k)$ analysiert werden. Zunächst sollen dazu die Begriffe *Konvergenzordnung* und *Konvergenzrate* einer Folge von Skalaren definiert werden.

Definition 2.1 (Konvergenzordnung, Konvergenzrate). Es sei $\{e_k\}$ eine Folge von Skalaren, die asymptotisch gegen den Grenzwert 0 konvergiert. Die *Konvergenzordnung* der Folge $\{e_k\}$ ist die kleinste obere Schranke (Supremum) der Menge von nichtnegativen Zahlen p , für die gilt

$$0 \leq \lim_{k \rightarrow \infty} \frac{|e_{k+1}|}{|e_k|^p} = \mu < \infty. \quad (2.17)$$

Als zugehörige *Konvergenzrate* bezeichnet man die Zahl μ . Es werden folgende Fälle unterschieden:

- Im Fall $p = 1$ und $\mu \in (0, 1)$ spricht man von *linearer* Konvergenz.
- Im Fall $p > 1$ mit $\mu > 0$ oder $p = 1$ mit $\mu = 0$ spricht man von *superlinearer* Konvergenz.
- Im Fall $p = 2$ mit $\mu > 0$ spricht man von *quadratischer* Konvergenz.
- Im Fall $p = 3$ mit $\mu > 0$ spricht man von *kubischer* Konvergenz.

Im Wesentlichen beschreiben die Konvergenzordnung und die Konvergenzrate das (lokale) Verhalten einer Folge für $k \rightarrow \infty$. Größere Werte der Konvergenzordnung p bedeuten, dass die Folge schneller konvergiert, da die Folgeelemente e_k (zumindest für sehr große Werte von k) mit der p -ten Potenz abnehmen. Analoges gilt für kleinere Werte der Konvergenzrate μ .

Beispiel 2.3. Die Folge $\{a^k\}$ mit $0 < a < 1$ konvergiert mit der Konvergenzordnung $p = 1$ und der Konvergenzrate $\mu = a$ gegen Null. Zunächst gilt, dass nur für $p \leq 1$ die Bedingung

$$\lim_{k \rightarrow \infty} \frac{a^{k+1}}{a^{kp}} = \lim_{k \rightarrow \infty} a^{1+k(1-p)} < \infty \quad (2.18)$$

erfüllt ist. Mit $p = 1$ folgt dann

$$\lim_{k \rightarrow \infty} \frac{a^{k+1}}{a^k} = a = \mu \quad (2.19)$$

für die Konvergenzrate μ .

Aufgabe 2.3. Zeigen Sie, dass die Folge $\{a^{2^k}\}$ mit $0 < a < 1$ mit der Konvergenzordnung 2 und der Konvergenzrate 1 gegen 0 konvergiert.

Beispiel 2.4. Die Folge $\{\frac{1}{k^k}\}$ hat die Konvergenzordnung 1, da nur für $p \leq 1$ die Bedingung

$$\lim_{k \rightarrow \infty} \frac{k^{kp}}{(k+1)^{k+1}} = \lim_{k \rightarrow \infty} \frac{k}{k+1} \left(\frac{k}{k+1} \right)^k k^{k(p-1)-1} = \lim_{k \rightarrow \infty} \frac{1}{e} k^{k(p-1)-1} < \infty \quad (2.20)$$

erfüllt ist. Mit $p = 1$ ergibt sich dann

$$\lim_{k \rightarrow \infty} \frac{k^k}{(k+1)^{k+1}} = \lim_{k \rightarrow \infty} \frac{1}{k+1} \left(\frac{k}{k+1} \right)^k = 0 \frac{1}{e} = \mu = 0. \quad (2.21)$$

Folglich konvergiert die Folge $\{\frac{1}{k^k}\}$ superlinear gegen Null.

Abschließend stellt sich die Frage, ob das beobachtete asymptotische Konvergenzverhalten eines Optimierungsalgorithmus von der gewählten Fehlerfunktion $e(\mathbf{x})$ abhängt. Es lässt sich zeigen (vgl. [2.2]), dass die Konvergenzordnung eines Optimierungsalgorithmus von der Wahl der Fehlerfunktion $e(\mathbf{x})$ weitgehend unabhängig ist. Dies gilt nicht für die

Konvergenzrate.

2.2.3 Abbruchkriterien

Bei iterativen Abstiegsverfahren muss entschieden werden, wann eine gefundene Lösung ausreichend genau ist und die Iteration abgebrochen wird. Für diese Entscheidung verwenden die Algorithmen *Abbruchkriterien* [2.1, 2.3, 2.4]. Einige Beispiele dafür sind nachfolgend angeführt, wobei meist die erstmalige Erfüllung eines Kriteriums zum Abbruch des Algorithmus führt.

- Änderung des Kostenfunktionswertes

$$f(\mathbf{x}_{k-1}) - f(\mathbf{x}_k) \leq \varepsilon_f^{\text{abs}} + \varepsilon_f^{\text{rel}} |f(\mathbf{x}_{k-1})| \quad (2.22)$$

- Änderung des aktuellen Punktes

$$\|\mathbf{x}_k - \mathbf{x}_{k-1}\| \leq \varepsilon_x^{\text{abs}} + \varepsilon_x^{\text{rel}} \|\mathbf{x}_{k-1}\| \quad (2.23)$$

- Wert des Gradienten (siehe notwendige Optimalitätsbedingung erster Ordnung gemäß Satz 2.1)

$$\|(\nabla f)(\mathbf{x}_k)\| \leq \varepsilon_{\nabla f}^{\text{abs}} + \varepsilon_{\nabla f}^{\text{rel}} \|(\nabla f)(\mathbf{x}_0)\| \quad (2.24)$$

- Anzahl der Iterationen

$$k \geq k_{\max} \quad (2.25)$$

- Anzahl der Funktionsaufrufe

$$\text{Anzahl der Auswertungen von } f(\mathbf{x}) \geq n_{\max} \quad (2.26)$$

- Benötigte Rechenzeit

$$t_{\text{CPU}} \geq t_{\text{CPU},\max} \quad (2.27)$$

Die letzten drei Kriterien führen zu einem vorzeitigen, erfolglosen Abbruch des Algorithmus. Die skalaren Parameter $\varepsilon_f^{\text{abs}}, \varepsilon_x^{\text{abs}}, \varepsilon_{\nabla f}^{\text{abs}} > 0$, $\varepsilon_f^{\text{rel}}, \varepsilon_x^{\text{rel}}, \varepsilon_{\nabla f}^{\text{rel}} \geq 0$ und $k_{\max}, n_{\max}, t_{\text{CPU},\max} \gg 0$ müssen entsprechend gewählt werden.

Die bekanntesten numerischen Verfahren zur Lösung der unbeschränkten statischen Optimierungsaufgabe (2.1) sind die so genannten *Liniensuchverfahren* (englisch: *line search methods*). Der folgende Abschnitt gibt einen kurzen Überblick über gängige Liniensuchverfahren. Im Anschluss daran werden mit der *Methode der Vertrauensbereiche* und dem *direkten Suchverfahren* zwei alternative Lösungsmethoden für unbeschränkte statische Optimierungsaufgaben vorgestellt.

```

Initialisierung:   $\mathbf{x}_0$       (Startlösung)
                    $k = 0$     (Startindex)

while   $\mathbf{x}_k$  ist nicht optimal
    Wähle geeignete Suchrichtung  $\mathbf{s}_k$ 
    Wähle optimale Schrittweite gemäß
         $\alpha_k = \arg \min_{\alpha > 0} f(\mathbf{x}_k + \alpha \mathbf{s}_k)$ 
     $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \alpha_k \mathbf{s}_k$ 
     $k \leftarrow k + 1$ 

end

```

Tabelle 2.1: Genereller Ablauf eines Liniensuchverfahrens.

2.3 Liniensuchverfahren

Tabelle 2.1 zeigt die grundsätzliche algorithmische Struktur eines Liniensuchverfahrens. Zum Iterationsschritt k ermittelt man vorerst eine geeignete *Suchrichtung* bzw. *Abstiegsrichtung* $\mathbf{s}_k$. Sie soll so gewählt werden, dass, wenn man sich hinreichend wenig vom Punkt $\mathbf{x}_k$ aus in diese Richtung bewegt, also

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{s}_k \quad (2.28)$$

mit einer geeigneten *Schrittweite* $\alpha_k > 0$, die Abstiegsbedingung (2.14), d. h.

$$f(\mathbf{x}_{k+1}) = f(\mathbf{x}_k + \alpha_k \mathbf{s}_k) < f(\mathbf{x}_k) \quad (2.29)$$

erfüllt ist. Nun wird die optimale Schrittweite $\alpha_k > 0$ durch Lösung des *skalaren Optimierungsproblems*

$$\min_{\alpha_k > 0} g(\alpha_k) = f(\mathbf{x}_k + \alpha_k \mathbf{s}_k) \quad (2.30)$$

bestimmt. Die Iteration wird solange wiederholt, bis ein Abbruchkriterium erfüllt ist.

Abbildung 2.2 veranschaulicht das Prinzip der Liniensuche anhand von einem Iterationsschritt für eine (nicht konvexe) Kostenfunktion $f(\mathbf{x})$ mit $\mathbf{x} \in \mathbb{R}^2$ bei einer gegebenen Suchrichtung $\mathbf{s}_k$. In diesem Zusammenhang wird auch der Name *Liniensuchverfahren* verständlich, da sich bei gegebener Suchrichtung $\mathbf{s}_k$ die Optimierungsaufgabe, d. h. die Wahl der Schrittweite α_k , auf das Auffinden eines Minimums entlang einer Linie reduziert.

2.3.1 Wahl der Schrittweite

2.3.1.1 Intervallschachtelungsverfahren („Goldener Schnitt“)

Das *Intervallschachtelungsverfahren* generiert für das skalare Optimierungsproblem (2.30) eine konvergierende Folge von Intervallschachtelungen, um das Minimum von $g(\alpha_k)$ einzugrenzen.

Zunächst muss ein Intervall $[l_0, r_0]$ gefunden werden, in dem die Funktion $g(\alpha_k)$ genau ein Minimum aufweist, siehe Abbildung 2.3. Dies kann z. B. dadurch erreicht werden,

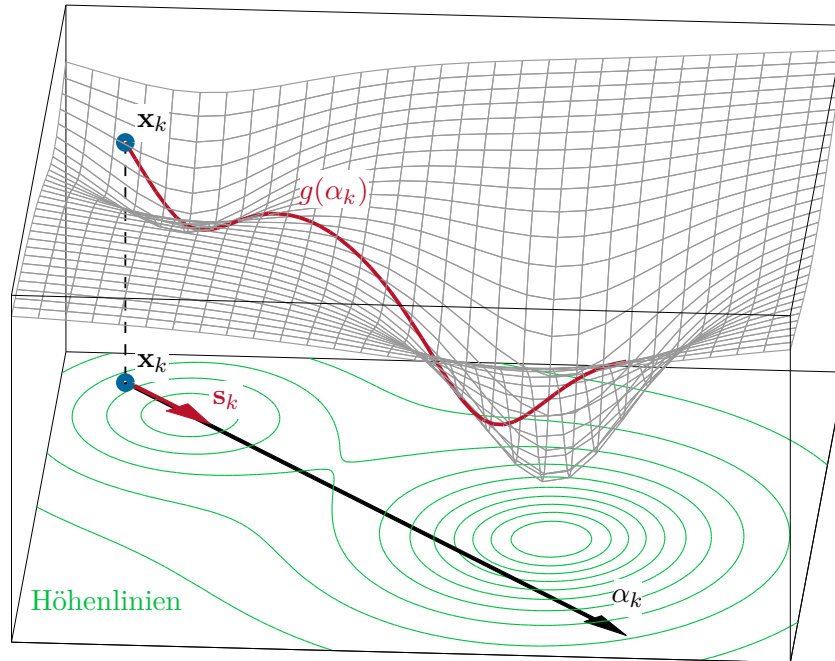


Abbildung 2.2: Veranschaulichung der Wahl der Schrittweite gemäß (2.30).

dass mit einem hinreichend kleinen l_0 gestartet und r_0 (ausgehend von l_0) sukzessive vergrößert wird, bis der Funktionswert $g(r_0)$ anfängt zuzunehmen. Für das Folgende wird vorausgesetzt, dass die Funktion $g(\alpha_k)$ im Intervall $[l_0, r_0]$ stetig ist und genau ein eindeutiges Minimum (ein striktes Minimum und keine weiteren lokalen Minima) besitzt.

Zum Iterationsschritt j liege das Intervall $[l_j, r_j]$ vor, das nach wie vor jenen Wert α_k^* beinhaltet, der die Funktion $g(\alpha_k)$ minimiert. Nun werden zwei neue Punkte $\hat{l}_j$ und $\hat{r}_j$, $l_j < \hat{l}_j < \hat{r}_j < r_j$ in der Form

$$\hat{l}_j = l_j + (1 - a)(r_j - l_j) = al_j + (1 - a)r_j \quad (2.31a)$$

$$\hat{r}_j = l_j + a(r_j - l_j) = (1 - a)l_j + ar_j \quad (2.31b)$$

mit dem Parameter $a \in (\frac{1}{2}, 1)$ berechnet. Es gilt nun folgender Satz:

Satz 2.6 (Zur Intervallschachtelung). *Es sei $l_j < \hat{l}_j < \hat{r}_j < r_j$ und $g(\alpha_k)$ eine stetige Funktion mit genau einem eindeutigen Minimum auf dem Intervall $[l_j, r_j]$. Bezeichnet man dieses Minimum mit $\alpha_k^* \in (l_j, r_j)$, dann gilt $\alpha_k^* \in [l_j, \hat{r}_j]$, wenn $g(\hat{l}_j) \leq g(\hat{r}_j)$ bzw. $\alpha_k^* \in [\hat{l}_j, r_j]$, wenn $g(\hat{l}_j) \geq g(\hat{r}_j)$.*

Beweis. Man betrachte den Fall $g(\hat{l}_j) \leq g(\hat{r}_j)$. Angenommen, $\alpha_k^* > \hat{r}_j$, dann gilt $\hat{l}_j < \alpha_k^*$. Da $g(\hat{l}_j) \leq g(\hat{r}_j)$ und $g(\alpha_k^*) \leq g(\hat{r}_j)$ ist, muss ein Punkt $\bar{\alpha}_k \in (\hat{l}_j, \alpha_k^*)$ so existieren, dass gilt $g(\bar{\alpha}_k) = \max_{\alpha_k \in [\hat{l}_j, \alpha_k^*]} g(\alpha_k)$, womit $\bar{\alpha}_k$ ein lokales Maximum von $g(\alpha_k)$ im Intervall $[l_j, r_j]$ beschreibt. Die Existenz eines lokalen Maximums ist aber

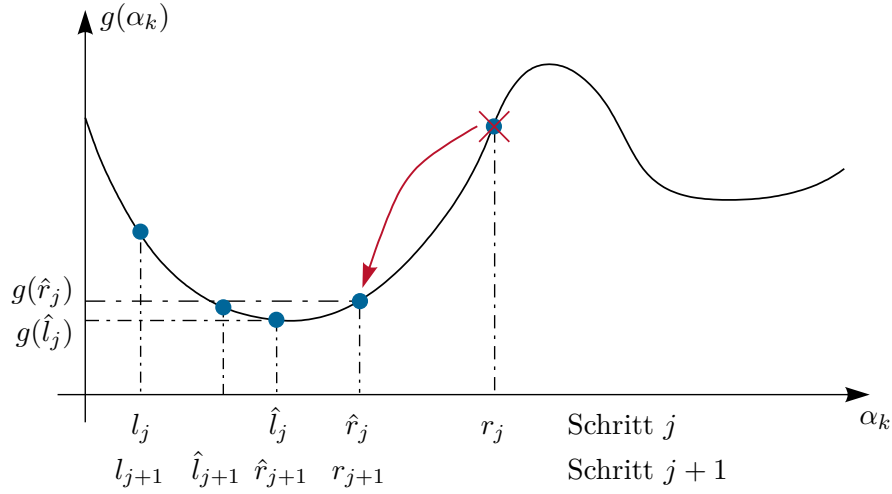


Abbildung 2.3: Veranschaulichung des Intervallschachtelungsverfahrens nach dem Prinzip des „Goldenen Schnittes“.

aufgrund der Forderung genau eines eindeutigen Minimums von $g(\alpha_k)$ nicht möglich. Für $g(\hat{l}_j) \geq g(\hat{r}_j)$ folgt der Beweis auf analoge Art und Weise. $\square$

Gemäß Satz 2.6 wird zum nächsten Iterationsschritt $j + 1$ für den Fall $g(\hat{l}_j) \leq g(\hat{r}_j)$ der äußere Punkt r_j verworfen und das Intervall ergibt sich demnach zu $[l_{j+1}, r_{j+1}]$ mit $l_{j+1} = l_j$, $r_{j+1} = \hat{r}_j$, siehe Abbildung 2.3. Für $g(\hat{l}_j) \geq g(\hat{r}_j)$ folgt das Intervall zum Iterationsschritt $j + 1$ zu $[l_{j+1}, r_{j+1}]$ mit $l_{j+1} = \hat{l}_j$, $r_{j+1} = r_j$.

Für die weitere Betrachtung nehme man an, dass, wie in Abbildung 2.3 dargestellt, $g(\hat{l}_j) \leq g(\hat{r}_j)$ ist. Die nachfolgenden Schritte lassen sich direkt auf den anderen Fall übertragen. Man führt zunächst mit $l_{j+1} = l_j$ und $r_{j+1} = \hat{r}_j = (1 - a)l_j + ar_j$ eine weitere Iteration zur Berechnung der Zwischenpunkte gemäß (2.31) in der Form

$$\hat{l}_{j+1} = al_{j+1} + (1 - a)r_{j+1} = (1 - a + a^2)l_j + (1 - a)ar_j \quad (2.32a)$$

$$\hat{r}_{j+1} = (1 - a)l_{j+1} + ar_{j+1} = (1 - a^2)l_j + a^2r_j \quad (2.32b)$$

durch. Aus einem Koeffizientenvergleich von (2.31a) und (2.32b) folgt, dass $\hat{r}_{j+1} = \hat{l}_j$ genau dann erreicht wird, wenn $a^2 = 1 - a$ gilt, d. h. wenn

$$a = \frac{\sqrt{5} - 1}{2} \approx 0.618. \quad (2.33)$$

Diese Wahl von a hat den Vorteil, dass je Iteration nur ein neuer Zwischenpunkt berechnet werden muss. Man beachte, dass die Berechnung jedes Zwischenpunktes mit einer Auswertung der Kostenfunktion $g(\alpha_k)$ verbunden ist. Die Zahl $1/a = 1 + a \approx 1.618$ ist bekannt als die Verhältniszahl des *Goldenen Schnittes*. Tabelle 2.2 fasst den Algorithmus nochmals zusammen.

Abschließend kann der optimale Wert α_k^* entweder aus der *Mittelung* der letzten Intervallgrenzen $\alpha_k^* = (l_j + r_j)/2$ oder aus einer *quadratischen Interpolation* (siehe nachfolgender

Abschnitt) zwischen den kleinsten drei Funktionswerten gewonnen werden. Das Intervallschachtelungsverfahren ist ein *einfaches* und *robustes* Verfahren, das allerdings im Vergleich zu anderen Verfahren meist mehr Iterationen benötigt.

Initialisierung:	l_0, r_0	(Startintervall mit Minimum im Inneren)
	$j = 0$	(Startindex)
	$a = 0.618$	(Goldener Schnitt-Parameter)
	ε_{lr}	(Schranke für Abbruch)
	$\hat{l}_0 \leftarrow al_0 + (1 - a)r_0$	(innere Punkte)
	$\hat{r}_0 \leftarrow (1 - a)l_0 + ar_0$	

repeat

if $g(\hat{l}_j) > g(\hat{r}_j)$

$l_{j+1} \leftarrow \hat{l}_j$

$r_{j+1} \leftarrow r_j$

$\hat{l}_{j+1} \leftarrow \hat{r}_j$

$\hat{r}_{j+1} \leftarrow (1 - a)l_{j+1} + ar_{j+1}$

else (d. h. $g(\hat{l}_j) \leq g(\hat{r}_j)$)

$r_{j+1} \leftarrow \hat{r}_j$

$l_{j+1} \leftarrow l_j$

$\hat{r}_{j+1} \leftarrow \hat{l}_j$

$\hat{l}_{j+1} \leftarrow al_{j+1} + (1 - a)r_{j+1}$

end if

$j \leftarrow j + 1$

until $|r_j - l_j| \leq \varepsilon_{lr}$

$\alpha_k^* = (l_j + r_j)/2$

Tabelle 2.2: Intervallschachtelungsverfahren („Goldener Schnitt“).

