MATTHIAS GERDTS

Optimierung

Universität der Bundeswehr München
Wintertrimester 2014

Addresse des Autors:

Matthias Gerdts

Institut für Mathematik und Rechneranwendung
Universität der Bundeswehr München
Werner-Heisenberg-Weg 39
85577 Neubiberg

E-Mail: matthias.gerdts@unibw.de
WWW: www.unibw.de/lrt1/gerdts

Vorläufige Version: 25. April 2017

Inhaltsverzeichnis

Vorwort

Diese Vorlesung basiert auf Vorlesungen, die von Prof. Dr. Oberle, Prof. Dr. Lempio und mir an den Universitäten Bayreuth und Hamburg gehalten wurden.
Sie vermittelt weiterführende Themen aus der nichtlinearen Optimierung und richtet sich an Studierende im Masterstudiengang Mathematical Engineering.
Die Vorlesung baut auf dem Bachelorstudium auf und hat zum Ziel, weiterführende Konzepte und Methoden der nichtlinearen Optimierung zu vermitteln.

Allgemeine Tipps und Hinweise:

- Mathematische Inhalte erschließen sich nicht von selbst und in den seltensten Fällen durch gelegentliches Überfliegen des Vorlesungsskriptes, sondern sie erfordern ein **aktives Mitarbeiten** und **viel Übung**.

  Arbeiten Sie deshalb die Vorlesungen (und Übungen) nach!

  Auch wenn die wöchentlichen Übungsaufgaben aus Kapazitätsgründen nicht eingesammelt und korrigiert werden können, so ist es zum Verständnis des Stoffes **sehr wichtig**, die Übungsaufgaben **regelmäßig** und **selbstständig** zu bearbeiten.

- Gehen Sie den Vorlesungsstoff und Übungsaufgaben mit Kommilitonen durch. Erklären und diskutieren Sie Definitionen, Sätze und Verfahren in eigenen Worten! Fragen Sie auch die Übungsleiter und den Leiter der Veranstaltung bei Unklarheiten.

- Es ist keine gute Idee, sich erst kurz vor der Prüfung mit dem Vorlesungsstoff zu beschäftigen. Dann ist es zu spät und es bleibt nichts hängen.

- Wichtige Informationen zur Vorlesung, wie z.B. Ansprechpartner, Übungsaufgaben und Zusatzmaterial, finden sich auf der WWW-Seite

  http://www.unibw.de/lrt1/gerdts/lehre/optimierung-msc

  Diese Seite wird laufend aktualisiert, so dass sie regelmäßig besucht werden sollte.

Literatur:

- M. Gerdts, F. Lempio: Mathematische Optimierungsverfahren des Operations Research, DeGruyter, 2011.

- C. Geiger, C. Kanzow: Numerische Verfahren zur Lösung unrestringierter Optimierungsaufgaben, Springer, Berlin-Heidelberg-New York, 1999.
- C. Geiger, C. Kanzow: Theorie und Numerik restringierter Optimierungsaufgaben, Springer, Berlin-Heidelberg-New York, 2002.
- F. Jarre, J. Stoer: Optimierung, Springer, Berlin-Heidelberg-New York, 2004.
- J. Nocedal, S. J. Wright: Numerical optimization, Springer Series in Operations Research, New York, 1999.
- R. Fletcher: Practical Methods of Optimization, John Wiley & Sons, 2. Ausgabe, Chichester–New York–Brisbane–Toronto–Singapore, 2003.

Kapitel 1

Einleitung, Wiederholung und Grundlagen

Optimierungsaufgaben treten in den Wirtschaftswissenschaften (Operations Research), in der Technik und in den Naturwissenschaften in vielfältiger Art und Weise auf. In der einführenden Optimierungsvorlesung haben wir uns hauptsächlich mit unrestringierten Optimierungsproblemen, linearen und konvexen Optimierungsproblemen sowie notwendigen und hinreichenden Bedingungen für restringierte nichtlineare Optimierungsprobleme beschäftigt. Gegenstand dieser Vorlesung sind restringierte Optimierungsprobleme und Lösungsverfahren für verschiedene Aufgabenstellungen.

1.1 Aufgabenstellung

Wir fassen die wesentlichen Resultate aus der Einführung in die Optimierung aus dem Bachelorstudium zusammen. Diese Resultate sind das Rüstzeug für diese Vorlesung und werden uns in den folgenden Kapiteln immer wieder begegnen.
Für gegebene (und hinreichend oft differenzierbare) Funktionen

$$\begin{aligned} f &: \mathbb{R}^n \longrightarrow \mathbb{R}, \\ g = (g_1, \ldots, g_m)^\top &: \mathbb{R}^n \longrightarrow \mathbb{R}^m, \\ h = (h_1, \ldots, h_p)^\top &: \mathbb{R}^n \longrightarrow \mathbb{R}^p \end{aligned}$$

betrachten wir das restringierte Standard-Optimierungsproblem:

Problem 1.1.1 **(Standard-Optimierungsproblem)**
Finde $x \in \mathbb{R}^n$, so dass $f(x)$ minimal wird unter den Nebenbedingungen

$$\begin{aligned} g_i(x) &\leq 0, \quad i = 1, \ldots, m, \\ h_j(x) &= 0, \quad j = 1, \ldots, p. \end{aligned}$$

In Kurzform schreiben wir auch:

$$\textit{Minimiere} \qquad f(x) \qquad \textit{u.d.N.} \qquad g(x) \leq 0,\ h(x) = 0.$$

Mit

$$\Sigma := \{x \in \mathbb{R}^n \mid g_i(x) \le 0,\ i = 1, \dots, m,\ h_j(x) = 0,\ j = 1, \dots, p\} \tag{1.1}$$

bezeichnen wir die **zulässige Menge** (oder den **zulässigen Bereich**) des Standard-Optimierungsproblems. Die Menge Σ in (1.1) ist eine abgeschlossene Menge, falls die Funktionen g_i und h_j stetig sind.