2.3.1.2 Quadratische Interpolation

Eine sehr effiziente Methode zur Lösung des skalaren Optimierungsproblems (2.30) besteht darin, durch drei Punkte eine quadratische Interpolationsfunktion zu legen. Dieser Ansatz wird gelegentlich auch als Newton-Methode bezeichnet. Es wird angenommen, dass die voneinander paarweise verschiedenen Punkte $\alpha_{k,1}$, $\alpha_{k,2}$ und $\alpha_{k,3}$ sowie deren Funktionswerte $g_1 = g(\alpha_{k,1})$, $g_2 = g(\alpha_{k,2})$ und $g_3 = g(\alpha_{k,3})$ bekannt sind. Gemäß Lagrangescher Interpolationsformel lautet die quadratische Interpolationsfunktion $q(\alpha_k)$ durch diese Punkte dann

$$q(\alpha_k) = \sum_{i=1}^3 \left(g_i \prod_{j \neq i} \frac{\alpha_k - \alpha_{k,j}}{\alpha_{k,i} - \alpha_{k,j}} \right) \quad (2.34)$$

und der optimale Wert α_k^* folgt in der Form

$$\alpha_k^* = \frac{1}{2} \frac{g_1(\alpha_{k,2}^2 - \alpha_{k,3}^2) + g_2(\alpha_{k,3}^2 - \alpha_{k,1}^2) + g_3(\alpha_{k,1}^2 - \alpha_{k,2}^2)}{g_1(\alpha_{k,2} - \alpha_{k,3}) + g_2(\alpha_{k,3} - \alpha_{k,1}) + g_3(\alpha_{k,1} - \alpha_{k,2})}. \quad (2.35)$$

Der so errechnete Wert α_k^* sollte als optimale Schrittweite nur akzeptiert werden, wenn die Interpolationsfunktion $q(\alpha_k)$ strikt konvex ist (nur dann ist α_k^* tatsächlich ein Minimum von $q(\alpha_k)$) und wenn die Bedingungen $\alpha_k^* > 0$, $g(\alpha_k^*) \leq g_1$, $g(\alpha_k^*) \leq g_2$, $g(\alpha_k^*) \leq g_3$ und $g(\alpha_k^*) \leq g(0)$ gelten.

Aufgabe 2.4. Zeigen Sie die Gültigkeit von (2.35).

2.3.1.3 Heuristische Wahl der Schrittweite

In der Praxis muss bei der Wahl der Schrittweite häufig ein Kompromiss zwischen numerischem Aufwand und erreichter Genauigkeit in Kauf genommen werden. Ungenauigkeiten treten z. B. auf, wenn ein iterativer Algorithmus zur Bestimmung der optimalen Schrittweite vorzeitig abgebrochen wird. Es gibt nun unterschiedliche *heuristische Abbruchkriterien*, die im Folgenden kurz erläutert werden. Den weiteren Betrachtungen liege das *skalare Optimierungsproblem*, siehe (2.30),

$$\min_{\alpha_k > 0} g(\alpha_k) = f(\mathbf{x}_k + \alpha_k \mathbf{s}_k) \quad (2.36)$$

zugrunde.

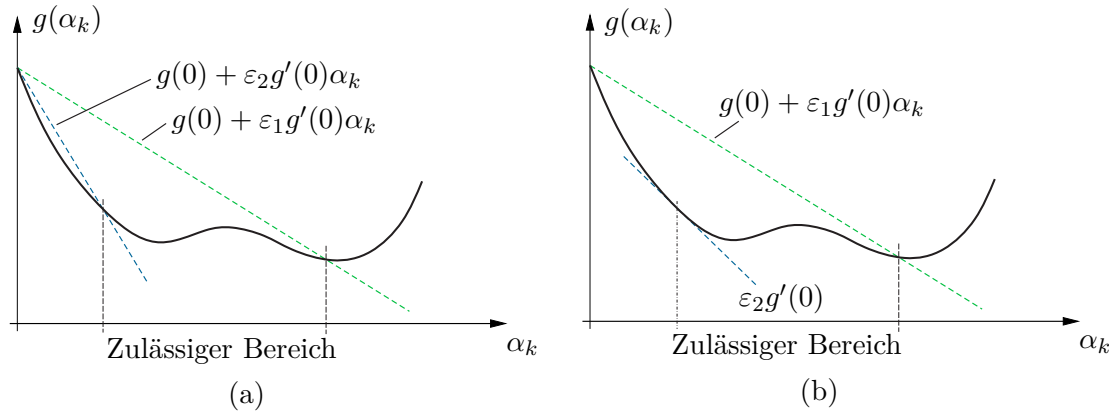


Abbildung 2.4: Veranschaulichung der Armijo-Goldstein-Bedingung (a) und Wolfe-Bedingung (b).

Armijo-Goldstein-Bedingung: Entwickelt man $g(\alpha_k)$ um $\alpha_k = 0$ in eine Taylorreihe und bricht nach dem linearen Glied ab, so erhält man

$$g(\alpha_k) \approx g(0) + g'(0) \alpha_k. \quad (2.37)$$

Bei der *Armijo-Goldstein-Bedingung* wird nun die Schrittweite α_k so gewählt, dass für ein festes ε_1 , $0 < \varepsilon_1 < 1$, die Ungleichung

$$g(\alpha_k) \leq g(0) + \varepsilon_1 g'(0) \alpha_k \quad (2.38)$$

erfüllt ist. Dies garantiert, dass die Schrittweite α_k nach oben hin beschränkt ist. Um sicherzustellen, dass die Schrittweite nicht zu klein wird, führt man bei der *Armijo-Goldstein-Bedingung* zusätzlich einen Parameter ε_2 mit $\varepsilon_1 < \varepsilon_2 < 1$ ein und fordert, dass die Schrittweite α_k der Ungleichung

$$g(\alpha_k) > g(0) + \varepsilon_2 g'(0) \alpha_k \quad (2.39)$$

genügt. Abbildung 2.4(a) zeigt eine grafische Veranschaulichung der Armijo-Goldstein-Bedingung. In der Praxis geht man häufig so vor, dass man in einem ersten Schritt einen (weitgehend beliebigen) Startwert für α_k wählt. Ist für dieses α_k die Ungleichung (2.38) erfüllt, dann erhöht man α_k sukzessive um den Faktor $\varepsilon_2/\varepsilon_1$ solange, bis die Ungleichung (2.38) erstmals verletzt wird. Der vorletzte Wert von α_k wird dann als geeignete Schrittweite gewählt. Umgekehrt, wenn der Startwert von α_k die Ungleichung (2.38) nicht erfüllt, dann wird α_k sukzessive durch den Faktor $\varepsilon_2/\varepsilon_1$ dividiert, bis erstmals die Ungleichung (2.38) erfüllt ist. Typische Parameterwerte sind $\varepsilon_1 = 0.1$ und $\varepsilon_2 = 0.2$. Man beachte jedoch, dass bei zu großem ε_1 die Abstiegsbedingung zu restriktiv wird, d. h. die Schrittweite wird zu klein.

Wolfe-Bedingung: Wenn die Ableitungen der Kostenfunktion $g(\alpha_k)$ sehr einfach berechnet werden können, eignet sich als Alternative zur Armijo-Goldstein-Bedingung die so genannte *Wolfe-Bedingung*. Dabei wird ein weiterer Parameter ε_2 mit $0 < \varepsilon_1 < \varepsilon_2 < 1$ eingeführt und von der Schrittweite α_k wird gefordert, dass sie die Ungleichungen (2.38) und

$$g'(\alpha_k) \geq \varepsilon_2 g'(0) \quad (2.40)$$

erfüllt. Abbildung 2.4(b) gibt eine grafische Veranschaulichung dieses Sachverhaltes. Typische Werte für ε_2 sind 0.9, wenn die Suchrichtung $\mathbf{s}_k$ über die Newton-Methode oder die Quasi-Newton-Methode und 0.1, wenn $\mathbf{s}_k$ über die konjugierte Gradientenmethode bestimmt wurde.

2.3.2 Wahl der Suchrichtung

2.3.2.1 Gradientenmethode

Bei der *Gradientenmethode*, sie wird auch *Methode des steilsten Abstiegs* (englisch: *steepest descent method*) genannt, wählt man als Suchrichtung $\mathbf{s}_k$ in (2.28) den *negativen Gradienten* an der Stelle $\mathbf{x}_k$, d. h. die Abstiegsrichtung

$$\mathbf{s}_k = -(\nabla f)(\mathbf{x}_k) . \quad (2.41)$$

Wird $g(\alpha_k) = f(\mathbf{x}_k + \alpha_k \mathbf{s}_k)$ um den Punkt $\alpha_k = 0$ in eine Taylorreihe mit $\mathbf{s}_k$ gemäß (2.41) entwickelt

$$g(\alpha_k) = f(\mathbf{x}_k + \alpha_k \mathbf{s}_k) = f(\mathbf{x}_{k+1}) = f(\mathbf{x}_k) - \alpha_k \|(\nabla f)(\mathbf{x}_k)\|_2^2 + \mathcal{O}(\alpha_k^2) , \quad (2.42)$$

wobei $\mathcal{O}(\alpha_k^2)$ den quadratischen Restterm bezeichnet, so zeigt sich, dass für hinreichend kleines α_k die Abstiegsbedingung (2.14) für $(\nabla f)(\mathbf{x}_k) \neq \mathbf{0}$ erfüllt ist.

Um die Konvergenzeigenschaften der Gradientenmethode näher zu untersuchen, betrachte man das quadratische Minimierungsproblem

$$\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} - \mathbf{x}^T \mathbf{b} \quad (2.43)$$

mit der symmetrischen positiv definiten Matrix $\mathbf{Q} \in \mathbb{R}^{n \times n}$. Da die Hessematrix $(\nabla^2 f)(\mathbf{x}) = \mathbf{Q}$ von $f(\mathbf{x})$ positiv definit ist, folgt aus der Eigenschaft (e) konvexer Funktionen von Abschnitt 1.3.6.2 die strikte Konvexität von $f(\mathbf{x})$. Auf Grund der Sätze 2.1 und 2.5 ergibt sich daher das strikte globale Minimum $\mathbf{x}^*$ von $f(\mathbf{x})$ aus der Beziehung

$$(\nabla f)(\mathbf{x}^*) = \mathbf{g}(\mathbf{x}^*) = \mathbf{Q} \mathbf{x}^* - \mathbf{b} = \mathbf{0} \quad (2.44)$$

in der Form

$$\mathbf{x}^* = \mathbf{Q}^{-1} \mathbf{b} . \quad (2.45)$$

Die Iterationsvorschrift gemäß Gradientenmethode lautet in diesem Fall, siehe (2.28)

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g}_k \quad \text{mit} \quad \mathbf{g}_k = \mathbf{g}(\mathbf{x}_k) = \mathbf{Q} \mathbf{x}_k - \mathbf{b} . \quad (2.46)$$

Die optimale Schrittweite α_k^* kann durch explizites Lösen des Optimierungsproblems gemäß (2.30)

$$\min_{\alpha_k > 0} f(\mathbf{x}_k + \alpha_k \mathbf{s}_k) = \frac{1}{2} (\mathbf{x}_k - \alpha_k \mathbf{g}_k)^T \mathbf{Q} (\mathbf{x}_k - \alpha_k \mathbf{g}_k) - (\mathbf{x}_k - \alpha_k \mathbf{g}_k)^T \mathbf{b} \quad (2.47)$$

in der Form

$$\alpha_k^* = \frac{\mathbf{g}_k^T \mathbf{g}_k}{\mathbf{g}_k^T \mathbf{Q} \mathbf{g}_k} \quad (2.48)$$

berechnet werden.

Aufgabe 2.5. Zeigen Sie die Gültigkeit von (2.48).

Zusammenfassend lässt sich damit die Gradientenmethode für die quadratische Kostenfunktion (2.43) wie folgt

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \frac{\mathbf{g}_k^T \mathbf{g}_k}{\mathbf{g}_k^T \mathbf{Q} \mathbf{g}_k} \mathbf{g}_k, \quad \mathbf{g}_k = \mathbf{Q} \mathbf{x}_k - \mathbf{b} \quad (2.49)$$

anschreiben.

Für weitere Überlegungen zum Konvergenzverhalten der Gradientenmethode ist es vorteilhaft, unter Verwendung von (2.45) anstelle von $f(\mathbf{x})$ die Fehlerfunktion

$$\begin{aligned} e(\mathbf{x}) &= \|\mathbf{x} - \mathbf{x}^*\|_{\mathbf{Q}} = \sqrt{(\mathbf{x} - \mathbf{x}^*)^T \mathbf{Q} (\mathbf{x} - \mathbf{x}^*)} \\ &= \sqrt{\mathbf{x}^T \mathbf{Q} \mathbf{x} - 2\mathbf{x}^T \mathbf{Q} \mathbf{x}^* + (\mathbf{x}^*)^T \mathbf{Q} \mathbf{x}^*} = \sqrt{2f(\mathbf{x}) + (\mathbf{x}^*)^T \mathbf{Q} \mathbf{x}^*} \end{aligned} \quad (2.50)$$

zu betrachten. Die Minima $\mathbf{x}^*$ der Funktionen $f(\mathbf{x})$ und $e(\mathbf{x})$ sind identisch. Die Vektornorm $\|\cdot\|_{\mathbf{Q}}$ mit einer symmetrischen positiv definiten Matrix $\mathbf{Q}$ wird auch als *Energienorm* bezeichnet.

Lemma 2.1 (Zur Konvergenzrate des Gradientenverfahrens). Mit der Iterationsvorschrift des Gradientenverfahrens (2.49) gilt für die Fehlerfunktion $e(\mathbf{x})$ die Beziehung

$$e(\mathbf{x}_{k+1}) = \left(1 - \frac{(\mathbf{g}_k^T \mathbf{g}_k)^2}{\mathbf{g}_k^T \mathbf{Q} \mathbf{g}_k \mathbf{g}_k^T \mathbf{Q}^{-1} \mathbf{g}_k} \right)^{\frac{1}{2}} e(\mathbf{x}_k) . \quad (2.51)$$

Beweis. Aus (2.45) und (2.49) erhält man

$$\mathbf{g}_k = \mathbf{Q} \mathbf{x}_k - \mathbf{b} = \mathbf{Q}(\mathbf{x}_k - \mathbf{x}^*) . \quad (2.52)$$

Folglich gilt

$$e(\mathbf{x}_k) = \sqrt{\mathbf{g}_k^T \mathbf{Q}^{-1} \mathbf{g}_k} . \quad (2.53)$$

Aus diesen Beziehungen und der Iterationsvorschrift (2.49) lässt sich nun direkt (2.51) berechnen.

$$\begin{aligned} e(\mathbf{x}_{k+1}) &= \left(\left(\mathbf{x}_k - \frac{\mathbf{g}_k^T \mathbf{g}_k}{\mathbf{g}_k^T \mathbf{Q} \mathbf{g}_k} \mathbf{g}_k - \mathbf{x}^* \right)^T \mathbf{Q} \left(\mathbf{x}_k - \frac{\mathbf{g}_k^T \mathbf{g}_k}{\mathbf{g}_k^T \mathbf{Q} \mathbf{g}_k} \mathbf{g}_k - \mathbf{x}^* \right) \right)^{\frac{1}{2}} \\ &= \left((\mathbf{x}_k - \mathbf{x}^*)^T \mathbf{Q} (\mathbf{x}_k - \mathbf{x}^*) - \frac{(\mathbf{g}_k^T \mathbf{g}_k)^2}{\mathbf{g}_k^T \mathbf{Q} \mathbf{g}_k} \right)^{\frac{1}{2}} \\ &= \left(1 - \frac{(\mathbf{g}_k^T \mathbf{g}_k)^2}{\mathbf{g}_k^T \mathbf{Q} \mathbf{g}_k \mathbf{g}_k^T \mathbf{Q}^{-1} \mathbf{g}_k} \right)^{\frac{1}{2}} e(\mathbf{x}_k) \end{aligned} \quad (2.54)$$

□

Um nun die Konvergenzrate der Gradientenmethode bei Verwendung der Fehlerfunktion $e(\mathbf{x})$ abschätzen zu können, wird noch folgendes Lemma benötigt.

Lemma 2.2 (Ungleichung von Kantorovich). Es sei $\mathbf{Q} \in \mathbb{R}^{n \times n}$ eine symmetrische positiv definite Matrix. Für jeden Vektor $\mathbf{x} \in \mathbb{R}^n$ mit $\mathbf{x} \neq \mathbf{0}$ gilt dann die Ungleichung

$$\frac{(\mathbf{x}^T \mathbf{x})^2}{(\mathbf{x}^T \mathbf{Q} \mathbf{x})(\mathbf{x}^T \mathbf{Q}^{-1} \mathbf{x})} \geq \frac{4\lambda_{\min} \lambda_{\max}}{(\lambda_{\min} + \lambda_{\max})^2} , \quad (2.55)$$

wobei $\lambda_{\min}$ und $\lambda_{\max}$ den kleinsten und größten (reellen und positiven) Eigenwert der Matrix $\mathbf{Q}$ bezeichnen.

Der Beweis von Lemma 2.2 ist z. B. in [2.2] skizziert.

Satz 2.7 (Konvergenz der Gradientenmethode für quadratische Kostenfunktionen). Für jeden Anfangswert $\mathbf{x}_0 \in \mathbb{R}^n$ konvergiert die Iterationsvorschrift (2.49) der Gradientenmethode gegen das strikte globale Minimum $\mathbf{x}^*$ der Kostenfunktion $f(\mathbf{x})$ gemäß (2.43). Die Fehlerfunktion $e(\mathbf{x})$ gemäß (2.50) konvergiert dabei linear gegen 0 mit der

Konvergenzrate

$$e(\mathbf{x}_{k+1}) \leq \frac{\kappa - 1}{\kappa + 1} e(\mathbf{x}_k) , \quad (2.56)$$

wobei $\kappa = \lambda_{\max}/\lambda_{\min}$ die spektrale Konditionszahl der Matrix $\mathbf{Q}$, also das Verhältnis des größten zum kleinsten (reellen und positiven) Eigenwert $\lambda_{\max}$ und $\lambda_{\min}$ der Matrix $\mathbf{Q}$, bezeichnet.

Beweis. Aus den Lemmas 2.1 und 2.2 folgt unmittelbar

$$e(\mathbf{x}_{k+1}) \leq \left(1 - \frac{4\lambda_{\min}\lambda_{\max}}{(\lambda_{\min} + \lambda_{\max})^2}\right)^{\frac{1}{2}} e(\mathbf{x}_k) = \frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}} e(\mathbf{x}_k) . \quad (2.57)$$

□

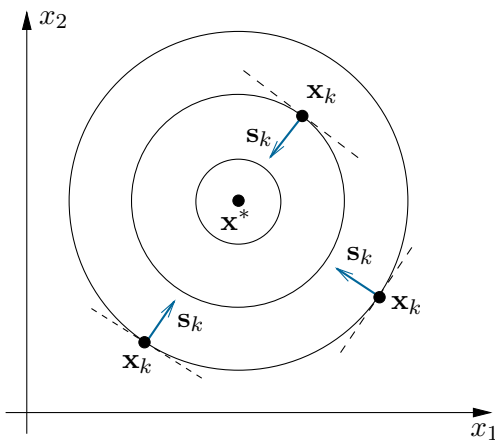


Abb. 2.5: Beispiel eines ideal konditionierten Problems für die Gradientenmethode.

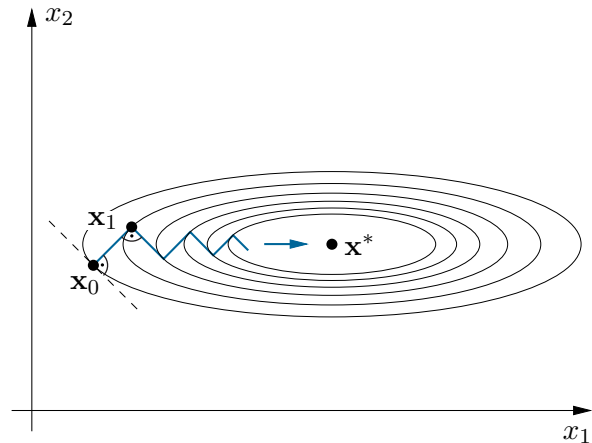


Abb. 2.6: Beispiel eines schlecht konditionierten Problems für die Gradientenmethode.