Ein Punkt $x \in \mathbb{R}^n$ heißt

- **zulässig**, falls $x \in \Sigma$ gilt, und
- **unzulässig**, falls $x \notin \Sigma$ gilt.

Für zulässige Punkte $x \in \Sigma$ bezeichnen wir mit

$$\mathcal{A}(x) := \{i \in \{1, \dots, m\} \mid g_i(x) = 0\}$$

die **Indexmenge der (in x) aktiven Ungleichungsrestriktionen** und nennen die Restriktion $g_i(x) \le 0$

- **aktiv in** x, wenn $g_i(x) = 0$ gilt, und
- **inaktiv in** x, wenn $g_i(x) < 0$ gilt.

Das Standard-Optimierungsproblem enthält die folgenden Problemklassen als Spezialfälle:

- **Unrestringiertes Optimierungsproblem:**

  $$\text{Minimiere} \qquad f(x) \qquad \text{u.d.N.} \qquad x \in \mathbb{R}^n.$$

  Hierin treten keine Ungleichungs- und Gleichungsrestriktionen der Form $g(x) \le 0$ oder $h(x) = 0$ auf (die Dimensionen von g und h sind Null, d.h. $m = 0$, $p = 0$).

- **Lineares Optimierungsproblem (in primaler Normalform):**

  $$\text{Minimiere} \qquad c^\top x \qquad \text{u.d.N.} \qquad Ax = b,\ x \ge 0.$$

  Spezialfall des Standard-Optimierungsproblems mit $f(x) = c^\top x$, $g(x) = -x$, $h(x) = Ax - b$ (oder $h(x) = b - Ax$), wobei $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ Vektoren und $A \in \mathbb{R}^{m \times n}$ eine Matrix sind. Das lineare Optimierungsproblem ist ein konvexes Optimierungsproblem.

- **Linear-quadratisches Optimierungsproblem:**

  $$\text{Minimiere} \qquad \frac{1}{2} x^\top Q x + c^\top x \qquad \text{u.d.N.} \qquad Ax = b,\ x \ge 0.$$

  Spezialfall des Standard-Optimierungsproblems mit $f(x) = \frac{1}{2} x^\top Q x + c^\top x$, $g(x) = -x$, $h(x) = Ax - b$ (oder $h(x) = b - Ax$), wobei $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ Vektoren und $Q \in \mathbb{R}^{n \times n}$ und $A \in \mathbb{R}^{m \times n}$ Matrizen sind. Falls Q symmetrisch und positiv semidefinit ist, liegt ein konvexes Optimierungsproblem vor.

- **Konvexes Optimierungsproblem:**

  $$\text{Minimiere} \qquad f(x) \qquad \text{u.d.N.} \qquad g(x) \le 0,\ Ax = b.$$

  Spezialfall des Standard-Optimierungsproblems, wobei f und g konvexe Funktionen, $A \in \mathbb{R}^{m\times n}$ eine Matrix und $b \in \mathbb{R}^m$ ein Vektor sind.

1.2 Notwendige Bedingungen für restringierte Optimierungsprobleme

Die folgenden Karush-Kuhn-Tucker (KKT) Bedingungen sind die Basis für viele theoretische Untersuchungen und numerische Algorithmen.
Die **Lagrange-Funktion** für das Standard-Optimierungsproblem ist definiert als

$$\begin{aligned} L(x,\lambda,\mu) &= f(x) + \lambda^\top g(x) + \mu^\top h(x) \\ &= f(x) + \sum_{i=1}^m \lambda_i g_i(x) + \sum_{j=1}^p \mu_j h_j(x), \end{aligned}$$

wobei $\lambda = (\lambda_1,\dots,\lambda_m)^\top \in \mathbb{R}^m$ und $\mu = (\mu_1,\dots,\mu_p)^\top \in \mathbb{R}^p$ als **Lagrange-Multiplikatoren** bezeichnet werden. Es gilt:

Satz 1.2.1 **(Notwendige Bedingungen erster Ordnung, KKT-Bedingungen)**
Voraussetzungen:

- *$\hat{x}$ ist ein lokales Minimum des Standard-Optimierungsproblems.*
- *Die Funktionen f, g_i, $i = 1,\dots,m$, und h_j, $j = 1,\dots,p$, sind stetig differenzierbar.*
- *Es gilt die Linear Independence Constraint Qualification (LICQ) in $\hat{x}$, d.h. die Vektoren*

  $$\nabla g_i(\hat{x}),\ i \in \mathcal{A}(\hat{x}) \qquad \textit{und} \qquad \nabla h_j(\hat{x}),\ j = 1,\dots,p,$$

  sind linear unabhängig.