Satz 2.7 lässt sich nun wie folgt interpretieren. Auf Grund der positiven Definitheit der Matrix $\mathbf{Q}$ sind die Höhenlinien ($f(\mathbf{x}) = \text{konst.}$) der Kostenfunktion (2.43) n -dimensionale Ellipsoide, deren Achsen mit den Richtungen der n paarweise orthogonalen Eigenvektoren der Matrix $\mathbf{Q}$ zusammenfallen und deren Längen invers proportional zum jeweiligen (positiv reellen) Eigenwert sind. Der Gradient $(\nabla f)(\mathbf{x}_k)$ steht orthogonal zur Höhenlinie durch den Punkt $\mathbf{x}_k$, siehe Abbildungen 2.5 und 2.6 für Beispiele mit $\mathbf{x} \in \mathbb{R}^2$. Wenn die Eigenwerte von $\mathbf{Q}$ in (2.43) alle in der gleichen Größenordnung liegen, weist die Gradientenmethode ein gutes Konvergenzverhalten auf, im Falle von $\lambda_{\min} = \lambda_{\max}$ bzw. $\kappa = 1$ konvergiert das Verfahren sogar in einem einzigen Schritt, siehe Abbildung 2.5. Bei schlecht konditionierten Problemen (κ sehr groß) kann die Gradientenmethode sehr langsam konvergieren, siehe Abbildung 2.6. Das in Abbildung 2.6 gezeigte Verhalten wird im Englischen als *zigzagging* bezeichnet.

Die Gradientenmethode kann natürlich auch auf nichtquadratische Kostenfunktionen angewandt werden. Für diesen Fall beschreibt der nachfolgende Satz das Konvergenzverhalten

der Gradientenmethode. Sein Beweis findet sich z. B. in [2.2].

Satz 2.8 (Konvergenz der Gradientenmethode für allgemeine Kostenfunktionen). Gegeben sei die Kostenfunktion $f \in C^2$ definiert im $\mathbb{R}^n$ mit $\mathbf{x}^*$ als lokales Minimum. Angenommen, die Hessematrix $(\nabla^2 f)(\mathbf{x}^*)$ hat den kleinsten und größten Eigenwert $\lambda_{\min} > 0$ und $\lambda_{\max} > 0$ und die spektrale Konditionszahl $\kappa = \lambda_{\max}/\lambda_{\min}$. Wenn die Folge $\{\mathbf{x}_k\}$ generiert durch die Gradientenmethode

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k (\nabla f)(\mathbf{x}_k) \quad (2.58)$$

für eine geeignete Schrittweite α_k gegen das lokale Minimum $\mathbf{x}^*$ konvergiert, dann konvergiert die Folge $\{e(\mathbf{x}_k)\}$ mit $e(\mathbf{x}) = f(\mathbf{x}) - f(\mathbf{x}^*)$ linear gegen 0 mit einer Konvergenzrate die im besten Fall $\left(\frac{\kappa-1}{\kappa+1}\right)^2$ beträgt.

Schlecht konditionierte Problemstellungen bei der Gradientenmethode können mitunter durch eine geeignete *Skalierung* oder *Transformation* verbessert werden. Die Idee beruht darauf, dass die Aufgabe, ein Minimum der Kostenfunktion $f(\mathbf{x})$ zu finden, äquivalent dazu ist, für die Funktion $h(\mathbf{y}) = f(\mathbf{T}\mathbf{y})$ mit $\mathbf{x} = \mathbf{T}\mathbf{y}$ und der regulären Matrix $\mathbf{T} \in \mathbb{R}^{n \times n}$ ein Minimum zu suchen. Entwickelt man die Funktion $h(\mathbf{y})$ um den optimalen Punkt $\mathbf{y}^* = \mathbf{T}^{-1}\mathbf{x}^*$ in eine Taylorreihe

$$\begin{aligned} h(\mathbf{y}) &= h(\mathbf{y}^*) + (\nabla h)^T(\mathbf{y}^*)(\mathbf{y} - \mathbf{y}^*) + \frac{1}{2}(\mathbf{y} - \mathbf{y}^*)^T (\nabla^2 h)(\mathbf{y}^*)(\mathbf{y} - \mathbf{y}^*) + \dots \\ &= h(\mathbf{y}^*) + (\nabla f)^T(\mathbf{x}^*)\mathbf{T}(\mathbf{y} - \mathbf{y}^*) \\ &\quad + \frac{1}{2}(\mathbf{y} - \mathbf{y}^*)^T \mathbf{T}^T (\nabla^2 f)(\mathbf{x}^*)\mathbf{T}(\mathbf{y} - \mathbf{y}^*) + \dots, \end{aligned} \quad (2.59)$$

so erkennt man, dass durch geeignete Wahl von $\mathbf{T}$ die Verteilung der Eigenwerte der Hessematrix

$$(\nabla^2 h)(\mathbf{y}^*) = \mathbf{T}^T (\nabla^2 f)(\mathbf{x}^*)\mathbf{T} \quad (2.60)$$

gegenüber der Verteilung der Eigenwerte von $(\nabla^2 f)(\mathbf{x}^*)$ verbessert werden kann. Aus (2.60) folgt, dass mit der idealen Wahl $\mathbf{T} = (\nabla^2 f)^{-\frac{1}{2}}(\mathbf{x}^*)$ für die Hessematrix $(\nabla^2 h)(\mathbf{y}^*) = \mathbf{E}$ mit der Einheitsmatrix $\mathbf{E} \in \mathbb{R}^{n \times n}$ folgen würde und das Gradientenverfahren bei quadratischen Optimierungsproblemen nach einem Schritt konvergieren würde (vgl. Abbildung 2.5). Praktisch ist diese Vorgehensweise sehr ähnlich zur später beschriebenen Newton-Methode. Sie hat aber den Nachteil, dass die gesamte Hessematrix explizit berechnet werden muss. Um diesen Rechenaufwand zu vermeiden, wird alternativ häufig eine Diagonalmatrix $\mathbf{T}$ verwendet, deren Diagonaleinträge beispielsweise in der Form $T_{ii} = ((\nabla^2 f)_{ii}(\mathbf{x}^*))^{-\frac{1}{2}}$, $i = 1, \dots, n$ gewählt werden können. Wird eine Diagonalmatrix $\mathbf{T}$ verwendet, so führt dies zu einer reinen Skalierung (Normierung) der Optimierungsvariablen.

Die Vor- und Nachteile der Gradientenmethode lassen sich wie folgt zusammenfassen:

- + einfaches Verfahren
- + Hessematrix $(\nabla^2 f)(\mathbf{x}_k)$ (Berechnungsaufwand $\mathcal{O}(n^2)$) nicht erforderlich, nur der Gradient $(\nabla f)(\mathbf{x}_k)$ (Berechnungsaufwand $\mathcal{O}(n)$) wird benötigt
- + Konvergenz auch für Startwerte, die weiter vom Minimum entfernt sind

- langsame Konvergenz bei schlecht konditionierten und schlecht skalierten Problemen
- lediglich lineare Konvergenzordnung

2.3.2.2 Konjugierte Gradientenmethode

Die konjugierte Gradientenmethode (englisch: *conjugate gradient method* oder kurz *CG method*) versucht nun bei nur geringfügig erhöhtem Rechen- und Speicheraufwand eine schnellere Konvergenz als die Gradientenmethode zu erreichen. Weitere Informationen zu der Methode finden sich z. B. in [2.2, 2.4–2.6]. Ursprünglich wurde diese Methode zur Lösung hochdimensionaler quadratischer Probleme der Form (siehe auch (2.43))

$$\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} - \mathbf{x}^T \mathbf{b} \quad (2.61)$$

sowie hochdimensionaler linearer Gleichungssysteme der Form

$$\mathbf{0} = \mathbf{Q} \mathbf{x} - \mathbf{b} \quad (2.62)$$

jeweils mit einer positiv definiten Matrix $\mathbf{Q} \in \mathbb{R}^{n \times n}$ und einem beliebigen Vektor $\mathbf{b} \in \mathbb{R}^n$ entwickelt. Da die Kostenfunktion (2.61) strikt konvex ist, entspricht (2.62) einem hinreichenden Optimalitätskriterium erster Ordnung für ein striktes globales Minimum. Soll ein lineares Gleichungssystem

$$\mathbf{0} = \mathbf{A} \mathbf{x} - \mathbf{c} \quad (2.63)$$

mit einer regulären aber nicht positiv definiten Matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ und einem beliebigen Vektor $\mathbf{c} \in \mathbb{R}^n$ gelöst werden, so kann es durch linksseitige Multiplikation mit $\mathbf{A}^T$ auf die Form

$$\mathbf{0} = \mathbf{A}^T \mathbf{A} \mathbf{x} - \mathbf{A}^T \mathbf{c} \quad (2.64)$$

mit der positiv definiten Matrix $\mathbf{A}^T \mathbf{A}$ gebracht werden und ebenfalls mit der konjugierten Gradientenmethode gelöst werden.

Bevor nun die konjugierte Gradientenmethode konkret erläutert wird, sollen einige Grundlagen dazu erarbeitet werden.

Definition 2.2 (Q-Orthogonalität). Nichttriviale Vektoren $\mathbf{d}_j$, $j = 0, \dots, k$ heißen *konjugiert bezüglich einer positiv definiten Matrix $\mathbf{Q}$* bzw. *Q-orthogonal*, wenn gilt $\mathbf{d}_i^T \mathbf{Q} \mathbf{d}_j = 0 \forall i \neq j$. Die Vektoren $\mathbf{d}_j$, $j = 0, \dots, k$ sind dann linear unabhängig.

Für $\mathbf{Q} = \mathbf{E}$ fällt der Begriff der Q-Orthogonalität folglich mit dem klassischen Begriff der Orthogonalität zusammen. In Abbildung 2.7 sind Vektoren mit gleicher Farbe Q-orthogonal.

Aufgabe 2.6. Beweisen Sie, dass Q-orthogonale Vektoren $\mathbf{d}_j$, $j = 0, \dots, k$ linear unabhängig sind.

Bei der konjugierten Gradientenmethode wird ausgehend von einer Startlösung $\mathbf{x}_0$ iterativ entlang von Q-orthogonalen Suchrichtungen $\mathbf{d}_j$ die optimale Lösung $\mathbf{x}^*$ des Optimierungsproblems (2.61) gesucht. In jedem Iterationsschritt k gilt daher

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k \quad (2.65)$$

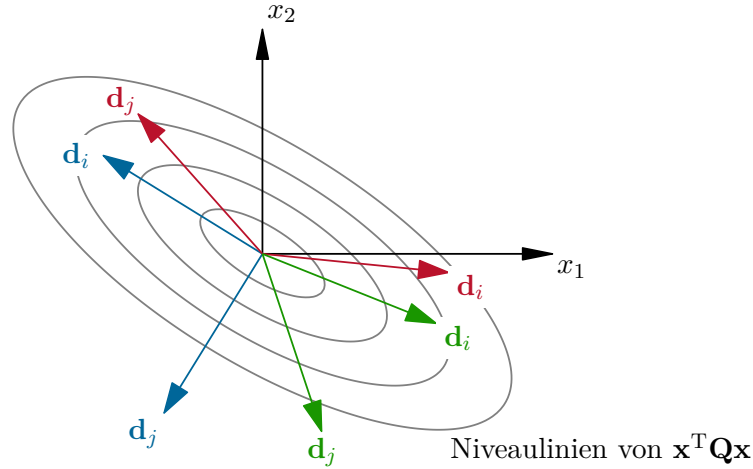


Abbildung 2.7: Beispiele für $\mathbf{Q}$ -orthogonale Vektoren im Fall $n = 2$. Vektoren mit gleicher Farbe sind $\mathbf{Q}$ -orthogonal.

mit der optimalen Schrittweite α_k sowie

$$\mathbf{x}_{k+1} - \mathbf{x}_0 = \sum_{j=0}^k \alpha_j \mathbf{d}_j . \quad (2.66)$$

Für den Gradienten von (2.61) wird die Abkürzung

$$\mathbf{g}_k = (\nabla f)(\mathbf{x}_k) = \mathbf{Q} \mathbf{x}_k - \mathbf{b} \quad (2.67)$$

eingeführt. Es soll nun gezeigt werden, dass die optimalen Schrittweiten α_j mit geringem mathematischem Aufwand berechnet werden können. Aus (2.61) und (2.66) folgt unter Ausnützung der $\mathbf{Q}$ -Orthogonalität der Vektoren $\mathbf{d}_j$

$$\begin{aligned} f(\mathbf{x}_k) &= \frac{1}{2} \left(\mathbf{x}_0 + \sum_{j=0}^{k-1} \alpha_j \mathbf{d}_j \right)^T \mathbf{Q} \left(\mathbf{x}_0 + \sum_{j=0}^{k-1} \alpha_j \mathbf{d}_j \right) - \left(\mathbf{x}_0 + \sum_{j=0}^{k-1} \alpha_j \mathbf{d}_j \right)^T \mathbf{b} \\ &= \frac{1}{2} \mathbf{x}_0^T \mathbf{Q} \mathbf{x}_0 - \mathbf{x}_0^T \mathbf{b} + \sum_{j=0}^{k-1} \left(\frac{1}{2} \alpha_j^2 \mathbf{d}_j^T \mathbf{Q} \mathbf{d}_j + \alpha_j \mathbf{g}_0^T \mathbf{d}_j \right) . \end{aligned} \quad (2.68)$$

Das Problem zerfällt also in unabhängige skalare Optimierungsprobleme für die Schrittweiten α_j . Deren Lösung lautet

$$\alpha_j = - \frac{\mathbf{g}_0^T \mathbf{d}_j}{\mathbf{d}_j^T \mathbf{Q} \mathbf{d}_j} = - \frac{\mathbf{g}_j^T \mathbf{d}_j}{\mathbf{d}_j^T \mathbf{Q} \mathbf{d}_j} , \quad (2.69)$$

wobei hier die Identität

$$\mathbf{g}_j^T \mathbf{d}_j = \left(\mathbf{Q} \left(\mathbf{x}_0 + \sum_{i=0}^{j-1} \alpha_i \mathbf{d}_i \right) - \mathbf{b} \right)^T \mathbf{d}_j = \mathbf{g}_0^T \mathbf{d}_j \quad (2.70)$$

verwendet wurde. Wird also in jedem Iterationsschritt (2.65) die Schrittweite α_k gemäß (2.69) gewählt, so minimiert $\mathbf{x}_{k+1}$ die Kostenfunktion $f(\mathbf{x}_{k+1})$ in dem von den Vektoren $\mathbf{d}_j$, $j = 0, \dots, k$ aufgespannten Unterraum des $\mathbb{R}^n$. Nachdem die Vektoren $\mathbf{d}_j$, $j = 0, \dots, k$ linear unabhängig sind (siehe Definition 2.2), ergibt sich daraus nach spätestens $k + 1 = n$ Iterationsschritten die optimale Lösung $\mathbf{x}^* = \mathbf{x}_{k+1}$ des quadratischen Optimierungsproblems (2.61) (endliche Konvergenz). Wie die noch unbekannten Vektoren $\mathbf{d}_j$ zu wählen sind, geht aus dem nachfolgend beschriebenen Algorithmus der konjugierten Gradientenmethode hervor.

Satz 2.9 (Konjugierte Gradientenmethode). *Für jeden Anfangswert $\mathbf{x}_0$ konvergiert die Folge*

$$\alpha_k = \frac{\mathbf{g}_k^T \mathbf{g}_k}{\mathbf{d}_k^T \mathbf{Q} \mathbf{d}_k} \quad (2.71a)$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k \quad (2.71b)$$

$$\beta_k = \frac{\mathbf{g}_{k+1}^T \mathbf{g}_{k+1}}{\mathbf{g}_k^T \mathbf{g}_k} \quad (2.71c)$$

$$\mathbf{d}_{k+1} = -\mathbf{g}_{k+1} + \beta_k \mathbf{d}_k \quad (2.71d)$$

mit $\mathbf{g}_k = \mathbf{Q}\mathbf{x}_k - \mathbf{b}$ und dem Anfangswert

$$\mathbf{d}_0 = -\mathbf{g}_0 = \mathbf{b} - \mathbf{Q}\mathbf{x}_0 \quad (2.72)$$

in höchstens n Iterationsschritten gegen die eindeutige Lösung $\mathbf{x}^*$ des quadratischen Optimierungsproblems (2.61). Die so konstruierte Folge von Suchrichtungen und Gradienten weist die Eigenschaften

$$\mathbf{g}_i^T \mathbf{g}_j = 0 \quad \forall i \neq j \quad (2.73a)$$

$$\mathbf{g}_i^T \mathbf{d}_j = 0 \quad \forall i > j \quad (2.73b)$$

$$\mathbf{d}_i^T \mathbf{Q} \mathbf{d}_j = 0 \quad \forall i \neq j \quad (2.73c)$$

auf. D. h. die Gradienten sind zueinander orthogonal, die Gradienten sind orthogonal auf alle bisherigen Suchrichtungen und die Suchrichtungen sind $\mathbf{Q}$ -orthogonal.

Beweis. Entsprechend der Iterationsvorschrift (2.71b) gilt der Zusammenhang

$$\mathbf{g}_{k+1} = \mathbf{Q}\mathbf{x}_{k+1} - \mathbf{b} = \mathbf{Q}\mathbf{x}_k - \mathbf{b} + \alpha_k \mathbf{Q}\mathbf{d}_k = \mathbf{g}_k + \alpha_k \mathbf{Q}\mathbf{d}_k. \quad (2.74)$$

Nachfolgend werden ohne explizite Erwähnung laufend die Beziehungen (2.71) und (2.72) verwendet. Zunächst sollen die Eigenschaften (2.73) mittels vollständiger Induktion gezeigt werden. Zum Induktionsbeginn gilt unter Zuhilfenahme von (2.74)

$$\mathbf{g}_0^T \mathbf{g}_1 = \mathbf{g}_0^T (\mathbf{g}_0 + \alpha_0 \mathbf{Q}\mathbf{d}_0) = \mathbf{g}_0^T \mathbf{g}_0 - \mathbf{d}_0^T \alpha_0 \mathbf{Q}\mathbf{d}_0 = 0 \quad (2.75a)$$

für (2.73a) und daher

$$\mathbf{d}_0^T \mathbf{g}_1 = 0 \quad (2.75b)$$

für (2.73b). Aus (2.74) folgt $\mathbf{Q}\mathbf{d}_0 = (\mathbf{g}_1 - \mathbf{g}_0)/\alpha_0$. Mit dieser Beziehung und (2.75a) ergibt sich die Gültigkeit von (2.73c) zum Induktionsbeginn in der Form

$$\mathbf{d}_0^T \mathbf{Q}\mathbf{d}_1 = \mathbf{d}_0^T \mathbf{Q}(-\mathbf{g}_1 + \beta_0 \mathbf{d}_0) = \frac{1}{\alpha_0} (\mathbf{g}_1 - \mathbf{g}_0)^T \left(-\mathbf{g}_1 - \frac{\mathbf{g}_1^T \mathbf{g}_1}{\mathbf{g}_0^T \mathbf{g}_0} \mathbf{g}_0 \right) = 0. \quad (2.75c)$$

Basierend auf der Induktionsannahme, dass (2.73) bei festem Index i für alle $j < i$ gilt, soll nun gezeigt werden, dass (2.73) auch für alle $j < i + 1$ gilt. Dabei wird (2.74) mehrfach verwendet. Aus

$$\mathbf{g}_{i+1}^T \mathbf{g}_j = (\mathbf{g}_i + \alpha_i \mathbf{Q}\mathbf{d}_i)^T \mathbf{g}_j = \mathbf{g}_i^T \mathbf{g}_j + \alpha_i \mathbf{d}_i^T \mathbf{Q}(-\mathbf{d}_j + \beta_{j-1} \mathbf{d}_{j-1}) = 0 \quad (2.76a)$$

folgt im ersten Schritt die Gültigkeit von (2.73a) für alle $j < i + 1$. Unter Verwendung von (2.76a) folgt aus

$$\begin{aligned} \mathbf{g}_{i+1}^T \mathbf{d}_j &= \mathbf{g}_{i+1}^T (-\mathbf{g}_j + \beta_{j-1} \mathbf{d}_{j-1}) \\ &= \mathbf{g}_{i+1}^T \beta_{j-1} \mathbf{d}_{j-1} = (\mathbf{g}_i + \alpha_i \mathbf{Q}\mathbf{d}_i)^T \beta_{j-1} \mathbf{d}_{j-1} = 0 \end{aligned} \quad (2.76b)$$

im zweiten Schritt die Gültigkeit von (2.73b) für alle $j < i + 1$. Wieder unter Verwendung von (2.76a) folgt aus

$$\begin{aligned} \mathbf{d}_{i+1}^T \mathbf{Q}\mathbf{d}_j &= (-\mathbf{g}_{i+1} + \beta_i \mathbf{d}_i)^T \mathbf{Q}\mathbf{d}_j = -\mathbf{g}_{i+1}^T \frac{1}{\alpha_j} (\mathbf{g}_{j+1} - \mathbf{g}_j) + \beta_i \mathbf{d}_i^T \mathbf{Q}\mathbf{d}_j \\ &= -\mathbf{g}_{i+1}^T \mathbf{g}_{j+1} \frac{\mathbf{d}_j^T \mathbf{Q}\mathbf{d}_j}{\mathbf{g}_j^T \mathbf{g}_j} + \beta_i \mathbf{d}_i^T \mathbf{Q}\mathbf{d}_j = 0 \end{aligned} \quad (2.76c)$$

im dritten Schritt die Gültigkeit von (2.73c) für alle $j < i + 1$. Damit ist (2.73) gezeigt. Wegen

$$\alpha_k = -\frac{\mathbf{g}_k^T \mathbf{d}_k}{\mathbf{d}_k^T \mathbf{Q}\mathbf{d}_k} = \frac{\mathbf{g}_k^T (\mathbf{g}_k - \beta_{k-1} \mathbf{d}_{k-1})}{\mathbf{d}_k^T \mathbf{Q}\mathbf{d}_k} = \frac{\mathbf{g}_k^T \mathbf{g}_k}{\mathbf{d}_k^T \mathbf{Q}\mathbf{d}_k} \quad (2.77)$$

liefert (2.71a) die optimale Schrittweite gemäß (2.69). Dass die konjugierte Gradientenmethode nach höchstens n Iterationsschritten zur optimalen Lösung $\mathbf{x}^*$ konvergiert, wenn die Schrittweiten α_k optimal gewählt werden und die Vektoren $\mathbf{d}_k$, $k = 0, \dots, n - 1$ $\mathbf{Q}$ -orthogonal sind, folgt bereits aus den Überlegungen, die dem Satz 2.9 vorangegangen sind. $\square$

Aufgabe 2.7. Zeigen Sie, dass die Suchrichtungen $\mathbf{d}_k$ der konjugierten Gradientenmethode nach Satz 2.9 Abstiegsrichtungen sind.