Dann existieren eindeutig bestimmte Multiplikatoren $\lambda = (\lambda_1,\dots,\lambda_m)^\top \in \mathbb{R}^m$ und $\mu = (\mu_1,\dots,\mu_p)^\top \in \mathbb{R}^p$, so dass die folgenden Bedingungen gelten:

(a) **Stationarität der Lagrange-Funktion:**

$$\nabla_x L(\hat{x},\lambda,\mu) = 0 \tag{1.2}$$

bzw.

$$\nabla f(\hat{x}) + \sum_{i=1}^m \lambda_i \nabla g_i(\hat{x}) + \sum_{j=1}^p \mu_j \nabla h_j(\hat{x}) = 0. \tag{1.3}$$

(c) **Komplementaritätsbedingungen:** *Für $i = 1, \dots, m$ gilt:*

$$\lambda_i \geq 0 \qquad \textit{und} \qquad \lambda_i g_i(\hat{x}) = 0. \tag{1.4}$$

(d) **Zulässigkeit:**

$$g(\hat{x}) \leq 0 \qquad \textit{und} \qquad h(\hat{x}) = 0. \tag{1.5}$$

Jeden Punkt $(\hat{x}, \lambda, \mu)$, der die Bedingungen (8.5)-(1.5) erfüllt, nennen wir **KKT-Punkt** oder **stationären Punkt**. Beachte, dass KKT-Punkte lediglich Kandidaten für optimale Lösungen liefern.

Bemerkung 1.2.2

Die LICQ ist nur eine von vielen sogenannten Constraint Qualifications (oder Regularitätsbedingungen) an den zulässigen Bereich. Die üblichen alternativen Constraint Qualifications (z.B. von Mangasarian-Fromowitz, Robinson, Abadie, Guignard) garantieren zwar auch die Gültigkeit der KKT-Bedingungen, allerdings liefern sie allesamt nicht die Eindeutigkeit der Multiplikatoren λ und μ. Zu beachten ist jedoch, dass die KKT-Bedingungen i.A. nicht mehr gelten, wenn überhaupt keine Constraint Qualification erfüllt ist (in diesem Fall gelten nur noch die sogenannten Fritz John Bedingungen). ■

Aus den KKT-Bedingungen ergeben sich folgende Spezialfälle:

- Für das **unrestringierte Optimierungsproblem**

  $$\text{Minimiere} \qquad f(x) \qquad \text{u.d.N.} \qquad x \in \mathbb{R}^n$$

  erhält man die notwendige Bedingung

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

- Für das **lineare Optimierungsproblem (in primaler Normalform)**

  $$\text{Minimiere} \qquad c^\top x \qquad \text{u.d.N.} \qquad Ax = b,\ x \geq 0$$

  erhält man mit der Lagrange-Funktion $L(x, \lambda, \mu) = c^\top x + \lambda^\top(-x) + \mu^\top(b - Ax)$ die notwendigen Bedingungen

  $$c - \lambda - A^\top \mu = 0, \qquad \lambda \geq 0, \qquad \lambda^\top(-\hat{x}) = 0$$

  bzw.

  $$A^\top \mu \leq c, \qquad \hat{x}^\top(c - A^\top \mu) = 0.$$

- Für das **linear-quadratische Optimierungsproblem**

$$\text{Minimiere} \qquad \frac{1}{2}x^\top Qx + c^\top x \qquad \text{u.d.N.} \qquad Ax = b,\ x \geq 0$$

erhält man mit der Lagrange-Funktion

$$L(x, \lambda, \mu) = \frac{1}{2}x^\top Qx + c^\top x + \lambda^\top(-x) + \mu^\top(b - Ax)$$

die notwendigen Bedingungen

$$Q\hat{x} + c - \lambda - A^\top\mu = 0, \qquad \lambda \geq 0, \qquad \lambda^\top(-\hat{x}) = 0.$$

- Für das **gleichungsrestringierte Optimierungsproblem**

$$\text{Minimiere} \qquad f(x) \qquad \text{u.d.N.} \qquad h(x) = 0$$

erhält man mit der Lagrange-Funktion $L(x, \mu) = f(x) + \mu^\top h(x)$ die notwendigen Bedingungen

$$\begin{pmatrix} \nabla_x L(\hat{x}, \mu) \\ h(\hat{x}) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Dies ist i.A. ein nichtlineares Gleichungssystem in den Unbekannten x und μ und kann mit dem Newtonverfahren gelöst werden ($\longrightarrow$ Lagrange-Newton-Verfahren).

Für eine notwendige Bedingung zweiter Ordnung in einem KKT-Punkt $(\hat{x}, \hat{\lambda}, \hat{\mu})$ benötigen wir den **kritischen Kegel**

$$\begin{aligned} T_K(x) \ := \ \{d \in \mathbb{R}^n \mid\ & \nabla g_i(x)^\top d \leq 0,\ i \in \mathcal{A}(x), \hat{\lambda}_i = 0, \\ & \nabla g_i(x)^\top d = 0,\ i \in \mathcal{A}(x), \hat{\lambda}_i > 0, \\ & \nabla h_j(x)^\top d = 0,\ j = 1, \ldots, p\}. \end{aligned}$$

Der kritische Kegel enthält tangentiale Richtungen d an den zulässigen Bereich für die die Richtungsableitung $\nabla f(\hat{x})^\top d$ gleich Null ist. Für diese kritischen Richtungen müssen Bedingungen zweiter Ordnung herangezogen werden.

Es gilt:

Satz 1.2.3 (Notwendige Bedingungen zweiter Ordnung)
Voraussetzungen:

- *f, g_i, $i = 1, \ldots, m$, und h_j, $j = 1, \ldots, p$ sind zweimal stetig differenzierbar.*
- *$(\hat{x}, \lambda, \mu)$ ist ein KKT-Punkt.*

- *$\hat{x}$ ist ein lokales Minimum des Standard-Optimierungsproblems.*
- *Es gilt die LICQ in $\hat{x}$.*

Dann gilt

$$d^\top \nabla^2_{xx} L(\hat{x}, \lambda, \mu) d \geq 0 \qquad \forall d \in T_K(\hat{x})$$

(Die Hessematrix der Lagrange-Funktion ist positiv semidefinit auf dem kritischen Kegel).

Beispiel 1.2.4

Für das **gleichungsrestringierte Optimierungsproblem**

$$\textit{Minimiere} \qquad f(x) \qquad \textit{u.d.N.} \qquad Ax - b = 0$$

lautet der kritische Kegel

$$T_K(x) = \{d \in \mathbb{R}^n \mid Ad = 0\}.$$

Für die notwendigen Bedingungen zweiter Ordnung muss man also überprüfen, ob die Hessematrix der Lagrange-Funktion positiv semidefinit auf dem Kern von A ist. ■

Treten keine Restriktionen $g(x) \leq 0$ und $h(x) = 0$ auf (unbeschränkter Fall), so ist der kritische Kegel durch $T_K(\hat{x}) = \mathbb{R}^n$ gegeben und die notwendige Bedingung zweiter Ordnung reduziert sich auf die Bedingung

$$H_f(\hat{x}) \text{ ist positiv semidefinit,}$$

wobei $H_f(\hat{x})$ die Hessematrix der Zielfunktion f in $\hat{x}$ bezeichnet.

1.3 Hinreichende Bedingungen für restringierte Optimierungsprobleme

Die hinreichenden Bedingungen zweiter Ordnung sind sehr nah an den notwendigen Bedingungen zweiter Ordnung.

Satz 1.3.1 (Hinreichende Bedingung zweiter Ordnung)

Voraussetzungen:

- *f, g_i, $i = 1, \dots, m$ und h_j, $j = 1, \dots, p$ sind zweimal stetig differenzierbar.*
- *$(\hat{x}, \lambda, \mu)$ ist KKT-Punkt des Standard-Optimierungsproblems.*

- *Es gilt*
$$d^\top \nabla^2_{xx} L(\hat{x}, \lambda, \mu) d > 0 \qquad \forall d \in T_K(\hat{x}),\ d \neq 0 \tag{1.6}$$
(Die Hessematrix der Lagrange-Funktion ist positiv definit auf dem kritischen Kegel).

Dann existiert eine Umgebung U von $\hat{x}$ und ein $\alpha > 0$ mit

$$f(x) \geq f(\hat{x}) + \alpha \|x - \hat{x}\|^2 \qquad \forall x \in \Sigma \cap U$$

(insbesondere ist $\hat{x}$ also lokales Minimum und f wächst lokal mindestens quadratisch).

Beispiel 1.3.2

Für das **gleichungsrestringierte Optimierungsproblem**

$$\textit{Minimiere} \qquad f(x) \qquad \textit{u.d.N.} \qquad Ax - b = 0$$

lautet der kritische Kegel

$$T_K(x) = \{d \in \mathbb{R}^n \mid Ad = 0\}.$$

Für die hinreichende Bedingung zweiter Ordnung muss man also überprüfen, ob die Hessematrix der Lagrange-Funktion positiv definit auf dem Kern von A ist. ■

Treten keine Restriktionen $g(x) \leq 0$ und $h(x) = 0$ auf (unbeschränkter Fall), so ist der kritische Kegel durch $T_K(\hat{x}) = \mathbb{R}^n$ gegeben und die hinreichende Bedingung zweiter Ordnung reduziert sich auf die Bedingung (1.6)

$$H_f(\hat{x}) \text{ ist positiv definit,}$$

wobei $\hat{x}$ ein stationärer Punkt von f sei.

1.4 Verfahren für unrestringierte Optimierungsprobleme

Für die Minimierung der Funktion $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ haben wir iterative Verfahren kennengelernt, die ausgehend von einer Startschätzung $x^{[0]}$ Näherungslösungen

$$x^{[k+1]} := x^{[k]} + \alpha_k d^{[k]} \qquad \text{für } k = 0, 1, 2, \ldots$$

berechnen. Hierin ist $d^{[k]}$ eine Suchrichtung und $\alpha_k > 0$ eine Schrittweite.
Zur Bestimmung der Suchrichtung gibt es u.a. die folgenden Ansätze:

- **Gradientenverfahren:** Beim Gradientenverfahren wird stets die Richtung des steilsten Abstiegs gewählt, also
$$d^{[k]} := -\nabla f(x^{[k]}).$$

Diese Richtung ist außer in einem stationären Punkt stets eine Abstiegsrichtung.

- **Newtonverfahren:** Beim Newtonverfahren wird die Richtung

$$d^{[k]} := -H_f(x^{[k]})^{-1} \nabla f(x^{[k]})$$

gewählt. Ist die Hessematrix $H_f(x^{[k]})$ positiv definit, so ist die Newton-Richtung außer in einem stationären Punkt eine Abstiegsrichtung.