Anders als es die Bezeichnung *konjugierte Gradientenmethode* erwarten lässt, sind nicht die lokalen Gradienten $\mathbf{g}_k$ der Kostenfunktion $\mathbf{Q}$ -orthogonal im Sinne der Definition 2.2, sondern die Suchrichtungen $\mathbf{d}_k$. Gemäß (2.71d) bildet der Vektor $\mathbf{d}_{k+1}$ eine

Linearkombination der Richtung des steilsten Abstieges (negativer Gradient $-\mathbf{g}_{k+1}$) und der vorangegangenen Suchrichtung $\mathbf{d}_k$. Der Faktor β_k wird dabei gemäß (2.71c) genau so gewählt, dass die $\mathbf{Q}$ -Orthogonalität zwischen $\mathbf{d}_{k+1}$ und allen vorangegangenen Suchrichtungen sichergestellt ist. Die Gleichung (2.71c) wird auch *Formel von Fletcher-Reeves* genannt. Alternativ dazu kann die sogenannte *Formel von Hestenes-Stiefel*

$$\beta_k = \frac{(\mathbf{g}_{k+1} - \mathbf{g}_k)^T \mathbf{g}_{k+1}}{(\mathbf{g}_{k+1} - \mathbf{g}_k)^T \mathbf{d}_k} \quad (2.78)$$

verwendet werden.

Aufgabe 2.8. Zeigen Sie die Äquivalenz von (2.71c) und (2.78).

Der in Satz 2.9 angegebene Algorithmus der konjugierten Gradientenmethode verursacht nur einen geringen numerischen Aufwand. Konkret ist in jeder Iteration nur einmal das Matrix-Vektor-Produkt $\mathbf{Q}\mathbf{d}_k$ auszurechnen. Der Aufwand dieser Operation reduziert sich deutlich, wenn $\mathbf{Q}$ dünn besetzt ist. Alle übrigen Rechenschritte der konjugierten Gradientenmethode sind elementare Vektoroperationen (inneres Produkt zweier Vektoren, Skalierung eines Vektors und Summe von Vektoren). Außerdem müssen mit Ausnahme der Matrix $\mathbf{Q}$ ausschließlich Vektoren und Skalare gespeichert werden. Alternativ zu (2.71a) kann die Schrittweite α_k auch mit einem der in Abschnitt 2.3.1 beschriebenen Verfahren bestimmt werden, sofern dieses tatsächlich die optimale Schrittweite liefert. Die explizite Berechnung des Gradienten $\mathbf{g}_k$ gemäß (2.67) kann durch eine iterative Berechnung gemäß (2.74) ersetzt werden. Abwandlungen des in Satz 2.9 angegebenen Algorithmus finden sich z. B. in [2.4, 2.6, 2.7].

Der nachfolgende Satz (siehe [2.5, 2.8]) gibt eine Aussage über das asymptotische Konvergenzverhalten der konjugierten Gradientenmethode.

Satz 2.10 (Konvergenz der konjugierten Gradientenmethode — Quadratische Kostenfunktion). Für jeden Anfangswert $\mathbf{x}_0 \in \mathbb{R}^n$ konvergiert die konjugierte Gradientenmethode gemäß Satz 2.9 gegen das strikte globale Minimum $\mathbf{x}^*$ der quadratischen Kostenfunktion $f(\mathbf{x})$ aus (2.61). Die Fehlerfunktion $e(\mathbf{x})$ gemäß (2.50) konvergiert dabei linear gegen 0 mit der Konvergenzrate

$$e(\mathbf{x}_{k+1}) \leq \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} e(\mathbf{x}_k) , \quad (2.79)$$

wobei $\kappa = \lambda_{\max}/\lambda_{\min}$ wieder die spektrale Konditionszahl der Matrix $\mathbf{Q}$ bezeichnet.

Die konjugierte Gradientenmethode ist ein sogenanntes *Krylov-Unterraum-Verfahren*. Um dies zu sehen, wird zunächst der Begriff des Krylov-Unterraums definiert.

Definition 2.3 (Krylov-Unterraum). Für eine quadratische Matrix $\mathbf{Q} \in \mathbb{R}^{n \times n}$ und einen Vektor $\mathbf{g} \in \mathbb{R}^n$ wird

$$\mathcal{K}_m(\mathbf{Q}, \mathbf{g}) = \text{span}\{\mathbf{g}, \mathbf{Q}\mathbf{g}, \dots, \mathbf{Q}^{m-1}\mathbf{g}\} \quad (2.80)$$

mit $m \leq n$ als Krylov-Unterraum der Ordnung m bezeichnet.

Mit einem Krylov-Unterraum-Verfahren wird das lineare Gleichungssystem $\mathbf{0} = \mathbf{Q}\mathbf{x} - \mathbf{b}$ ausgehend von einer Startlösung $\mathbf{x}_0$ iterativ so gelöst, dass $\mathbf{x}_k - \mathbf{x}_0$ im Krylov-Unterraum $\mathcal{K}_k(\mathbf{Q}, \mathbf{g}_0)$ mit $\mathbf{g}_0 = \mathbf{Q}\mathbf{x}_0 - \mathbf{b}$ liegt und eine Norm des Residuums $\mathbf{g}_k = \mathbf{Q}\mathbf{x}_k - \mathbf{b}$ minimiert. In jedem Iterationsschritt wird die Dimension des Krylov-Unterraums um eins erhöht.

Satz 2.11 (Konjugierte Gradientenmethode als Krylov-Unterraum-Verfahren). *Die konjugierte Gradientenmethode gemäß Satz 2.9 ist ein Krylov-Unterraum-Verfahren bei dem ausgehend von einer Startlösung $\mathbf{x}_0$ in jedem Iterationsschritt $k > 0$ das Optimierungsproblem*

$$\min_{\mathbf{x}_k \in \mathbf{x}_0 + \mathcal{K}_k(\mathbf{Q}, \mathbf{g}_0)} \frac{1}{2} \mathbf{x}_k^T \mathbf{Q} \mathbf{x}_k - \mathbf{x}_k^T \mathbf{b} \quad (2.81)$$

mit einer symmetrisch positiv definiten Matrix $\mathbf{Q}$, dem Krylov-Unterraum

$$\begin{aligned} \mathcal{K}_k(\mathbf{Q}, \mathbf{g}_0) &= \text{span}\{\mathbf{g}_0, \mathbf{Q}\mathbf{g}_0, \dots, \mathbf{Q}^{k-1}\mathbf{g}_0\} \\ &= \text{span}\{\mathbf{g}_0, \mathbf{g}_1, \dots, \mathbf{g}_{k-1}\} = \text{span}\{\mathbf{d}_0, \mathbf{d}_1, \dots, \mathbf{d}_{k-1}\} \end{aligned} \quad (2.82)$$

und $\mathbf{g}_0 = \mathbf{Q}\mathbf{x}_0 - \mathbf{b}$ gelöst wird.

Aufgabe 2.9. Beweisen Sie (2.82). Dies ist besonders einfach mit Hilfe der vollständigen Induktion. Der Rest des Beweises von Satz 2.11 folgt aus den Überlegungen, die dem Satz 2.9 vorangegangen sind.

Eine gelegentlich verwendete Erweiterung des Algorithmus nach Satz 2.9 ist die sogenannte *partielle konjugierte Gradientenmethode*. Dabei werden lediglich $m + 1 < n$ Iterationsschritte ausgeführt, ehe das Verfahren an dem so erhaltenen Punkt $\mathbf{x}_{m+1}$ neu initialisiert wird, d. h. es wird mit den Anfangswerten $\mathbf{x}_0 \leftarrow \mathbf{x}_{m+1}$ und $\mathbf{d}_0 = \mathbf{b} - \mathbf{Q}\mathbf{x}_0$ neu gestartet. Für das asymptotische Konvergenzverhalten dieser Methode gilt folgender Satz, welcher z. B. in [2.2] bewiesen wird.

Satz 2.12 (Partielle konjugierte Gradientenmethode). *Gegeben ist das quadratische Optimierungsproblem (2.61). Wenn die positiv definite Matrix $\mathbf{Q}$ $n - m$ Eigenwerte im Intervall $[l, r]$ ($l > 0$) und m Eigenwerte größer als r besitzt, dann zeigt die partielle konjugierte Gradientenmethode, welche nach jeweils $m + 1$ Schritten neu gestartet wird, für die Fehlerfunktion $e(\mathbf{x})$ gemäß (2.50) das Konvergenzverhalten*

$$e(\mathbf{x}_{k+1}) \leq \frac{r - l}{r + l} e(\mathbf{x}_k) . \quad (2.83)$$

Im Sinne dieses Satzes wird der Punkt $\mathbf{x}_{k+1}$ durch $m + 1$ Zwischeniterationen nach Satz 2.9 mit dem Anfangswert $\mathbf{x}_k$ erreicht. Für $m = 0$ (reine Gradientenmethode) liefert Satz 2.12 die gleichen Aussagen wie Satz 2.7.

Soll die konjugierte Gradientenmethode für *nichtquadratische Kostenfunktionen* $f(\mathbf{x})$ verwendet werden, so müssen in Satz 2.9 lediglich die Substitutionen

$$\mathbf{g}_k \leftrightarrow (\nabla f)(\mathbf{x}_k) , \quad \mathbf{Q} \leftrightarrow (\nabla^2 f)(\mathbf{x}_k) \quad (2.84)$$

vorgenommen werden. In diesem Fall wird der Algorithmus im Allgemeinen nicht mehr

nach spätestens n Schritten terminieren. Für nichtquadratische Kostenfunktionen $f(\mathbf{x})$ hat sich auch die Anwendung der partiellen konjugierten Gradientenmethode als günstig erwiesen, da die Eigenschaft der $\mathbf{Q}$ -Orthogonalität der Suchrichtungen mit fortschreitenden Iterationen mehr und mehr verletzt sein kann. Um die aufwändige Berechnung der Hessematrix $(\nabla^2 f)(\mathbf{x}_k)$ zu vermeiden, kann die Schrittweite α_k wieder alternativ mit einem der in Abschnitt 2.3.1 beschriebenen Verfahren festgelegt werden.

Die Vor- und Nachteile der konjugierten Gradientenmethode können wie folgt zusammengefasst werden:

- + einfaches Verfahren, geringer Rechenaufwand und Speicherbedarf, geeignet für große Optimierungsprobleme
- + nur der Gradient $(\nabla f)(\mathbf{x}_k)$ (Berechnungsaufwand $\mathcal{O}(n)$) wird benötigt
- + konvergiert bei quadratischen Optimierungsproblemen nach spätestens n Iterationsschritten
- Konvergenzverhalten variiert je nach Problemstellung, Konvergenzverhalten besser als jenes der Gradientenmethode

2.3.2.3 Newton-Methode

Die Idee der Newton-Methode besteht darin, die allgemeine Kostenfunktion $f(\mathbf{x})$ lokal durch eine quadratische Funktion zu approximieren und diese zu minimieren. Entwickelt man $f(\mathbf{x}) = f(\mathbf{x}_k + \mathbf{s}_k)$ um den Iterationspunkt $\mathbf{x}_k$ in eine Taylorreihe und bricht diese nach dem quadratischen Term ab, so erhält man

$$f(\mathbf{x}) \approx f(\mathbf{x}_k) + \mathbf{s}_k^T (\nabla f)(\mathbf{x}_k) + \frac{1}{2} \mathbf{s}_k^T (\nabla^2 f)(\mathbf{x}_k) \mathbf{s}_k, \quad (2.85)$$

siehe Abbildung 2.8. Die so genannte *Newton-Richtung* $\mathbf{s}_k$ ergibt sich unmittelbar durch Minimierung der rechten Seite von (2.85) bezüglich $\mathbf{s}_k$ in der Form

$$\mathbf{s}_k = -(\nabla^2 f)^{-1}(\mathbf{x}_k) (\nabla f)(\mathbf{x}_k). \quad (2.86)$$

Falls die Hessematrix $(\nabla^2 f)(\mathbf{x}^*)$ am Minimum positiv definit ist, existiert in einer Umgebung um das Minimum die Inverse $(\nabla^2 f)^{-1}(\mathbf{x}_k)$ und die Methode ist wohldefiniert. Man beachte, dass die Berechnung von $\mathbf{s}_k$ gemäß (2.86) keine tatsächliche Inversion von $(\nabla^2 f)(\mathbf{x}_k)$ erfordert. Der nachfolgende Satz gibt die Konvergenzordnung der Newton-Methode an. Sein Beweis ist z. B. in [2.2, 2.5, 2.9] zu finden.

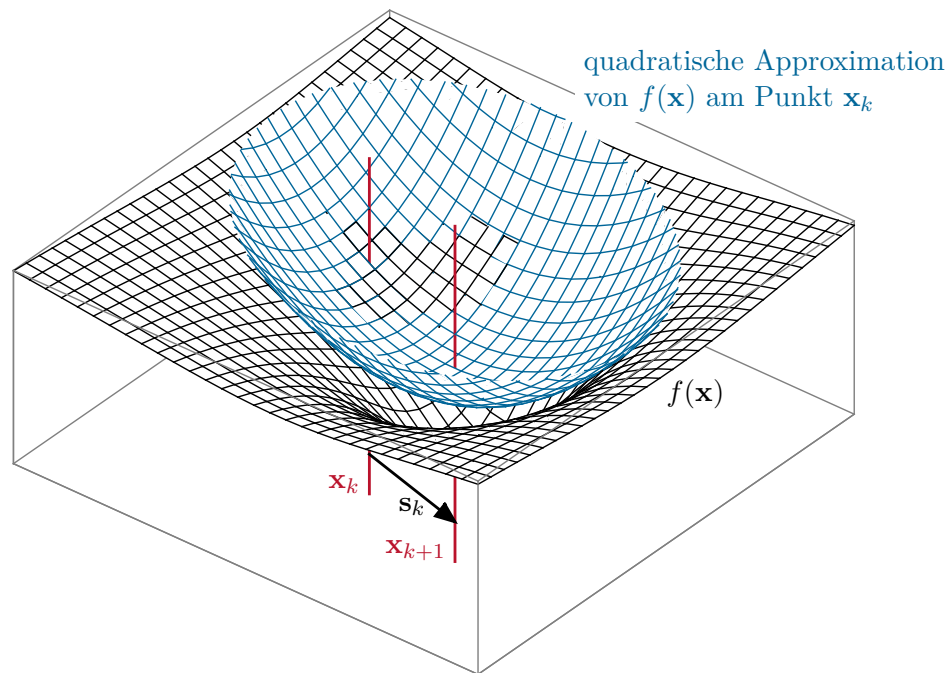


Abbildung 2.8: Lokale quadratische Approximation der Kostenfunktion im Zuge der Newton-Iteration (bei Schrittweite $\alpha_k = 1$).

Satz 2.13 (Konvergenzordnung der Newton-Methode). Gegeben sei die Kostenfunktion $f \in C^3$ definiert im $\mathbb{R}^n$ mit dem lokalen Minimum $\mathbf{x}^*$. Wenn die Hessematrix $(\nabla^2 f)(\mathbf{x}^*)$ positiv definit ist und der Anfangswert $\mathbf{x}_0$ in einer hinreichend nahen Umgebung des Minimums liegt, dann konvergiert die Newton-Iteration

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\nabla^2 f \right)^{-1}(\mathbf{x}_k) (\nabla f)(\mathbf{x}_k) \quad (2.87)$$

mit der Konvergenzordnung 2 gegen das Minimum $\mathbf{x}^*$.

Aufgabe 2.10. Zeigen Sie, dass ein Newton-Iterationsschritt gemäß (2.87) exakt einem Iterationsschritt des Newton-Raphson-Verfahrens zur Lösung der Gleichung

$$(\nabla f)(\mathbf{x}^*) = \mathbf{0} , \quad (2.88)$$

also der notwendigen Optimalitätsbedingung erster Ordnung, entspricht.

Für die praktische Anwendung der Methode führt man noch eine geeignete Schrittweite α_k ein, so dass die Iterationsvorschrift

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \left(\nabla^2 f \right)^{-1}(\mathbf{x}_k) (\nabla f)(\mathbf{x}_k) \quad (2.89)$$

lautet und die Abstiegsbedingung (2.29) erfüllt ist. Das Verfahren wird dann häufig *gedämpfte Newton-Methode* genannt und α_k wird als *Dämpfungsparameter* bezeichnet.

Gelegentlich wird die Einschränkung $\alpha_k \leq 1$ verwendet. Es ist zu erwarten, dass in der Nähe des Minimums die optimale Schrittweite $\alpha_k \approx 1$ ist, weshalb man typischerweise eine iterative Schrittweitsuche mit dem Wert $\alpha_k = 1$ beginnt. Strategien zur Bestimmung der Schrittweite α_k wurden bereits im Abschnitt 2.3.1 erläutert.

Ein Problem, das in diesem Zusammenhang gelegentlich auftritt, besteht im Verlust der positiven Definitheit von $(\nabla^2 f)(\mathbf{x}_k)$, wenn $\mathbf{x}_k$ zu weit vom Minimum entfernt ist. Dann besitzt die rechte Seite von (2.85) kein oder kein striktes Minimum und die Invertierbarkeit von $(\nabla^2 f)(\mathbf{x}_k)$ kann verloren gehen. Ist $(\nabla^2 f)(\mathbf{x}_k)$ nicht positiv definit, so kann statt (2.89) die Iterationsvorschrift

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{M}_k (\nabla f)(\mathbf{x}_k), \quad \mathbf{M}_k = \left((\nabla^2 f)(\mathbf{x}_k) + \varepsilon_k \mathbf{E} \right)^{-1} \quad (2.90)$$

mit einem geeigneten positiven Parameter ε_k verwendet werden. Die Iterationsvorschrift (2.90) geht für $\varepsilon_k = 0$ in die Newton-Methode gemäß (2.87) und für sehr große ε_k in die Gradientenmethode gemäß (2.58) über. Eine geeignete Wahl von ε_k erweist sich jedoch als nicht sehr einfach. Typischerweise wird ε_k beginnend bei einem Startwert $\varepsilon_k > 0$ sukzessive erhöht, bis die Matrix $(\nabla^2 f)(\mathbf{x}_k) + \varepsilon_k \mathbf{E}$ positiv definit ist.