- **Quasi-Newtonverfahren:** Beim Quasi-Newtonverfahren wird die Richtung

$$d^{[k]} := -H_k^{-1} \nabla f(x^{[k]})$$

gewählt, wobei die Matrix H_k stets symmetrisch und positiv definit gewählt wird und durch eine sogenannte Update-Formel in jedem Iterationsschritt aufdatiert wird. Diese Quasi-Newton-Richtung ist außer in einem stationären Punkt stets eine Abstiegsrichtung.

- **Trust-Region-Verfahren:** Die Suchrichtung ist durch Lösen des Trust-Region-Hilfsproblems

$$\text{Minimiere} \qquad \frac{1}{2} d^\top H_k d + \nabla f(x^{[i]})^\top d \qquad \text{u.d.N.} \qquad \|d\| \leq \Delta_k$$

gegeben, wobei der Trust-Region-Radius $\Delta_k > 0$ und die Matrix H_k in jedem Schritt angepasst werden. Als Schrittweite wird beim Trust-Region-Verfahren stets $\alpha_k = 1$ gewählt, da die Schrittlänge durch den Trust-Region-Radius Δ_k gesteuert wird.

Beim Trust-Region-Verfahren arbeitet man mit der Schrittweite $\alpha_k = 1$ und verzichtet auf die nachfolgende Liniensuche, da sie durch die Steuerung des Trust-Region-Radius Δ_k überflüssig ist. Für alle anderen Verfahren kann die Schrittweite $\alpha_k > 0$ mithilfe einer eindimensionalen Liniensuche für die Funktion

$$\varphi(\alpha) := f(x^{[k]} + \alpha d^{[k]})$$

berechnet werden. Voraussetzung ist aber, dass $d^{[k]}$ eine Abstiegsrichtung ist. Üblicherweise verwendet man dann das Armijo-Verfahren oder darauf aufbauende Verfahren, z.B. die Wolfe-Powell-Regeln.

Algorithmus 1.4.1 **(Armijo-Regel)**

(i) Wähle $\beta \in (0,1)$, $\sigma \in (0,1)$ *und setze* $\alpha := 1$.

(ii) Falls die Bedingung

$$\varphi(\alpha) \leq \varphi(0) + \sigma \cdot \alpha \cdot \varphi'(0)$$

erfüllt ist, setze $\alpha_k := \alpha$ und beende das Verfahren. Andernfalls gehe zu (iii).

(iii) Setze $\alpha := \beta \cdot \alpha$ und gehe zu (ii).

1.5 Überblick

Zur numerischen Lösung des Standard-Optimierungsproblems gibt es im Wesentlichen die folgenden Herangehensweisen:

(a) **Penalty- und Multiplikator-Verfahren:** Diese Verfahren basieren auf der Ankopplung der Nebenbedingungen an die Zielfunktion mithilfe eines gewichteten Strafterms (Penalty-Term), der unzulässige Punkte bestraft. Dadurch werden die Nebenbedingungen eliminiert und man kann Verfahren der unrestringierten Optimierung (Gradientenverfahren, Newtonverfahren, Quasi-Newton-Verfahren) anwenden. Zielfunktion und Strafterm müssen jedoch geeignet gewichtet werden, damit man letztendlich eine zulässige Lösung bekommt.

(b) **Sequentielle quadratische Programmierung (SQP)**: SQP-Verfahren basieren auf der lokalen Approximation des Standard-Optimierungsproblems durch ein linear-quadratisches Optimierungsproblem, dessen Lösung die Suchrichtung in einem iterativen Verfahren liefert. SQP-Verfahren erweitern das Lagrange-Newton-Verfahren auf Probleme mit Ungleichungsrestriktionen.

(c) **Innere-Punkte-Verfahren (IP):** Innere-Punkte-Verfahren verwenden sogenannte Barriere-Funktionen, um Ungleichungsnebenbedingungen zu eliminieren und diese ähnlich wie bei Penalty-Verfahren mithilfe eines gewichteten Stratfterms an die Zielfunktion zu koppeln. Im Gegensatz zu Penalty- und Multiplikator-Verfahren wird dabei nicht nur das Verlassen des zulässigen Bereichs bestraft, sondern es wird bereits die Annäherung an den Rand des zulässigen Bereichs bestraft (den Rand des zulässigen Bereichs kann man sich als unüberwindliche Barriere vorstellen).

(d) **Verfahren für Komplementaritätsprobleme:** Diese Verfahren, zu denen semiglatte Newtonverfahren oder Variationsmethoden gehören, versuchen, die KKT-Bedingungen direkt zu lösen. Dazu werden die Komplementaritätsprobleme entweder als Variationsungleichung umgeschrieben und mit geeigneten Verfahren gelöst,

oder sie werden mithilfe von speziellen Funktionen als Gleichung reformuliert. Letzteres führt auf ein nichtdifferenzierbares Gleichungssystem, auf das Varianten des Newtonverfahrens angewendet werden können.

Jede dieser Verfahrensklassen verwendet zusätzlich Strategien, um Konvergenz von beliebigen Startschätzungen zu erreichen (Globalisierungsstrategien). Die üblichen Strategien sind

- eindimensionale Liniensuche (z.B. Armijo-Verfahren)
- Trust-Region-Verfahren (Approximation auf einem Vertrauensbereich)
- Filterverfahren (versuchen, nicht-dominierte Iterierte zu erzeugen, wobei Zielfunktionswert und Verletzung der Restriktionen als zwei Kriterien mitgeführt werden)

Kapitel 2

Penalty- und Multiplikator-Verfahren

Penalty- und Multiplikatorverfahren sind beliebte Verfahren, die auf der Ankopplung der Nebenbedingungen an die Zielfunktion mithilfe eines gewichteten Strafterms (Penalty-Term) basieren. Der Strafterm bestraft unzulässige Punkte. Der Vorteil der Verfahren ist, dass durch die Ankopplung der Nebenbedingungen an die Zielfunktion Nebenbedingungen eliminiert werden, so dass Verfahren der unrestringierten Optimierung angewendet werden können.
Das Konzept der Penalty-Verfahren für die allgemeine Aufgabenstellung

$$(P) \qquad \text{Minimiere} \qquad f(x) \qquad \text{u.d.N.} \qquad x \in \Sigma$$

funktioniert wie folgt. Man benötigt eine Funktion $r : \mathbb{R}^n \to [0, \infty)$ mit der Eigenschaft

$$r(x) \begin{cases} = 0, & \text{falls } x \in \Sigma, \\ > 0, & \text{falls } x \notin \Sigma. \end{cases}$$

Dann minimiert man für eine geeignet gewählte Folge von Gewichtungsparametern $\{\eta_k\}_{k\in\mathbb{N}}$ mit $\eta_k > 0$ die unrestringierte Penalty-Funktion

$$P(x; \eta_k) := f(x) + \eta_k \cdot r(x). \tag{2.1}$$

Für jedes $\eta_k > 0$ erhält man eine Lösung $x^{[k]} := x(\eta_k)$ und es stellt sich die Frage, wie die Gewichtungsparameter η_k, $k \in \mathbb{N}$, gewählt werden müssen, damit die Folge $\{x^{[k]}\}_{k\in\mathbb{N}}$ gegen ein Minimum von (P) konvergiert.
Die Funktion r kann auf verschiedene Arten definiert werden, wobei differenzierbare Varianten ideal sind, um die uns bekannten Verfahren der unrestringierten Verfahren anwenden zu können. Wird r als stetige, aber nicht stetig differenzierbare Funktion gewählt, so gestaltet sich die Lösung des unrestringierten Penalty-Problems schwieriger.

2.1 Penalty-Verfahren

Wir illustrieren die Idee der Penalty-Verfahren an einem Beispiel.

Beispiel 2.1.1

Gegeben sei das folgende Optimierungsproblem:

$$\textit{Minimiere} \qquad f(x_1, x_2) := x_1 + x_2 \qquad \textit{u.d.N.} \qquad h(x_1, x_2) := x_1^2 - x_2 = 0.$$

Wir möchten nun die Nebenbedingung eliminieren. Wir könnten dies erreichen durch Auflösen nach x_2 und Einsetzen in die Zielfunktion. Hier verfolgen wir aber eine andere Strategie und koppeln einen Strafterm, der Punkte mit $x_1^2 - x_2 \neq 0$ bestraft, an die Zielfunktion. Ein solcher Strafterm kann z.B. mithilfe der Funktion

$$r(x_1, x_2) := (x_1^2 - x_2)^2 = h(x_1, x_2)^2$$

realisert werden, denn es gilt $r(x_1, x_2) = 0$ genau dann, wenn $h(x_1, x_2) = x_1^2 - x_2 = 0$ gilt. Beachte, dass r differenzierbar ist. Man hätte auch $|h(x_1, x_s)|$ verwenden können, aber diese Funktion ist nicht differenzierbar.
Anstelle von f minimiert man dann die sogenannte Penalty-Funktion

$$P(x_1, x_2; \eta) := f(x_1, x_2) + \frac{\eta}{2} r(x_1, x_2) = x_1 + x_2 + \frac{\eta}{2}(x_1^2 - x_2)^2,$$

wobei $\eta > 0$ ein Gewichtungsfaktor ist.
Zur Minimierung von P können Verfahren der unrestringierten Optimierung verwendet werden. Allerdings hängt die Lösung vom Gewichtungsfaktor η ab und es stellt sich die Frage, unter welchen Bedingungen diese Lösung gegen die Lösung des restringierten Ausgangsproblems konvergieren.
Dazu untersuchen wir die Minima von P in Abhängigkeit von η. Notwendig gilt

$$0 = \nabla_x P(x_1, x_2; \eta) = \begin{pmatrix} 1 + 2\eta x_1(x_1^2 - x_2) \\ 1 - \eta(x_1^2 - x_2) \end{pmatrix}.$$

Daraus ergeben sich die stationären Punkte

$$\begin{pmatrix} x_1(\eta) \\ x_2(\eta) \end{pmatrix} = \begin{pmatrix} -\frac{1}{2} \\ \frac{1}{4} - \frac{1}{\eta} \end{pmatrix}.$$