Aufgabe 2.11. Zeigen Sie, dass die Newton-Methode für quadratische Kostenfunktionen

$$\min_{\mathbf{x} \in \mathbb{R}^n} \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} - \mathbf{b}^T \mathbf{x} \quad (2.91)$$

mit der positiv definiten Matrix $\mathbf{Q}$ unabhängig vom Startpunkt $\mathbf{x}_0$ innerhalb von nur einem Iterationsschritt konvergiert.

Die Vor- und Nachteile der Newton-Methode können wie folgt zusammengefasst werden:

- + Konvergenzordnung von 2, wenn die Hessematrix $(\nabla^2 f)(\mathbf{x}_k)$ positiv definit ist, was meist zumindest in der Nähe des Minimums $\mathbf{x}^*$ der Fall ist
- + Konvergenzverhalten besser als jenes der konjugierten Gradientenmethode
- schlecht geeignet in Gebieten in denen $(\nabla^2 f)(\mathbf{x}_k)$ nicht positiv definit ist
- Berechnungsaufwand $\mathcal{O}(n^2)$ für die benötigte Hessematrix $(\nabla^2 f)(\mathbf{x}_k)$, Berechnungsaufwand $\mathcal{O}(n^3)$ für die Suchrichtung $\mathbf{s}_k$

2.3.2.4 Quasi-Newton-Methode

Einer der Hauptnachteile der Newton-Methode liegt in der aufwändigen Berechnung der Hessematrix $(\nabla^2 f)(\mathbf{x}_k)$. Aus diesem Grund versucht man bei der Quasi-Newton-Methode die *inverse* Hessematrix iterativ zu bestimmen. Für das Weitere sei angenommen, dass die Kostenfunktion $f \in C^2$ ist, und es wird die Abkürzung $\mathbf{g}_k = (\nabla f)(\mathbf{x}_k)$ verwendet. Aus einer Taylorreihenentwicklung folgt die Näherung

$$\mathbf{g}_{k+1} - \mathbf{g}_k \approx (\nabla^2 f)(\mathbf{x}_k) \mathbf{p}_k \quad (2.92)$$

mit $\mathbf{p}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$. Nimmt man nun an, dass die Hessematrix $(\nabla^2 f)(\mathbf{x}_k) = \mathbf{K}$ konstant ist, dann gilt

$$\mathbf{q}_k = \mathbf{g}_{k+1} - \mathbf{g}_k = \mathbf{K} \mathbf{p}_k . \quad (2.93)$$

Wenn n linear unabhängige Vektoren $\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_{n-1}$ mit den zugehörigen $\mathbf{q}_0, \mathbf{q}_1, \dots, \mathbf{q}_{n-1}$ zur Verfügung stehen, dann lässt sich die Hessematrix in der Form

$$\mathbf{K} = [\mathbf{q}_0 \ \mathbf{q}_1 \ \dots \ \mathbf{q}_{n-1}] [\mathbf{p}_0 \ \mathbf{p}_1 \ \dots \ \mathbf{p}_{n-1}]^{-1} \quad (2.94)$$

berechnen. Das Ziel ist es nun, unter der Annahme einer konstanten Hessematrix $\mathbf{K}$ in n Iterationsschritten die inverse Hessematrix $\mathbf{K}^{-1}$ iterativ in der Form

$$\mathbf{H}_{k+1} \mathbf{q}_j = \mathbf{p}_j, \quad j = 0, \dots, k \quad (2.95)$$

zu konstruieren, woraus sich $\mathbf{H}_n = \mathbf{K}^{-1}$ ergibt. Diese iterative Konstruktion kann auf unterschiedliche Art und Weise erfolgen. Eine mögliche Variante wird im Folgenden beschrieben. Da die Hessematrix und ihre Inverse symmetrisch sind, ist es naheliegend, auch eine symmetrische Matrix für die Rekursion

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \mathbf{z}_k \mathbf{z}_k^T \quad (2.96)$$

anzusetzen. Das dyadische Produkt $\mathbf{z}_k \mathbf{z}_k^T$ erhält die Symmetrie und hat (für $\mathbf{z}_k \neq \mathbf{0}$) den Rang 1, weshalb diese Korrektur auch als *Rang 1 Korrektur* bezeichnet wird. Setzt man (2.96) in (2.95) ein, so erhält man für $j = k$

$$\mathbf{p}_k = \mathbf{H}_{k+1} \mathbf{q}_k = \mathbf{H}_k \mathbf{q}_k + \mathbf{z}_k \mathbf{z}_k^T \mathbf{q}_k \quad (2.97)$$

und nach Multiplikation mit $\mathbf{q}_k$ das Skalarprodukt

$$\mathbf{q}_k^T \mathbf{p}_k = \mathbf{q}_k^T \mathbf{H}_k \mathbf{q}_k + (\mathbf{z}_k^T \mathbf{q}_k)^2 . \quad (2.98)$$

Aus (2.97) und (2.98) folgt

$$\begin{aligned} (\mathbf{p}_k - \mathbf{H}_k \mathbf{q}_k)(\mathbf{p}_k - \mathbf{H}_k \mathbf{q}_k)^T &= \mathbf{z}_k \mathbf{z}_k^T \mathbf{q}_k (\mathbf{z}_k \mathbf{z}_k^T \mathbf{q}_k)^T = \mathbf{z}_k \mathbf{z}_k^T \mathbf{q}_k \mathbf{q}_k^T \mathbf{z}_k \mathbf{z}_k^T \\ &= \mathbf{z}_k \mathbf{z}_k^T (\mathbf{z}_k^T \mathbf{q}_k)^2 = \mathbf{z}_k \mathbf{z}_k^T \mathbf{q}_k^T (\mathbf{p}_k - \mathbf{H}_k \mathbf{q}_k) . \end{aligned} \quad (2.99)$$

Wird der daraus resultierende Ausdruck für $\mathbf{z}_k \mathbf{z}_k^T$ in (2.96) eingesetzt, so ergibt sich

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \frac{(\mathbf{p}_k - \mathbf{H}_k \mathbf{q}_k)(\mathbf{p}_k - \mathbf{H}_k \mathbf{q}_k)^T}{\mathbf{q}_k^T (\mathbf{p}_k - \mathbf{H}_k \mathbf{q}_k)} . \quad (2.100)$$

Damit lässt sich folgender Satz formulieren.

Satz 2.14 (Quasi-Newton-Methode mit Rang 1 Korrektur). Angenommen $\mathbf{K}$ ist eine konstante symmetrische Matrix und $\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_k$ sind linear unabhängige Vektoren. Mit $\mathbf{q}_j = \mathbf{K} \mathbf{p}_j$, $j = 0, \dots, k$ gilt für jede symmetrische Startmatrix $\mathbf{H}_0$ und die Iterationsvorschrift

$$\mathbf{H}_{j+1} = \mathbf{H}_j + \frac{(\mathbf{p}_j - \mathbf{H}_j \mathbf{q}_j)(\mathbf{p}_j - \mathbf{H}_j \mathbf{q}_j)^T}{\mathbf{q}_j^T (\mathbf{p}_j - \mathbf{H}_j \mathbf{q}_j)} \quad (2.101)$$

die Beziehung

$$\mathbf{p}_j = \mathbf{H}_{k+1} \mathbf{q}_j, \quad j = 0, \dots, k. \quad (2.102)$$

Aufgabe 2.12. Beweisen Sie Satz 2.14 mit vollständiger Induktion. **Hinweis:** Dieser Beweis ist z. B. in [2.9] skizziert.

Ein zentraler Nachteil der Rang 1 Korrektur ist, dass die positive Definitheit von $\mathbf{H}_{k+1}$ nur gesichert ist, wenn $\mathbf{q}_k^T(\mathbf{p}_k - \mathbf{H}_k \mathbf{q}_k) > 0$ gilt. Aus diesem Grund wurden weitere iterative Korrekturformeln für $\mathbf{H}_k$ entwickelt. Beispiele dafür sind die Korrekturformel nach Davidon-Fletcher-Powell (DFP)

$$\mathbf{H}_{k+1} = \left(\mathbf{E} - \frac{\mathbf{H}_k \mathbf{q}_k \mathbf{q}_k^T}{\mathbf{q}_k^T \mathbf{H}_k \mathbf{q}_k} \right) \mathbf{H}_k \left(\mathbf{E} - \frac{\mathbf{q}_k \mathbf{q}_k^T \mathbf{H}_k}{\mathbf{q}_k^T \mathbf{H}_k \mathbf{q}_k} \right) + \frac{\mathbf{p}_k \mathbf{p}_k^T}{\mathbf{q}_k^T \mathbf{p}_k} \quad (2.103)$$

und die etwas häufiger verwendete Korrekturformel nach Broyden-Fletcher-Goldfarb-Shanno (BFGS)

$$\mathbf{H}_{k+1} = \left(\mathbf{E} - \frac{\mathbf{p}_k \mathbf{q}_k^T}{\mathbf{q}_k^T \mathbf{p}_k} \right) \mathbf{H}_k \left(\mathbf{E} - \frac{\mathbf{q}_k \mathbf{p}_k^T}{\mathbf{q}_k^T \mathbf{p}_k} \right) + \frac{\mathbf{p}_k \mathbf{p}_k^T}{\mathbf{q}_k^T \mathbf{p}_k}. \quad (2.104)$$

Beide werden als *Rang 2 Korrekturformeln* bezeichnet, da die aktuelle Approximation der inversen Hessematrix jeweils durch Addition einer Matrix mit Rang 2 korrigiert wird.

Aufgabe 2.13. Zeigen Sie, dass die drei Iterationsvorschriften (2.101), (2.103) und (2.104) im Falle $\mathbf{p}_k = \mathbf{H}_k \mathbf{q}_k$ (d. h. $\mathbf{H}_k$ erfüllt bereits die Anforderung (2.95) an $\mathbf{H}_{k+1}$ für $j = k$) $\mathbf{H}_{k+1} = \mathbf{H}_k$ liefern.

Linearkombinationen der beiden Formeln (2.103) und (2.104) in der Art $\mathbf{H}_{k+1} = \phi \mathbf{H}_{k+1}^{\text{DFP}} + (1-\phi) \mathbf{H}_{k+1}^{\text{BFGS}}$ mit $\phi \in [0, 1]$ können ebenfalls verwendet werden. Alle so erhaltenen Rang 2 Korrekturformeln bilden die sogenannte *Broyden Familie*. Diese Korrekturformeln erhalten natürlich die Symmetrie von $\mathbf{H}_k$.

Basierend auf der aktuellen Schätzung $\mathbf{H}_k$ der inversen Hessematrix wird bei der Quasi-Newton-Methode die Suchrichtung in der Form

$$\mathbf{s}_k = -\mathbf{H}_k(\nabla f)(\mathbf{x}_k) \quad (2.105)$$

gewählt. Man beachte, dass hierfür, anders als bei der Newton-Methode (siehe (2.86)), lediglich die Kenntnis des Gradienten $(\nabla f)(\mathbf{x}_k)$ und eine Matrixmultiplikation von Nöten sind. Der Algorithmus der Quasi-Newton-Methode ist unter Verwendung der BFGS-Korrekturformel in Tabelle 2.3 zusammengefasst.

Es lässt sich zeigen (siehe [2.1, 2.4, 2.5]), dass die Korrekturformeln (2.103) und (2.104) die positive Definitheit von $\mathbf{H}_k$ erhalten, wenn

$$\mathbf{q}_k^T \mathbf{p}_k > 0 \quad (2.106)$$

gilt. Die Erfüllung dieser Bedingung lässt sich durch eine geeignete Wahl der Schrittweite $\alpha_k > 0$ in $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{s}_k$ bzw. $\mathbf{p}_k = \mathbf{x}_{k+1} - \mathbf{x}_k = \alpha_k \mathbf{s}_k$ sicherstellen. Genügt die Wahl der Schrittweite α_k beispielsweise der Wolfe-Bedingung gemäß Abschnitt 2.3.1.3, so ist

Initialisierung: $\mathbf{H}_0$ (Startwert, positiv definite Matrix)
 $k = 0$ (Startindex)
 $\mathbf{x}_0$ (Startlösung)
 $\mathbf{g}_0 = (\nabla f)(\mathbf{x}_0)$ (Gradient an der Stelle $\mathbf{x}_0$)
 ε_x (Schwellwert für Abbruchkriterium)

repeat

Schritt 1: Berechne die Suchrichtung $\mathbf{s}_k = -\mathbf{H}_k \mathbf{g}_k$

Schritt 2: Löse die Minimierungsaufgabe $\min_{\alpha_k \geq 0} f(\mathbf{x}_k + \alpha_k \mathbf{s}_k)$

Schritt 3: $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{s}_k$

$$\mathbf{p}_k = \mathbf{x}_{k+1} - \mathbf{x}_k = \alpha_k \mathbf{s}_k$$

$$\mathbf{g}_{k+1} = (\nabla f)(\mathbf{x}_{k+1})$$

$$\mathbf{q}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$$

Schritt 4: BFGS-Korrekturformel

$$\mathbf{H}_{k+1} = \left(\mathbf{E} - \frac{\mathbf{p}_k \mathbf{q}_k^T}{\mathbf{q}_k^T \mathbf{p}_k} \right) \mathbf{H}_k \left(\mathbf{E} - \frac{\mathbf{q}_k \mathbf{p}_k^T}{\mathbf{q}_k^T \mathbf{p}_k} \right) + \frac{\mathbf{p}_k \mathbf{p}_k^T}{\mathbf{q}_k^T \mathbf{p}_k}$$

$$k \leftarrow k + 1$$

until $\|\mathbf{x}_{k+1} - \mathbf{x}_k\| \leq \varepsilon_x$

Tabelle 2.3: Quasi-Newton-Methode mit der BFGS-Korrekturformel.

(2.106) automatisch erfüllt. Um dies zu sehen, beachte man, dass aus (2.40)

$$g'(\alpha_k) = \mathbf{g}_{k+1}^T \mathbf{s}_k \geq \varepsilon_2 g'(0) = \varepsilon_2 \mathbf{g}_k^T \mathbf{s}_k \quad (2.107)$$

mit $0 < \varepsilon_2 < 1$ folgt. Daraus erhält man

$$\mathbf{q}_k^T \mathbf{p}_k = \alpha_k \mathbf{q}_k^T \mathbf{s}_k = \alpha_k (\mathbf{g}_{k+1} - \mathbf{g}_k)^T \mathbf{s}_k \geq \underbrace{\alpha_k (\varepsilon_2 - 1) \mathbf{g}_k^T \mathbf{s}_k}_{> 0}, \quad (2.108)$$

wobei die rechte Seite dieser Ungleichung (abseits des optimalen Punktes $\mathbf{x}^*$) strikt positiv sein muss, da $\mathbf{s}_k$ eine Abstiegsrichtung ist. Somit ist (2.106) erfüllt.

Für unbeschränkte nichtlineare Optimierungsprobleme mit konvexer Kostenfunktion $f(\mathbf{x})$ konvergiert die Quasi-Newton-Methode mit superlinearer Konvergenzordnung. Für das unbeschränkte quadratische Optimierungsproblem (2.61) konvergiert die Quasi-Newton-Methode nach spätestens n Iterationsschritten. Konvergiert die Methode in diesem Fall genau nach n Iterationsschritten, so kann gezeigt werden (siehe [2.1]), dass $\mathbf{H}_n = (\nabla^2 f)^{-1}(\mathbf{x}^*) = \mathbf{Q}^{-1}$, d. h. der Algorithmus liefert die exakte inverse Hessematrix.

Die Vor- und Nachteile der Quasi-Newton-Methode können wie folgt zusammengefasst werden:

+ einfaches Verfahren mit moderatem Rechenaufwand

- + nur der Gradient $(\nabla f)(\mathbf{x}_k)$ (Berechnungsaufwand $\mathcal{O}(n)$) wird benötigt
- + konvergiert bei quadratischen Optimierungsproblemen nach spätestens n Iterationsschritten
- + meist superlineares Konvergenzverhalten
- Matrix $\mathbf{H}_k$ muss gespeichert werden (Speicherplatzbedarf $\mathcal{O}(n^2)$)

2.3.2.5 Gauss-Newton-Methode

Bei der Gauss-Newton-Methode wird die aufwändige Berechnung der Hessematrix $(\nabla^2 f)(\mathbf{x}_k)$ durch eine weniger rechenintensive näherungsweise Berechnung ersetzt. Diese Optimierungsmethode ist nur anwendbar, wenn die Kostenfunktion $f(\mathbf{x})$ die Struktur

$$f(\mathbf{x}) = \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|_2^2 = \frac{1}{2} \sum_{i=1}^m r_i^2(\mathbf{x}) \quad (2.109)$$

mit einer beliebigen Funktion $\mathbf{r} : \mathbb{R}^n \rightarrow \mathbb{R}^m$ besitzt. Die Komponenten $r_i(\mathbf{x})$ von $\mathbf{r}(\mathbf{x})$ sollen $r_i \in C^2$ erfüllen. Die exakte Hessematrix von $f(\mathbf{x})$ lautet in diesem Fall

$$(\nabla^2 f)(\mathbf{x}) = (\nabla \mathbf{r})(\mathbf{x})(\nabla \mathbf{r})^T(\mathbf{x}) + \sum_{i=1}^m r_i(\mathbf{x}) (\nabla^2 r_i)(\mathbf{x}), \quad (2.110)$$

wobei die Spalten der Jacobi-Matrix $(\nabla \mathbf{r})(\mathbf{x}) \in \mathbb{R}^{n \times m}$ die Gradienten $(\nabla r_i)(\mathbf{x})$ enthalten.

Aufgabe 2.14. Rechnen Sie (2.110) ausgehend von (2.109) nach.

Bei der Gauss-Newton-Methode wird der zweite Summand in (2.110) vernachlässigt, so dass sich die Näherung

$$(\nabla^2 f)(\mathbf{x}) \approx (\nabla \mathbf{r})(\mathbf{x})(\nabla \mathbf{r})^T(\mathbf{x}) \quad (2.111)$$

ergibt. Der damit verbundene Näherungsfehler ist also klein, wenn $r_i(\mathbf{x})$ oder $(\nabla^2 r_i)(\mathbf{x}) \forall i = 1, \dots, m$ betraglich kleine Werte annimmt. Ist z. B. bekannt, dass $f(\mathbf{x}^*) = 0$, so folgt daraus $r_i(\mathbf{x}^*) = 0 \forall i = 1, \dots, m$. Der Fall $(\nabla^2 r_i)(\mathbf{x}) = \mathbf{0} \forall i = 1, \dots, m$ tritt ein, wenn $\mathbf{r}(\mathbf{x})$ affin in $\mathbf{x}$ ist (vgl. Aufgabe 2.15).

Unter Verwendung des Gradienten $(\nabla f)(\mathbf{x}) = (\nabla \mathbf{r})(\mathbf{x})\mathbf{r}(\mathbf{x})$ und der Approximation (2.111) ergibt sich der Gauss-Newton-Schritt

$$\begin{aligned} \mathbf{s}_k &= -((\nabla \mathbf{r})(\mathbf{x}_k)(\nabla \mathbf{r})^T(\mathbf{x}_k))^{-1}(\nabla f)(\mathbf{x}_k) \\ &= -((\nabla \mathbf{r})(\mathbf{x}_k)(\nabla \mathbf{r})^T(\mathbf{x}_k))^{-1}(\nabla \mathbf{r})(\mathbf{x}_k)\mathbf{r}(\mathbf{x}_k). \end{aligned} \quad (2.112)$$

Die Iterationsvorschrift der Gauss-Newton-Methode lautet damit

$$\mathbf{x}_{k+1} = \mathbf{x}_k - ((\nabla \mathbf{r})(\mathbf{x}_k)(\nabla \mathbf{r})^T(\mathbf{x}_k))^{-1}(\nabla \mathbf{r})(\mathbf{x}_k)\mathbf{r}(\mathbf{x}_k). \quad (2.113)$$

Mit der Gauss-Newton-Methode wird also im Iterationsschritt k die lokale quadratische Approximation

$$f(\mathbf{x}_k) + (\nabla f)^T(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k) + \frac{1}{2}(\mathbf{x} - \mathbf{x}_k)^T(\nabla \mathbf{r})(\mathbf{x}_k)(\nabla \mathbf{r})^T(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k) \quad (2.114)$$

von $f(\mathbf{x})$ am Punkt $\mathbf{x}_k$ bezüglich $\mathbf{x}$ exakt minimiert. Für die praktische Anwendung der Methode führt man in (2.113) noch eine geeignete Schrittweite α_k ein, so dass die Iterationsvorschrift

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k ((\nabla \mathbf{r})(\mathbf{x}_k)(\nabla \mathbf{r})^T(\mathbf{x}_k))^{-1} (\nabla \mathbf{r})(\mathbf{x}_k) \mathbf{r}(\mathbf{x}_k) \quad (2.115)$$

lautet und die Abstiegsbedingung (2.29) erfüllt ist. Das Verfahren wird dann häufig *gedämpfte Gauss-Newton-Methode* genannt und α_k wird als *Dämpfungsparameter* bezeichnet. Gelegentlich wird die Einschränkung $\alpha_k \leq 1$ verwendet.