Wie hängen diese Lösungen mit der Lösung des Ausgangsproblems zusammen? Dazu bestimmen wir die stationären Punkte der Lagrange-Funktion

$$L(x_1, x_2, \mu) = x_1 + x_2 + \mu(x_1^2 - x_2)$$

und erhalten

$$0 = \nabla_x L(\hat{x}_1, \hat{x}_2, \mu) = \begin{pmatrix} 1 + 2\mu\hat{x}_1 \\ 1 - \mu \end{pmatrix} \qquad \Longleftrightarrow \qquad \begin{pmatrix} \hat{x}_1 \\ \hat{x}_2 \end{pmatrix} = \begin{pmatrix} -\frac{1}{2} \\ \frac{1}{4} \end{pmatrix}, \ \mu = 1.$$

Man erkennt, dass die Lösungen des Penalty-Problems für $\eta \longrightarrow \infty$ gegen die Lösung des restringierten Problems konvergieren:

$$\lim_{\eta\to\infty} \begin{pmatrix} x_1(\eta) \\ x_2(\eta) \end{pmatrix} = \lim_{\eta\to\infty} \begin{pmatrix} -\frac{1}{2} \\ \frac{1}{4} - \frac{1}{\eta} \end{pmatrix} = \begin{pmatrix} -\frac{1}{2} \\ \frac{1}{4} \end{pmatrix} = \begin{pmatrix} \hat{x}_1 \\ \hat{x}_2 \end{pmatrix}.$$

Man muss die Gewichtungsparameter also so wählen, dass sie gegen ∞ gehen. ■

Die Beobachtungen in Beispiel 2.1.1 gelten auch allgemein unter relativ schwachen Voraussetzungen. Dazu betrachten wir zunächst das folgende gleichungsrestringierte Optimierungsproblem mit stetigen (!) Funktionen $f : \mathbb{R}^n \to \mathbb{R}$ und $h_j : \mathbb{R}^n \to \mathbb{R}$, $j = 1, \ldots, p$.

Problem 2.1.2 **(Restringiertes Optimierungsproblem)**

$$\textit{Minimiere} \qquad f(x) \qquad \textit{u.d.N.} \quad x \in \Sigma$$

mit

$$\Sigma = \{x \in \mathbb{R}^n \mid h_j(x) = 0,\ j = 1, \ldots, p\}.$$

Die Idee des Penalty-Verfahrens besteht darin, die Lösung $\hat{x}$ des Ausgangsproblems iterativ durch die Lösungen von unrestringierten Hilfsproblemen zu approximieren. Diese Hilfsprobleme bestehen in der Minimierung der **Penalty-Funktion**

$$P(x; \eta) := f(x) + \frac{\eta}{2} \sum_{j=1}^{p} (h_j(x))^2$$

für geeignete Werte von $\eta > 0$. Durch die Ankopplung der Nebenbedingungen wird ein Verlassen des zulässigen Bereichs Σ „bestraft“. Die Konstante η stellt einen Gewichtungsfaktor dar, mit dessen Hilfe die „Stärke der Bestrafung“ gesteuert werden kann. Das Penalty-Verfahren ist durch folgenden Algorithmus gegeben.

Algorithmus 2.1.3 **(Penalty-Verfahren)**

(i) *Wähle $\eta_0 > 0$ und setze $k = 0$.*

(ii) *Bestimme $x^{[k]}$ als Lösung von*

$$\textit{Minimiere} \qquad P(x; \eta_k) \qquad \textit{u.d.N.} \qquad x \in \mathbb{R}^n\ .$$

(iii) *Ist $h(x^{[k]}) \approx 0$, STOP.*

(iv) *Bestimme $\eta_{k+1} > \eta_k$, setze $k := k + 1$ und gehe zu (ii).*

Da P i.A. nicht differenzierbar ist, werden für das Hilfsproblem in Schritt (ii) Verfahren der unrestringierten, nichtdifferenzierbaren Optimierung benötigt. Es stellt sich natürlich die Frage, ob das Verfahren tatsächlich gegen eine Lösung des Ausgangsproblems konvergiert.

Satz 2.1.4 (Konvergenzsatz für das Penalty-Verfahren)
Seien f und h_j, $j = 1, \dots, p$ stetig und $\{\eta_k\}$ streng monoton wachsend mit $\eta_k \longrightarrow \infty$. Die zulässige Menge $\Sigma := \{x \in \mathbb{R}^n \mid h_j(x) = 0, j = 1, \dots, p\}$ sei nichtleer, und $\{x^{[k]}\}$ sei eine durch Algorithmus 2.1.3 erzeugte Folge (die Existenz der Folge sei vorausgesetzt). Dann gelten die folgenden Aussagen:

(a) Die Folge der Zielfunktionswerte der Penalty-Funktion $\{P(x^{[k]}; \eta_k)\}_{k\in\mathbb{N}}$ ist monoton wachsend.

(b) Die Folge der Verletzung der Nebenbedingungen $\{\|h(x^{[k]})\|\}_{k\in\mathbb{N}}$ ist monoton fallend.

(c) Die Folge der Zielfunktionswerte $\{f(x^{[k]})\}_{k\in\mathbb{N}}$ ist monoton wachsend.

(d) Es gilt $\lim_{k\to\infty} h(x^{[k]}) = 0$.

(e) Jeder Häufungspunkt der Folge $\{x^{[k]}\}_{k\in\mathbb{N}}$ ist eine Lösung des Ausgangsproblems.