Man beachte, dass die Berechnung von $\mathbf{s}_k$ gemäß (2.112) keine tatsächliche Inversion von $(\nabla \mathbf{r})(\mathbf{x}_k)(\nabla \mathbf{r})^T(\mathbf{x}_k)$ erfordert. Die Wahl der Suchrichtung gemäß (2.112) ist nur sinnvoll, wenn die $n \times n$ Matrix $(\nabla \mathbf{r})(\mathbf{x})(\nabla \mathbf{r})^T(\mathbf{x})$ positiv definit ist. Ist sie nicht positiv definit (oder schlecht konditioniert), kann ähnlich wie bei der Newton-Methode die alternative Iterationsvorschrift

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{M}_k (\nabla f)(\mathbf{x}_k), \quad \mathbf{M}_k = \left((\nabla \mathbf{r})(\mathbf{x}_k)(\nabla \mathbf{r})^T(\mathbf{x}_k) + \varepsilon_k \mathbf{E} \right)^{-1} \quad (2.116)$$

mit einem geeigneten positiven Parameter $\varepsilon_k \geq 0$ verwendet werden. Das Verfahren wird dann (für den Fall $\alpha_k = 1$) als *Levenberg-Marquardt-Methode* bezeichnet. Die Iterationsvorschrift (2.116) geht für $\varepsilon_k \rightarrow 0$ in die Gauss-Newton-Methode gemäß (2.115) und für sehr große ε_k in die Gradientenmethode gemäß (2.58) über. Die Levenberg-Marquardt-Methode kombiniert daher das Verhalten der Gauss-Newton-Methode mit jenem der Gradientenmethode. Häufig wird dabei der Parameter ε_k ausgehend von einem hohen Anfangswert heuristisch adaptiert. Dazu wird die tatsächliche Kostenfunktionsänderung $f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k)$ betrachtet. Ist sie positiv (kein Abstieg), so wird der Schritt verworfen und ε_k vergrößert (Verhalten wird der Gradientenmethode ähnlicher). Ist sie negativ, so wird der Schritt akzeptiert. Stimmt sie überdies gut mit der gemäß (2.114) prädizierten (negativen) Kostenfunktionsänderung $(\nabla f)^T(\mathbf{x}_k)(\mathbf{x}_{k+1} - \mathbf{x}_k) + \frac{1}{2}(\mathbf{x}_{k+1} - \mathbf{x}_k)^T (\nabla \mathbf{r})(\mathbf{x}_k)(\nabla \mathbf{r})^T(\mathbf{x}_k)(\mathbf{x}_{k+1} - \mathbf{x}_k)$ überein, wird ε_k verkleinert (Verhalten wird der Gauss-Newton-Methode ähnlicher). Eine detaillierter dargestellte, ähnliche Heuristik findet sich in Abschnitt 2.4.

Der nachfolgende Satz liefert eine Aussage über die Konvergenzordnung der Gauss-Newton-Methode. Sein Beweis ist z. B. in [2.10] zu finden.

Satz 2.15 (Konvergenzordnung der Gauss-Newton-Methode). *Gegeben sei die Kostenfunktion $f(\mathbf{x}) = \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|_2^2$ mit $\mathbf{r} \in C^2$ definiert im $\mathbb{R}^n$ und dem lokalen Minimum $\mathbf{x}^*$. Wenn die Jacobi-Matrix $(\nabla \mathbf{r})(\mathbf{x}^*)$ zeilenregulär ist, $\sum_{i=1}^m r_i(\mathbf{x}^*)(\nabla^2 r_i)(\mathbf{x}^*) = \mathbf{0}$ gilt und der Anfangswert $\mathbf{x}_0$ in einer hinreichend nahen Umgebung des Minimums liegt, dann konvergiert die Gauss-Newton-Iteration*

$$\mathbf{x}_{k+1} = \mathbf{x}_k - ((\nabla \mathbf{r})(\mathbf{x}_k)(\nabla \mathbf{r})^T(\mathbf{x}_k))^{-1} (\nabla \mathbf{r})(\mathbf{x}_k) \mathbf{r}(\mathbf{x}_k) \quad (2.117)$$

mit der Konvergenzordnung 2 gegen das Minimum $\mathbf{x}^*$.

Ist der Term $\sum_{i=1}^m r_i(\mathbf{x}^*)(\nabla^2 r_i)(\mathbf{x}^*) \neq \mathbf{0}$ aber klein gegenüber $(\nabla \mathbf{r})(\mathbf{x}_k)(\nabla \mathbf{r})^T(\mathbf{x}_k)$, so kann zumindest lineares Konvergenzverhalten nachgewiesen werden. Wenn jedoch $\sum_{i=1}^m r_i(\mathbf{x}^*)(\nabla^2 r_i)(\mathbf{x}^*)$ betragsmäßig zu große Werte annimmt, kann es sein, dass die Gauss-Newton-Methode nicht konvergiert [2.10].

Aufgabe 2.15. Zeigen Sie, dass die Gauss-Newton-Methode für die Optimierungsaufgabe

$$\min_{\mathbf{x} \in \mathbb{R}^n} \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|_2^2 \quad \text{mit} \quad \mathbf{r}(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b} \quad (2.118)$$

und einer spaltenregulären Matrix $\mathbf{A}$ (dies impliziert $m \geq n$) unabhängig vom Startpunkt $\mathbf{x}_0$ innerhalb von nur einem Iterationsschritt zum optimalen Punkt $\mathbf{x}^* = -(\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{b}$ konvergiert.

Kostenfunktionen wie in der Optimierungsaufgabe (2.118) treten z. B. bei der Methode der kleinsten Fehlerquadrate mit parametrisch linearem Modell (lineare Regressionsanalyse) auf. Kostenfunktionen der Art (2.109) treten z. B. bei der Methode der kleinsten Fehlerquadrate mit parametrisch nichtlinearem Modell (nichtlineare Regressionsanalyse) [2.11] auf.

Aufgabe 2.16. Zeigen Sie, dass die iterative Lösung der nichtlinearen Gleichung

$$\mathbf{r}(\mathbf{x}) = \mathbf{0} \quad (2.119)$$

mit $\mathbf{r} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ und $\mathbf{r} \in C^1$ gemäß dem Newton-Raphson-Verfahren genau der Gauss-Newton-Methode angewandt auf die Optimierungsaufgabe

$$\min_{\mathbf{x} \in \mathbb{R}^n} \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|_2^2 \quad (2.120)$$

entspricht.

Lösung von Aufgabe 2.16. Damit die Gleichung (2.119) eine eindeutige Lösung $\mathbf{x}^*$ besitzt, muss $(\nabla \mathbf{r})^T(\mathbf{x}^*)$ regulär sein. Dies muss wegen $\mathbf{r} \in C^1$ auch für $(\nabla \mathbf{r})^T(\mathbf{x}_k)$ in einer Umgebung von $\mathbf{x}^*$ gelten. Die Iterationsvorschrift beim Newton-Raphson-Verfahren lautet

$$\mathbf{x}_{k+1} = \mathbf{x}_k - ((\nabla \mathbf{r})^T(\mathbf{x}_k))^{-1} \mathbf{r}(\mathbf{x}_k) . \quad (2.121)$$

Die Iterationsvorschrift (2.113) der Gauss-Newton-Methode vereinfacht sich bei regulärem $(\nabla \mathbf{r})^T(\mathbf{x}_k)$ zu

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k - ((\nabla \mathbf{r})(\mathbf{x}_k)(\nabla \mathbf{r})^T(\mathbf{x}_k))^{-1} (\nabla \mathbf{r})(\mathbf{x}_k) \mathbf{r}(\mathbf{x}_k) \\ &= \mathbf{x}_k - ((\nabla \mathbf{r})^T(\mathbf{x}_k))^{-1} \mathbf{r}(\mathbf{x}_k) . \end{aligned} \quad (2.122)$$

Die Vor- und Nachteile der Gauss-Newton-Methode können wie folgt zusammengefasst werden:

- + keine zweiten Ableitungen nötig, nur die Jacobi-Matrix $(\nabla \mathbf{r})(\mathbf{x}_k)$ wird benötigt
- + konvergiert im ersten Iterationsschritt, wenn $\mathbf{r}(\mathbf{x})$ affin in $\mathbf{x}$ ist
- + unter bestimmten Voraussetzungen kann quadratisches Konvergenzverhalten erreicht werden
- Konvergenz hängt vom jeweiligen Problem ab und ist nicht garantiert

- Berechnungsaufwand $\mathcal{O}(mn)$ für die Jacobi-Matrix $(\nabla \mathbf{r})(\mathbf{x}_k)$, Berechnungsaufwand $\mathcal{O}(n^3)$ für die Suchrichtung $\mathbf{s}_k$

2.4 Methode der Vertrauensbereiche

Bei den Liniensuchverfahren wird eine geeignete Abstiegsrichtung (Suchrichtung) $\mathbf{s}_k$ (beispielsweise der *negative Gradient* an der Stelle $\mathbf{x}_k$ gemäß (2.41) bei der Gradientenmethode oder die *Newton-Richtung* gemäß (2.86) bei der Newton-Methode) gewählt und anschließend wird über das skalare Optimierungsproblem (2.30) die (optimale) Schrittweite $\alpha_k > 0$ in diese Abstiegsrichtung bestimmt. Bei der Methode der Vertrauensbereiche (englisch: *trust region method*) wird die zu minimierende Kostenfunktion $f(\mathbf{x})$ in der Umgebung von $\mathbf{x}_k$ durch eine quadratische Ansatzfunktion m_k in der Form

$$f(\mathbf{x}_k + \mathbf{s}_k) \approx m_k(\mathbf{s}_k) = f(\mathbf{x}_k) + \mathbf{s}_k^T (\nabla f)(\mathbf{x}_k) + \frac{1}{2} \mathbf{s}_k^T \mathbf{B}_k \mathbf{s}_k \quad (2.123)$$

mit einer geeigneten symmetrischen Matrix $\mathbf{B}_k$ approximiert. Grundsätzlich sollte $\mathbf{B}_k$ eine mit geringem Aufwand berechenbare Näherung der Hessematrix $(\nabla^2 f)(\mathbf{x}_k)$ sein. Der mit $m_k(\mathbf{s}_k)$ verbundene Approximationsfehler ist in der Größenordnung von $\|\mathbf{s}_k\|^2$ und wenn $\mathbf{B}_k$ mit der Hessematrix $(\nabla^2 f)(\mathbf{x}_k)$ übereinstimmt sogar von $\|\mathbf{s}_k\|^3$. Der *Vertrauensbereich* wird durch $\|\mathbf{s}_k\|_2 \leq \Delta_k$ mit einem skalaren Parameter Δ_k charakterisiert und definiert eine Umgebung um den Punkt $\mathbf{x}_k$, in der die Kostenfunktion $f(\mathbf{x}_k + \mathbf{s}_k)$ hinreichend genau durch die quadratische Ansatzfunktion $m_k(\mathbf{s}_k)$ beschrieben wird. Es wird nun in jedem Iterationsschritt das *beschränkte* Optimierungsproblem

$$\min_{\mathbf{s}_k \in \mathbb{R}^n} \quad m_k(\mathbf{s}_k) = f(\mathbf{x}_k) + \mathbf{s}_k^T (\nabla f)(\mathbf{x}_k) + \frac{1}{2} \mathbf{s}_k^T \mathbf{B}_k \mathbf{s}_k \quad (2.124a)$$

$$\text{u.B.v.} \quad \|\mathbf{s}_k\|_2 \leq \Delta_k \quad (2.124b)$$

für ein geeignetes $\Delta_k > 0$ gelöst und die Vorschrift $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$ verwendet. Man beachte, dass (im Gegensatz zu den meisten Liniensuchverfahren) die Abstiegsrichtung und die Schrittweite *gleichzeitig* bestimmt werden.

Ein wesentlicher Entwurfsfreiheitsgrad dieser Methode liegt nun in der Wahl von Δ_k . Dazu wird in jedem Iterationsschritt die *Übereinstimmung der quadratischen Ansatzfunktion m_k mit der Kostenfunktion f* überprüft, indem das Verhältnis

$$\rho_k(\mathbf{s}_k) = \frac{f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{s}_k)}{m_k(\mathbf{0}) - m_k(\mathbf{s}_k)} \quad (2.125)$$

berechnet wird. Der Zählerterm in (2.125) beschreibt die *tatsächliche Reduktion* der Kostenfunktion, während der Nennerterm die *prädizierte Reduktion* wiedergibt. Dieser Nennerterm ist stets größer gleich Null, da $\mathbf{s}_k$ die Funktion m_k gemäß (2.124) innerhalb des Vertrauensbereiches minimiert. Ist nun $\rho_k(\mathbf{s}_k) < 0$, so bedeutet dies, dass der Wert der Kostenfunktion am nächsten Iterationspunkt $f(\mathbf{x}_k + \mathbf{s}_k)$ größer als am vorigen Iterationspunkt $f(\mathbf{x}_k)$ wäre, weshalb dieser Iterationsschritt verworfen und der Vertrauensbereich verkleinert werden muss. Andererseits kann bei $\rho_k(\mathbf{s}_k) \approx 1$ der Vertrauensbereich vergrößert werden, da die Kostenfunktion $f(\mathbf{x}_k)$ in diesem Fall gut von der Ansatzfunktion

Initialisierung:	$\bar{\Delta}, \Delta_0 \in (0, \bar{\Delta})$	(Vertrauensbereich: Grenz- & Startwert)
	$\eta \in [0, \frac{1}{4})$	(Parameter)
	$k \leftarrow 0$	(Startindex)
	$\mathbf{x}_0$	(Startlösung)
	ε_x	(Schwellwert für Abbruchkriterium)
repeat		
	$m_k(\mathbf{s}_k)$ nach (2.123)	(Modell)
	$\mathbf{s}_k$ Lösung von (2.124)	(evtl. approximativ gelöst)
	ρ_k nach (2.125)	(Modellgüte)
	if $\rho_k < \frac{1}{4}$	
	$\Delta_{k+1} \leftarrow \frac{1}{4} \Delta_k$	(Reduktion)
	else if $\rho_k > \frac{3}{4}$ and $\ \mathbf{s}_k\ _2 = \Delta_k$	
	$\Delta_{k+1} \leftarrow \min\{2\Delta_k, \bar{\Delta}\}$	(Vergrößerung)
	else	
	$\Delta_{k+1} \leftarrow \Delta_k$	
	end if	
	if $\rho_k > \eta$	
	$\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \mathbf{s}_k$	(nächster Schritt)
	$\mathbf{B}_{k+1} \leftarrow \mathbf{B}_k + \dots$	(Approximation der Hessematrix)
	else	
	$\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k$	(Schritt mit $\Delta_{k+1} < \Delta_k$ wiederholen)
	end if	
	$k \leftarrow k + 1$	
until	$\ \mathbf{x}_k - \mathbf{x}_{k-1}\ _2 \leq \varepsilon_x$	

Tabelle 2.4: Methode der Vertrauensbereiche.

beschrieben wird. Für den Fall, dass $\rho_k(\mathbf{s}_k)$ positiv und deutlich kleiner als 1 ist, wird der Vertrauensbereich im nächsten Schritt verkleinert.

Ein Algorithmus zur Methode der Vertrauensbereiche ist in Tabelle 2.4 aufgelistet. Man beachte, dass hier $\bar{\Delta}$ eine obere Schranke für Δ_k darstellt und dass eine Vergrößerung des Vertrauensbereiches im nächsten Iterationsschritt nur dann stattfindet, wenn $\mathbf{s}_k$ durch die Grenze des Vertrauensbereiches beschränkt wurde (Bedingung $\|\mathbf{s}_k\|_2 = \Delta_k$).

Vorschläge für die in Tabelle 2.4 nicht dargestellte Iterationsvorschrift der Matrix $\mathbf{B}_k$ werden z. B. in [2.1, 2.5, 2.9] beschrieben. Wenn sich die Kostenfunktion $f(\mathbf{x}_k)$ in der Form (2.109), d. h. $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{r}(\mathbf{x})\|_2^2$ darstellen lässt, kann die Berechnung von $\mathbf{B}_k$ wie bei der Gauss-Newton-Methode in der Form (2.111) erfolgen, d. h. $\mathbf{B}_k = (\nabla \mathbf{r})(\mathbf{x}_k)(\nabla \mathbf{r})^T(\mathbf{x}_k)$. Es

wird nun gezeigt, dass bei dieser Wahl die Methode der Vertrauensbereiche exakt mit der Levenberg-Marquardt-Methode übereinstimmt [2.10]. Dazu wird die Ungleichungsbeschränkung in (2.124) äquivalent umgeschrieben, womit sich das Optimierungsproblem

$$\min_{\mathbf{s}_k \in \mathbb{R}^n} \quad m_k(\mathbf{s}_k) = f(\mathbf{x}_k) + \mathbf{s}_k^T (\nabla f)(\mathbf{x}_k) + \frac{1}{2} \mathbf{s}_k^T (\nabla \mathbf{r})(\mathbf{x}_k) (\nabla \mathbf{r})^T(\mathbf{x}_k) \mathbf{s}_k \quad (2.126a)$$

$$\text{u.B.v.} \quad h(\mathbf{s}_k) = \frac{1}{2} (\mathbf{s}_k^T \mathbf{s}_k - \Delta_k^2) \leq 0 \quad (2.126b)$$

ergibt. Basierend auf Satz ?? lauten notwendige Optimalitätsbedingungen für das beschränkte Optimierungsproblem (2.126)

$$(\nabla f)(\mathbf{x}_k) + (\nabla \mathbf{r})(\mathbf{x}_k) (\nabla \mathbf{r})^T(\mathbf{x}_k) \mathbf{s}_k^* + \varepsilon_k^* \mathbf{s}_k^* = \mathbf{0} \quad (2.127a)$$

$$\varepsilon_k^* \geq 0 \quad (2.127b)$$

$$\varepsilon_k^* h(\mathbf{s}_k^*) = 0 \quad (2.127c)$$

mit einem noch zu bestimmenden Multiplikator ε_k^* . Aus (2.126b) und (2.127) lassen sich $\mathbf{s}_k^*$ und ε_k^* eindeutig berechnen. Für $\mathbf{s}_k^*$ folgt

$$\mathbf{s}_k^* = - \left((\nabla \mathbf{r})(\mathbf{x}_k) (\nabla \mathbf{r})^T(\mathbf{x}_k) + \varepsilon_k^* \mathbf{E} \right)^{-1} (\nabla f)(\mathbf{x}_k), \quad (2.128)$$

was genau der Levenberg-Marquardt-Methode (vgl. (2.116) mit $\alpha_k = 1$) entspricht. Bei der Methode der Vertrauensbereiche wird der Parameter Δ_k gewählt. Bei der Levenberg-Marquardt-Methode wird der Parameter ε_k gewählt. Diese Parameter lassen sich über (2.126b) und (2.127) ineinander umrechnen und es gilt $\varepsilon_k = 0$ für $\Delta_k \rightarrow \infty$ sowie $\varepsilon_k \rightarrow \infty$ für $\Delta_k \rightarrow 0$.

Mit der in Tabelle 2.5 zusammengefassten sogenannten *Dogleg-Methode* wird eine näherungsweise Lösung des Optimierungsproblems (2.124) berechnet. Sie kombiniert dabei das Verhalten der Newton-Methode mit jenem der Gradientenmethode. Wenn mit einem unbeschränkten Gradientenschritt der Vertrauensbereich verlassen würde, wird $\mathbf{s}_k$ durch Einschränkung des Gradientenschrittes auf den Vertrauensbereich gewählt. Wenn mit einem unbeschränkten Newtonschritt der Vertrauensbereich nicht verlassen wird, wird $\mathbf{s}_k$ entsprechend diesem Schritt gewählt. Andernfalls wird zwischen dem unbeschränkten Gradientenschritt und dem unbeschränkten Newtonschritt genau so linear interpoliert, dass $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$ am Rand des Vertrauensbereiches zu liegen kommt.

Für ein Beispiel mit $n = 2$ zeigt Abbildung 2.9 die Niveaulinien einer lokalen quadratischen Ansatzfunktion $m_k(\mathbf{s}_k)$ in einer Umgebung von $\mathbf{x}_k$. Die grüne Linie beinhaltet die Näherungslösungen $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$ der Dogleg Methode für verschiedene Werte $\Delta_k \in (0, \infty)$. Zum Vergleich zeigt die rote Linie die exakten Lösungen des Optimierungsproblems (2.124) für verschiedene Werte $\Delta_k \in (0, \infty)$. Im Fall $\mathbf{B}_k = (\nabla \mathbf{r})(\mathbf{x}_k) (\nabla \mathbf{r})^T(\mathbf{x}_k)$ stimmt die rote Linie natürlich auch mit den Lösungen der Levenberg-Marquardt-Methode überein.

Bemerkung 2.1. Die Lösungen einer nichtlinearen Gleichung

$$\mathbf{r}(\mathbf{x}) = \mathbf{0} \quad (2.129)$$

$m_k(\mathbf{s}_k)$ nach (2.123)	(Modell)
$\mathbf{s}_{G,k} = -\frac{(\nabla f)(\mathbf{x}_k)^T (\nabla f)(\mathbf{x}_k)}{(\nabla f)(\mathbf{x}_k)^T \mathbf{B}_k (\nabla f)(\mathbf{x}_k)} (\nabla f)(\mathbf{x}_k)$	(unbeschränkter Gradientenschritt)
if $\ \mathbf{s}_{G,k}\ _2 \geq \Delta_k$	
$\mathbf{s}_k = \Delta_k \frac{\mathbf{s}_{G,k}}{\ \mathbf{s}_{G,k}\ _2}$	
else	
$\mathbf{s}_{N,k} = -\mathbf{B}_k^{-1} (\nabla f)(\mathbf{x}_k)$	(unbeschränkter Newtonschritt)
if $\ \mathbf{s}_{N,k}\ _2 \leq \Delta_k$	
$\mathbf{s}_k = \mathbf{s}_{N,k}$	
else	
$\mathbf{s}_k = \beta_k \mathbf{s}_{G,k} + (1 - \beta_k) \mathbf{s}_{N,k}$	(Interpolation)
mit $\beta_k \in (0, 1)$ so, dass $\ \mathbf{s}_k\ _2 = \Delta_k$	
end if	
end if	

Tabelle 2.5: Näherungsweise Lösung von (2.124) mittels Dogleg-Methode.

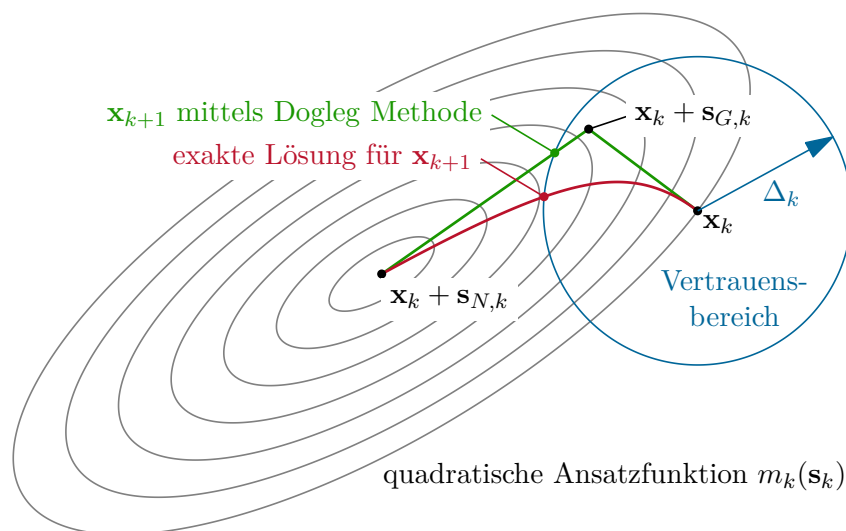


Abbildung 2.9: Lösung des Optimierungsproblems (2.124) im Rahmen einer Iteration der Methode der Vertrauensbereiche, Vergleich zwischen exakter Lösung (rot) und Dogleg Methode (grün).

mit $\mathbf{r} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ können durch Lösen der Optimierungsaufgabe

$$\min_{\mathbf{x} \in \mathbb{R}^n} \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|_2^2 \quad (2.130)$$

berechnet werden (siehe dazu auch Aufgabe 2.16). Diese Möglichkeit zur Lösung nichtlinearer Gleichungen wird von der MATLAB-Funktion `fsolve` aus der Optimization Toolbox genutzt. Die Funktion `fsolve` verwendet standardmäßig die Methode der Vertrauensbereiche, wobei zwischen der Basisvariante, der Dogleg-Methode und der Levenberg-Marquardt-Methode ausgewählt werden kann (siehe Dokumentation der Funktion in MATLAB).

2.5 Direkte Suchverfahren

Die bisher betrachteten sogenannten *ableitungsbehafteten* Lösungsverfahren verwenden den Gradienten $(\nabla f)(\mathbf{x}_k)$ (und mitunter die Hessematrix $(\nabla^2 f)(\mathbf{x}_k)$), um mittels einer geeigneten Iterationsvorschrift einen neuen Punkt $\mathbf{x}_{k+1}$ zu bestimmen. Es soll dabei eine hinreichend hohe Reduktion der Kostenfunktion $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$ erreicht werden.

Allerdings sind in manchen praktischen Fällen die dazu erforderlichen Ableitungen nicht verfügbar oder nicht mit vertretbarem Aufwand berechenbar, da das betrachtete Problem *zu komplex* oder *nicht stetig differenzierbar* ist. Abhilfe verschaffen in diesem Fall sogenannte *direkte* oder *ableitungsfreie Suchverfahren*, die mit Hilfe von Stichproben eine Reihe von Funktionswerten berechnen, um daraus einen neuen Iterationspunkt $\mathbf{x}_{k+1}$ zu bestimmen.

Ein bekanntes und gleichzeitig einfaches Verfahren in der nichtlinearen Optimierung ist das *Simplex-Verfahren* nach *Nelder* und *Mead*. Dieses heuristische Verfahren unterscheidet sich grundsätzlich vom Simplex-Algorithmus in der *Linearen Programmierung* und sollte nicht mit ihm verwechselt werden.

Der Algorithmus basiert auf der Weiterentwicklung eines sogenannten *Simplex* im n -dimensionalen Raum der Optimierungsvariablen. Unter einem Simplex versteht man in diesem Zusammenhang jene konvexe Hülle, die von $n + 1$ Punkten $\mathbf{x}_{k,i}$, $i = 0, \dots, n$ zum Iterationsschritt k im n -dimensionalen Suchraum aufgespannt wird (für $n = 1$ ist dies eine Linie, für $n = 2$ ein Dreieck, etc.). Im Weiteren werden mit $\mathbf{x}_{k,\min}$ und $\mathbf{x}_{k,\max}$ jene Eckpunkte des Simplex bezeichnet welche den kleinsten und größten Kostenfunktionswert f aufweisen, d. h. es gilt

$$\begin{aligned} \mathbf{x}_{k,\min} &= \arg \min_{i=0,\dots,n} f(\mathbf{x}_{k,i}) \\ \mathbf{x}_{k,\max} &= \arg \max_{i=0,\dots,n} f(\mathbf{x}_{k,i}) . \end{aligned} \quad (2.131)$$

Der *Schwerpunkt* $\hat{\mathbf{x}}_k$ aller Eckpunkte des Simplex ohne $\mathbf{x}_{k,\max}$ errechnet sich zu

$$\hat{\mathbf{x}}_k = \frac{1}{n} \left(\sum_{i=0}^n \mathbf{x}_{k,i} - \mathbf{x}_{k,\max} \right) . \quad (2.132)$$

Der Algorithmus beruht nun auf der Idee, den Punkt $\mathbf{x}_{k,\max}$ im Simplex durch einen

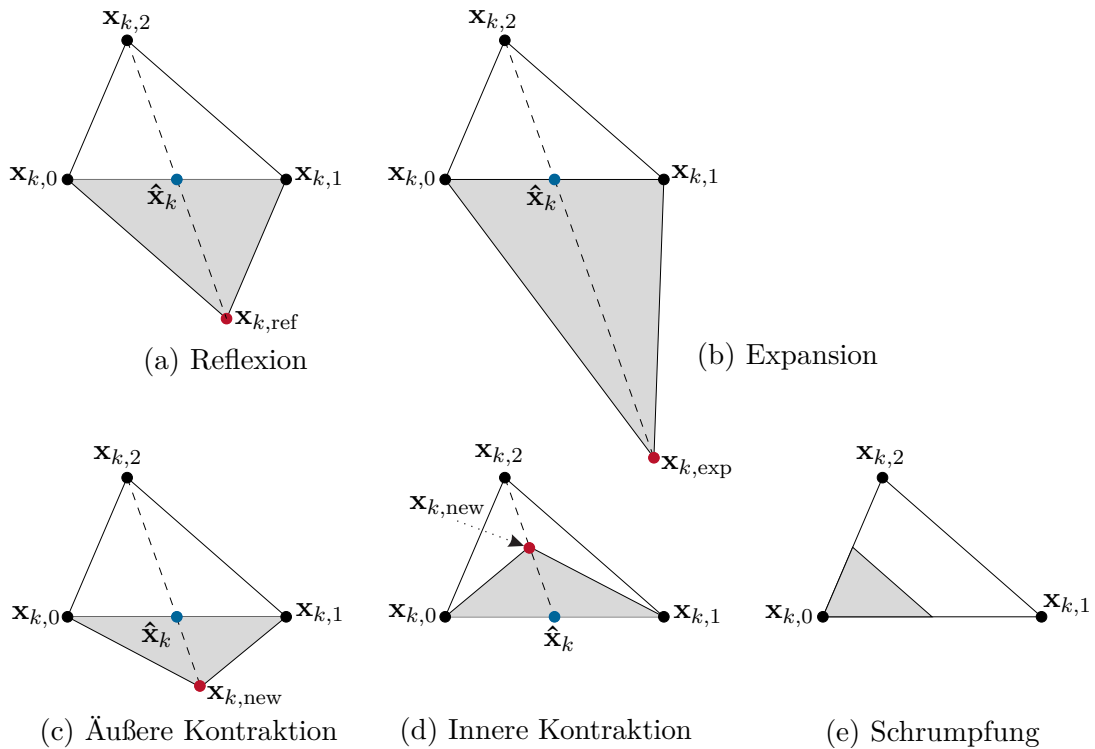


Abbildung 2.10: Operationen des Simplex-Verfahrens nach Nelder und Mead für $\beta = 1$, $\gamma = 1$ und $\theta = 1/2$.

anderen Punkt mit einem niedrigeren Kostenfunktionswert zu ersetzen. Eine wichtige Operation dabei ist die Berechnung des *Reflexionspunktes*

$$\mathbf{x}_{k,\text{ref}} = \hat{\mathbf{x}}_k + (\hat{\mathbf{x}}_k - \mathbf{x}_{k,\text{max}}), \quad (2.133)$$

der auf einer Geraden durch die Punkte $\mathbf{x}_{k,\text{max}}$ und $\hat{\mathbf{x}}_k$ liegt und symmetrisch bezüglich $\hat{\mathbf{x}}_k$ zu $\mathbf{x}_{k,\text{max}}$ ist, siehe Abbildung 2.10(a). Abhängig von $f(\mathbf{x}_{k,\text{ref}})$ im Vergleich zu den Funktionswerten der anderen Punkte des Simplex wird ein neuer Punkt $\mathbf{x}_{k,\text{new}}$ konstruiert, der im nächsten Iterationsschritt den Punkt $\mathbf{x}_{k,\text{max}}$ ersetzt. Der Algorithmus ist in seiner Grundfunktion in Tabelle 2.6 aufgelistet und die unterschiedlichen Operationen sind in Abbildung 2.10 visualisiert. Man beachte, dass die Schrumpfung im Algorithmus von Tabelle 2.6 stets bezüglich des Eckpunktes $\mathbf{x}_{k,\text{min}}$ mit dem kleinsten Kostenfunktionswert ausgeführt wird, d. h. in Abbildung 2.10(e) gilt $\mathbf{x}_{k,0} = \mathbf{x}_{k,\text{min}}$. Während der Iteration wandert der Simplex in Richtung des Optimums und zieht sich sukzessive zusammen. Allerdings ist die Konvergenz zu einem optimalen Punkt im Allgemeinen nicht garantiert und es kann vorkommen, dass das Simplex-Verfahren zu einem *nicht-optimalen Punkt* konvergiert. In der Praxis führt das Simplex-Verfahren dennoch häufig zu guten Ergebnissen und akzeptablem Konvergenzverhalten.

Initialisierung:	$\mathbf{x}_{0,i}, i = 0, \dots, n$	(Startsimplex)
	$k \leftarrow 0$	(Iterationsindex)
	$\beta > 0$	(Reflexionskoeffizient, typisch $\beta = 1$)
	$\gamma > 0$	(Expansionskoeffizient, typisch $\gamma = 1$)
	$\theta \in (0, 1)$	(Kontraktionskoeffizient, typisch $\theta = 1/2$)
	ε_x	(Schwellwert für Abbruchkriterium)
repeat		
	$\mathbf{x}_{k,\min}, \mathbf{x}_{k,\max}$ gemäß (2.131)	(Punkte mit min. und max. Kostenfunktionswert)
	$\hat{\mathbf{x}}_k$ gemäß (2.132)	(Schwerpunkt)
	$\mathbf{x}_{k,\text{ref}} = \hat{\mathbf{x}}_k + \beta(\hat{\mathbf{x}}_k - \mathbf{x}_{k,\max})$	(Reflexionsschritt, Abb. 2.10(a))
	if $f(\mathbf{x}_{k,\text{ref}}) < f(\mathbf{x}_{k,\min})$	
	$\mathbf{x}_{k,\text{exp}} = \mathbf{x}_{k,\text{ref}} + \gamma(\mathbf{x}_{k,\text{ref}} - \hat{\mathbf{x}}_k)$	(Expansionsschritt, Abb. 2.10(b))
	if $f(\mathbf{x}_{k,\text{exp}}) < f(\mathbf{x}_{k,\text{ref}})$	
	$\mathbf{x}_{k,\text{new}} = \mathbf{x}_{k,\text{exp}}$	
	else	
	$\mathbf{x}_{k,\text{new}} = \mathbf{x}_{k,\text{ref}}$	(Reflexionspunkt beibehalten)
	end if	
	else if $f(\mathbf{x}_{k,\text{ref}}) > \max_{\mathbf{x}_{k,i} \neq \mathbf{x}_{k,\max}, i=0,\dots,n} f(\mathbf{x}_{k,i})$	
	if $f(\mathbf{x}_{k,\max}) \leq f(\mathbf{x}_{k,\text{ref}})$	
	$\mathbf{x}_{k,\text{new}} = \theta \mathbf{x}_{k,\max} + (1 - \theta) \hat{\mathbf{x}}_k$	(Innere Kontraktion, Abb. 2.10(d))
	else	
	$\mathbf{x}_{k,\text{new}} = \theta \mathbf{x}_{k,\text{ref}} + (1 - \theta) \hat{\mathbf{x}}_k$	(Äußere Kontraktion, Abb. 2.10(c))
	end if	
	else	
	$\mathbf{x}_{k,\text{new}} = \mathbf{x}_{k,\text{ref}}$	(Reflexionspunkt beibehalten)
	end if	
	if $f(\mathbf{x}_{k,\text{new}}) \geq f(\mathbf{x}_{k,\max})$	(ev. bei nichtkonv. Kostenfkt.)
	$\mathbf{x}_{k+1,i} \leftarrow \frac{1}{2}(\mathbf{x}_{k,i} + \mathbf{x}_{k,\min}), i = 0, \dots, n$	(Schrumpfung, Abb. 2.10(e))
	else	
	$\mathbf{x}_{k,\max} \leftarrow \mathbf{x}_{k,\text{new}}$	
	$\mathbf{x}_{k+1,i} \leftarrow \mathbf{x}_{k,i}, i = 0, \dots, n$	
	end if	
	$k \leftarrow k + 1$	
until	$\max_{i=0,\dots,n} \ \mathbf{x}_{k,i} - \mathbf{x}_{k,\min}\ _\infty \leq \varepsilon_x$	

Tabelle 2.6: Simplex-Verfahren nach Nelder und Mead.

2.6 Beispiel: Rosenbrock's „Bananenfunktion“

Ein bekanntes Beispiel in der Optimierung ist das *Rosenbrock*-Problem

$$\min_{\mathbf{x} \in \mathbb{R}^2} f(\mathbf{x}) \quad \text{mit} \quad f(\mathbf{x}) = 100(x_2 - x_1^2)^2 + (x_1 - 1)^2. \quad (2.134)$$

Abbildung 2.11 zeigt die Form und die Höhenlinien der Funktion, die auch als *Bananenfunktion* bezeichnet wird. Das Rosenbrock-Problem soll als Beispiel verwendet werden, um die *Konvergenzeigenschaften der behandelten Verfahren* numerisch mit Hilfe von MATLAB zu untersuchen.

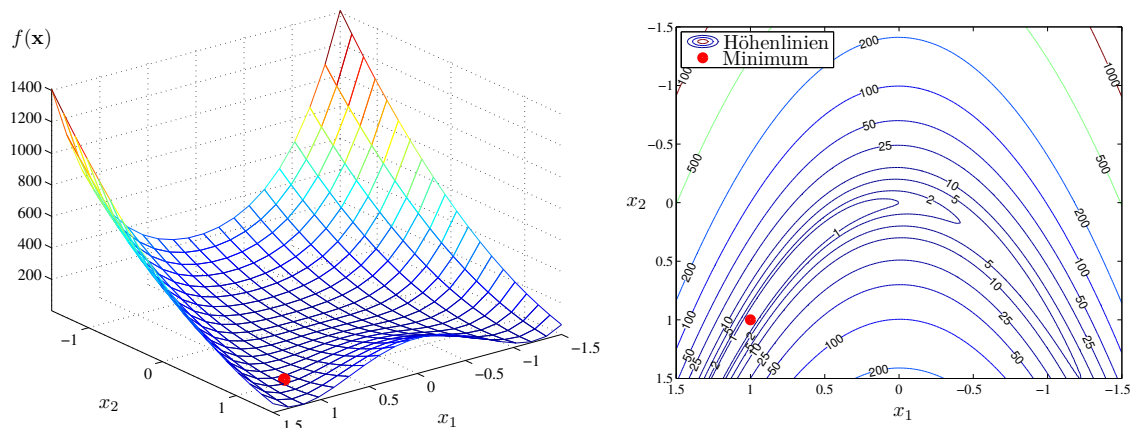


Abbildung 2.11: Profil und Höhenlinien von Rosenbrock's Bananenfunktion.

Aufgabe 2.17. Verifizieren Sie, dass der Punkt $\mathbf{x}^* = [1 \ 1]^T$ ein Minimum darstellt. Ist das Minimum $\mathbf{x}^*$ global und strikt? Existieren weitere lokale Minima? Sind die Funktion $f(\mathbf{x})$ und das Optimierungsproblem (2.134) konvex?

Zur Lösung von unbeschränkten Optimierungsproblemen stellt die *Optimization Toolbox* von MATLAB die folgenden Funktionen zur Verfügung

- **fminunc:** Liniensuche: Gradientenmethode, Quasi-Newton-Methode
Methode der Vertrauensbereiche: Newton-Methode
- **fminsearch:** Simplex-Verfahren nach Nelder-Mead

Eine empfehlenswerte Alternative ist die in [2.12] frei verfügbare MATLAB-Funktion

- **minFunc:** Liniensuche: Gradientenmethode, Konjugierte Gradientenmethode, Quasi-Newton-Methode, Newton-Methode

Tabelle 2.7 zeigt einige Vergleichsdaten für die numerische Lösung des Rosenbrock-Problems (ausgehend vom Startwert $\mathbf{x}_0 = [-1 \ -1]^T$), die mit Hilfe von **fminunc**, **fminsearch** und **minFunc** berechnet wurden. Abbildung 2.13 stellt zusätzlich die Iterationsverläufe für einige Verfahren dar. In der Code-Auflistung 2.1 am Ende dieses Abschnitts ist der MATLAB-Code für das Rosenbrock-Problem (2.134) angegeben, um zu verdeutlichen, wie die einzelnen Optimierungsverfahren mit den Befehlen **fminunc**, **fminsearch** und **minFunc** aufgerufen werden können.

Das *Gradientenverfahren* weist eine *langsame Konvergenz* auf. Selbst nach dem Erreichen der maximalen Anzahl an Funktionsauswertungen von 200 ist das Minimum immer noch

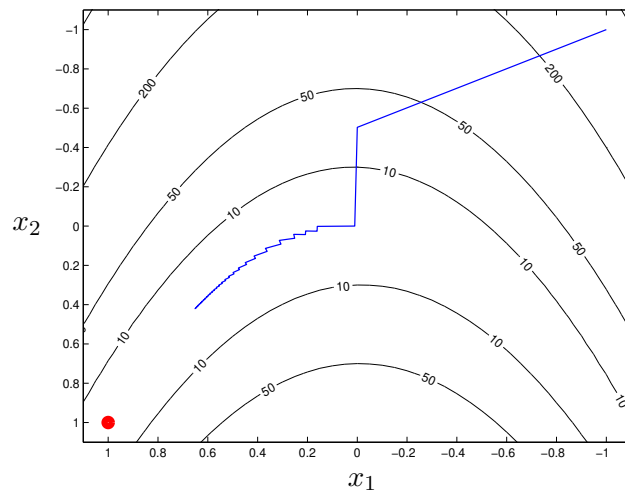


Abbildung 2.12: Iterationen des Gradientenverfahrens.

Verfahren	Funktion	Iter.	$f(\mathbf{x}^*)$	$\ (\nabla f)(\mathbf{x}^*)\ $	Funktionsaufrufe		
					$f(\mathbf{x})$	$(\nabla f)(\mathbf{x})$	$(\nabla^2 f)(\mathbf{x})$
LS: Gradientenmethode	fminunc	57	0.1232	1.1978	200	200	–
LS: Konj. Gradientenm.	minFunc	28	$7.4 \cdot 10^{-21}$	$7.7 \cdot 10^{-11}$	65	65	–
LS: Newton-Methode	minFunc	20	$3.8 \cdot 10^{-16}$	$7.3 \cdot 10^{-7}$	32	32	26
LS: Quasi-Newton (BFGS)	fminunc	23	$5.4 \cdot 10^{-12}$	$9.2 \cdot 10^{-6}$	29	29	–
LS: Gauss-Newton-Meth.	minFunc	4	$1.5 \cdot 10^{-29}$	$7.8 \cdot 10^{-15}$	9	9	–
VB: Newton-Methode	fminunc	25	$2.2 \cdot 10^{-18}$	$2.1 \cdot 10^{-8}$	26	26	26
Nelder-Mead Simplex-Verf.	fminsearch	67	$5.3 \cdot 10^{-10}$	–	125	–	–

Tabelle 2.7: Vergleich der numerischen Verfahren angewandt auf das Rosenbrock-Problem (LS=Liniensuche, VB=Methode der Vertrauensbereiche).

nicht gefunden. In Abbildung 2.12 ist der Iterationsverlauf des Gradientenverfahrens über den Höhenlinien der Rosenbrock-Funktion (2.134) dargestellt. Das Gradientenverfahren verwendet die Richtung des steilsten Abstieges, welche orthogonal zur jeweiligen Höhenlinie verläuft. In Richtung des Minimums werden die Iterationsschritte immer kleiner. Die niedrige Konvergenzgeschwindigkeit wurde bereits in Abbildung 2.6 veranschaulicht und soll in der folgenden Aufgabe näher untersucht werden.

Aufgabe 2.18. Berechnen Sie für das Minimum $\mathbf{x}^* = [1 \ 1]^T$ des Rosenbrock-Problems (2.134) die Konvergenzrate des Gradientenverfahrens gemäß Satz 2.8.

Das Konvergenzverhalten der *Quasi-Newton-Methode* in Abbildung 2.13 ist wesentlich

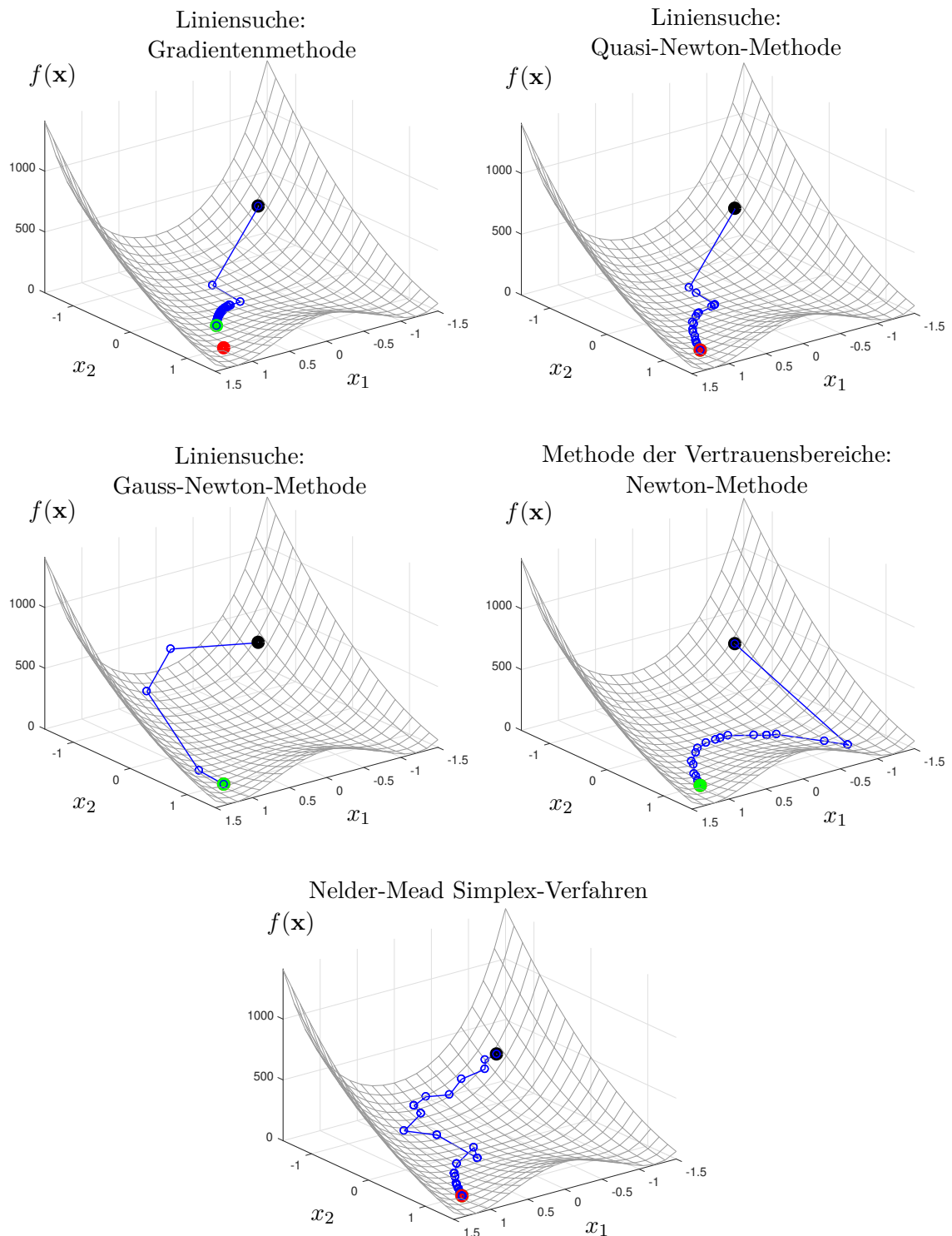


Abbildung 2.13: Rosenbrock's Bananenfunktion: Vergleich der numerischen Verfahren mit fminunc, fminsearch und minFunc.

besser als beim Gradientenverfahren. Die *Newton-Methode* (*Methode der Vertrauensbereiche*) in Abbildung 2.13 startet zunächst in die „falsche“ Richtung, was durch die *quadratische Approximation* (2.123) am Startpunkt $\mathbf{x}_0 = [-1 \ -1]^T$ zu erklären ist, deren Minimum in der Nähe von $\mathbf{x} \approx [-1 \ 1]^T$ liegt. Allerdings sind die einzelnen Schritte entlang des Tales der Rosenbrock-Funktion deutlich größer, da die Newton-Methode die *Hessematrix explizit verwendet* und nicht auf eine Approximation angewiesen ist wie im Fall der Quasi-Newton-Methode. Die *Gauss-Newton-Methode* weist für die Rosenbrock-Funktion ein sehr gutes Konvergenzverhalten auf. Trotz der Approximation der Hessematrix kann auch im Tal der Kostenfunktion mit großer Schrittweite das Minimum gefunden werden. Dieses besonders gute Verhalten ist aber auf die Form der Kostenfunktion zurückzuführen, welche dominante Terme enthält, die affin in $\mathbf{x}$ sind (vgl. dazu Aufgabe 2.15).

Zusätzlich sind in Tabelle 2.7 und Abbildung 2.13 die Ergebnisse für das *Simplex-Verfahren von Nelder-Mead*, welches in der MATLAB-Funktion `fminsearch` implementiert ist, angegeben. Allerdings bietet die Grafikausgabe unter `fminsearch` nicht die Möglichkeit, die einzelnen Simplexe darzustellen. In der nächsten Aufgabe soll das Simplex-Verfahren deshalb näher untersucht werden, um einen Eindruck von den Simplex-Operationen und der Robustheit des Verfahrens zu erhalten.

Aufgabe 2.19. Schreiben Sie eine MATLAB-Funktion, die das Rosenbrock-Problem (2.134) mit Hilfe des Simplex-Verfahrens nach Nelder-Mead numerisch löst (siehe den Algorithmus von Tabelle 2.6). Konstruieren Sie den ersten Simplex mit den Eckpunkten

$$\mathbf{x}_{0,1} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \quad \mathbf{x}_{0,2} = \mathbf{x}_{0,1} + s \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \mathbf{x}_{0,3} = \mathbf{x}_{0,1} + s \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (2.135)$$

in Abhängigkeit der Seitenlänge $s = 1$. Verwenden Sie für die Abbruchbedingung in Tabelle 2.6 die Schranke $\varepsilon_f = 10^{-9}$ und vergleichen Sie Ihre Ergebnisse mit jenen von `fminsearch` in Tabelle 2.7. Stellen Sie die Simplexe aus den einzelnen Iterationen grafisch dar. Untersuchen Sie die Robustheit und das Konvergenzverhalten des Simplex-Verfahrens für verschiedene Seitenlängen s des Startsimplex und unterschiedliche Startpunkte $\mathbf{x}_{0,1}$.

Aufgabe 2.20. Schreiben Sie eine MATLAB-Funktion, die das Rosenbrock-Problem (2.134) mit Hilfe der Newton-Methode (Liniensuche) numerisch löst, siehe Abschnitt 2.3.2.3. Verwenden Sie die heuristischen Bedingungen von Abschnitt 2.3.1.3 für die Schrittweitenbestimmung von α_k . Vergleichen Sie das beobachtete Verhalten mit den Werten in Tabelle 2.7. Stellen Sie die einzelnen Iterationen grafisch dar und variieren Sie die Startpunkte $\mathbf{x}_0$.

Code-Auflistung 2.1: MATLAB-Code für das Rosenbrock-Problem (`fminunc`, `fminsearch`, `minFunc`).

```
function Xopt = rosenbrock_problem(Xinit,methodQ)
% Xinit: Startpunkt
% methodQ: 1 - fminunc: Liniensuche: Gradientenmethode
%           2 - fminunc: Liniensuche: Quasi-Newton
```

```

%      3 - fminunc: Methode der Vertrauensbereiche: Newton-Methode
%      4 - fminsearch: Nelder-Mead Simplex-Verfahren
%      5 - minFunc: CG-Methode
%      6 - minFunc: Newton-Methode
%      7 - minFunc: Gauss-Newton-Methode
global old
old = [Xinit; rosenbrock(Xinit)];

% Optionen für fminunc:
opt_fminu = optimoptions('fminunc','Display','iter','PlotFcn',@plot_iterates);
% Optionen für fminsearch:
opt_fmins = optimset('Display','iter','PlotFcn',@plot_iterates);
% Optionen für minFunc:
opt=struct('display','iter','outputFcn',@(x,state,varargin) plot_iterates(x,[],state));

% Generelles: 'SpecifyObjectiveGradient' gibt an, dass die Funktion zur Berechnung der
% Kostenfunktion auch den Wert des Gradienten retourniert. Ist auch 'HessianFcn' auf
% 'objective' gesetzt, muss zusätzlich die Hessematrix zurückgegeben werden.
switch methodQ
case 1 % fminunc: Liniensuche: Gradientenmethode
    opt_fminu = optimoptions(opt_fminu,'Algorithm','quasi-newton','HessUpdate','steepdesc',...
        'SpecifyObjectiveGradient',true);
    [Xopt,fopt,exitflag,output] = fminunc(@rosenbrock,Xinit,opt_fminu);
case 2 % fminunc: Liniensuche: Quasi-Newton
    opt_fminu = optimoptions(opt_fminu,'Algorithm','quasi-newton','HessUpdate','bfgs',...
        'SpecifyObjectiveGradient',true);
    [Xopt,fopt,exitflag,output] = fminunc(@rosenbrock,Xinit,opt_fminu);
case 3 % fminunc: Methode der Vertrauensbereiche: Newton-Methode
    opt_fminu = optimoptions(opt_fminu,'Algorithm','trust-region','HessianFcn','objective',...
        'SpecifyObjectiveGradient',true);
    [Xopt,fopt,exitflag,output] = fminunc(@rosenbrock,Xinit,opt_fminu);
case 4 % fminsearch: Nelder-Mead Simplex-Verfahren
    [Xopt,fopt,exitflag,output] = fminsearch(@rosenbrock,Xinit,opt_fmins);
case 5 % minFunc: CG-Methode
    figure
    opt.Method = 'cg';
    [Xopt,fopt,exitflag,output] = minFunc(@rosenbrock,Xinit,opt);
case 6 % minFunc: Newton-Methode
    figure
    opt.Method = 'newton';
    [Xopt,fopt,exitflag,output] = minFunc(@rosenbrock,Xinit,opt);
case 7 % minFunc: Gauss-Newton-Methode
    figure
    opt.Method = 'newton';
    [Xopt,fopt,exitflag,output] = minFunc(@rosenbrock_gauss_newton,Xinit,opt);
end
end

function [f, grad, H] = rosenbrock(x)
    grad = [];
    H = [];
    f = 100*(x(2)-x(1)^2)^2 + (x(1)-1)^2; % Rosenbrock-Funktion
    if nargin>1 % falls Gradient angefordert wird
        grad = [ -400*(x(2)-x(1)^2)*x(1)+2*(x(1)-1);
            200*(x(2)-x(1)^2) ];
    end
    if nargin>2 % falls Hessematrix angefordert wird
        H = [ -400*(x(2)-3*x(1)^2)+2, -400*x(1);
            -400*x(1), 200 ];
    end
end

function [f, grad, H]=rosenbrock_gauss_newton(x)

```

```

% Rosenbrock-Funktion wird umformuliert als Norm einer
% vektorwertigen Funktion r(x):
% f(x)=100*(x2-x1^2)^2+(x1-1)^2 = 1/2*norm(r(x))^2
% mit
% r1(x)=sqrt(200)*(x2-x1^2)
% r2(x)=sqrt(2)*(x1-1)
r=[sqrt(200)*(x(2)-x(1)^2);
  sqrt(2)*(x(1)-1)];
f=1/2*norm(r)^2;
if nargin>1
    grad_r=[-sqrt(200)*2*x(1),sqrt(2);
            sqrt(200),0];
    grad=grad_r*r;
end
if nargin>2
    H=grad_r*grad_r';
end
end

function stop = plot_iterates(x,info,state)
    global old
    f = rosenbrock(x);
    switch state
        case 'init'
            plot_surface(x,f);
        case 'iter'
            plot3([old(1),x(1)], [old(2),x(2)], [old(3),f], 'b-o', 'LineWidth',1);
        case 'done'
            plot3(x(1),x(2),f, 'go', 'LineWidth',5);
    end
    stop = false;
    old = [x;f];
end

function plot_surface(x,f)
    [X1,X2] = meshgrid(-1.5:0.15:1.5);
    F = 100*(X2-X1.^2).^2 + (X1-1).^2;
    h = surf(X1,X2,F, 'EdgeColor',0.6*[1,1,1], 'FaceColor','none');
    hold on; axis tight;
    plot3(x(1),x(2),f, 'ko', 'LineWidth',5);
    plot3(1,1,0, 'ro', 'LineWidth',5);
    xlabel('x_1'); ylabel('x_2'); zlabel('f')
    set(gcf, 'ToolBar','figure');
    set(gca, 'Xdir','reverse', 'Ydir','reverse');
    set(gca, 'clipping','off');
end

```

% Grafische Ausgabe:
% Initialisierung
% Iterationen
% nach letzter Iteration
% kein Abbruchkriterium
% 3D-Profil von
% Rosenbrock-Funktion
% Startpunkt
% optimale Lösung
% Aktivieren der Menüleiste (Zoom, etc.)

2.7 Literatur

- [2.1] D. P. Bertsekas, *Nonlinear Programming*, 2. Aufl. Athena Scientific, 1999.
- [2.2] D. G. Luenberger und Y. Ye, *Linear and Nonlinear Programming* (International Series in Operations Research & Management Science), 3. Aufl. Springer, 2008, Bd. 116.
- [2.3] P. E. Gill, W. Murray und M. H. Wright, *Practical Optimization*. Academic Press, 1981.
- [2.4] R. Fletcher, *Practical methods of optimization*, 2. Aufl. John Wiley & Sons, 1987.
- [2.5] I. Griva, S. Nash und A. Sofer, *Linear and Nonlinear Optimization*, 2. Aufl. Society for Industrial und Applied Mathematics, 2009.
- [2.6] E. Chong und S. Żak, *An Introduction to Optimization* (Wiley Series in Discrete Mathematics and Optimization), 4. Aufl. Wiley, 2013.
- [2.7] M. Hestenes und E. Stiefel, „Methods of conjugate gradients for solving linear systems,“ *Journal of Research of the National Bureau of Standards*, Jg. 49, Nr. 6, S. 409–436, 1952.
- [2.8] J. Shewchuk, *An introduction to the conjugate gradient method without the agonizing pain*, Technical Report CMU-CS-94-125, Pittsburgh, PA: School of Computer Science, Carnegie Mellon University, 1994.
- [2.9] J. Nocedal und S. J. Wright, *Numerical Optimization* (Springer Series in Operations Research and Financial Engineering), 2. Aufl. Springer, 2006.
- [2.10] J. Dennis und R. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations* (Classics in Applied Mathematics). Society for Industrial und Applied Mathematics, 1996.
- [2.11] M. Papageorgiou, M. Leibold und M. Buss, *Optimierung: Statische, dynamische, stochastische Verfahren für die Anwendung*, 4. Aufl. Springer, 2015.
- [2.12] M. Schmidt, „minFunc,“ abrufbar unter <http://www.cs.ubc.ca/~schmidt/Software/minFunc.html>, University of British Columbia, Vancouver, 2005. besucht am 28. Sep. 2016.
- [2.13] S. Boyd und L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.
- [2.14] C. T. Kelley, *Iterative Methods for Optimization*. Society for Industrial und Applied Mathematics, 1999.
- [2.15] B. C. Chachuat, „Nonlinear and Dynamic Optimization: From Theory to Practice,“ abrufbar unter <http://infoscience.epfl.ch/record/111939>, Laboratoire d'Automatique, École Polytechnique Fédérale de Lausanne, 2007. besucht am 30. Sep. 2020.
- [2.16] H. T. Jongen, K. Meer und E. Triesch, *Optimization Theory*. Kluwer Academic Publishers, 2004.