Beweis: (vgl. [GK02], S. 208)

(a) Aus $\eta_{k+1} > \eta_k$ und der Definition von $x^{[k]}$ folgt

$$P(x^{[k]}; \eta_k) \le P(x^{[k+1]}; \eta_k) \le P(x^{[k+1]}; \eta_{k+1}).$$

(b) Es gilt

$$P(x^{[k]}; \eta_k) + P(x^{[k+1]}; \eta_{k+1}) \le P(x^{[k+1]}; \eta_k) + P(x^{[k]}; \eta_{k+1}).$$

Mit der Definition von P folgt

$$\eta_k \|h(x^{[k]})\|^2 + \eta_{k+1}\|h(x^{[k+1]})\|^2 \le \eta_k \|h(x^{[k+1]})\|^2 + \eta_{k+1}\|h(x^{[k]})\|^2$$

bzw.

$$(\eta_k - \eta_{k+1}) \left(\|h(x^{[k]})\|^2 - \|h(x^{[k+1]})\|^2\right) \le 0.$$

Wegen $\eta_k < \eta_{k+1}$ folgt $\|h(x^{[k]})\|^2 \ge \|h(x^{[k+1]})\|^2$ für alle k.

(c) folgt aus $P(x^{[k]}; \eta_k) \le P(x^{[k+1]}; \eta_k)$ und Teil (b).

(d) Wegen $\Sigma \neq \emptyset$ folgt

$$f(x^{[k]}) \le P(x^{[k]}; \eta_k) \le \inf_{x\in\Sigma} P(x; \eta_k) = \inf_{x\in\Sigma} f(x) =: \hat{f} < \infty.$$

Wegen $\eta_k \to \infty$ und $f(x^{[k]}) \ge f(x^{[0]})$ nach (c) folgt $\lim_{k\to\infty} \|h(x^{[k]})\| = 0$.

(e) Sei $\hat{x}$ Häufungspunkt der Folge $\{x^{[k]}\}$ und $\{x^{[k_j]}\}$, $j = 1, 2, \ldots$ eine gegen $\hat{x}$ konvergente Teilfolge. Nach (d) gilt $h(\hat{x}) = 0$, d.h. $\hat{x}$ ist zulässig. Ausserdem gilt

$$f(\hat{x}) = \lim_{j\to\infty} f(x^{[k_j]}) \leq \lim_{j\to\infty} P(x^{[k_j]}; \eta_{k_j}) \leq \inf_{x\in\Sigma} f(x) =: \hat{f}.$$

Daraus folgt (e).

■

Bemerkung 2.1.5

Da nur die Stetigkeit der auftretenden Funktionen benötigt wird, ist das Verfahren auch auf Problemstellungen mit Ungleichungsnebenbedingungen

$$g_i(x) \leq 0, \quad i = 1, \ldots, m,$$

anwendbar. Denn diese Nebenbedingungen können äquivalent als stetige Nebenbedingungen

$$\max\{0, g_i(x)\} = 0, \quad i = 1, \ldots, m,$$

geschrieben werden. Die Penaltyfunktion lautet dann

$$P(x;\eta) = f(x) + \frac{\eta}{2}\sum_{j=1}^{p}(h_j(x))^2 + \frac{\eta}{2}\sum_{i=1}^{m}\left(\max\{0, g_i(x)\}\right)^2.$$

■

Ein wesentlicher Nachteil des Penalty-Verfahrens ist die Tatsache, dass die Gewichtungsfaktoren η_k gegen ∞ streben müssen, um Konvergenz zu erhalten. Dies führt dazu, dass die Teilprobleme in (ii) des Algorithmus für großes η_k sehr schlecht konditioniert sind[1] und numerisch nur sehr schwer zu lösen sind.

2.1.1 Schätzung der Lagrange-Multiplikatoren

Wir untersuchen, wie aus der Folge $\{x^{[k]}\}_{k\in\mathbb{N}}$ eine Folge $\{\mu^{[k]}\}_{k\in\mathbb{N}}$ von Näherungen der Lagrange-Multiplikatoren konstruiert werden kann, so dass $x^{[k]}$ und $\mu^{[k]}$ gegen einen KKT-Punkt $\hat{x}$ und $\hat{\mu}$ des Ausgangsproblems konvergieren.
Hierzu benötigen wir die stetige Differenzierbarkeit der Funktionen f und h_j, $j = 1, \ldots, p$. Ein KKT-Punkt $(\hat{x}, \hat{\mu})$ des Ausgangsproblems erfüllt

$$0 = \nabla f(\hat{x}) + \sum_{j=1}^{p} \hat{\mu}_j \nabla h_j(\hat{x}).$$

[1] Einige Eigenwerte der Hessematrix $\nabla^2_{xx} P(x^{[k]}; \eta_k)$ streben gegen ∞ und somit strebt die Spektral-Konditionszahl der Hessematrix gegen unendlich.

Da $x^{[k]}$ Minimalstelle der Penaltyfunktion mit Gewichtungsparameter η_k ist, gilt notwendig

$$0 = \nabla_x P(x^{[k]}; \eta_k) = \nabla f(x^{[k]}) + \eta_k \sum_{j=1}^{p} h_j(x^{[k]}) \nabla h_j(x^{[k]}).$$

Vergleicht man die beiden Ausdrücke, so liegt es nahe,

$$\mu_j^{[k]} = \eta_k h_j(x^{[k]}) \tag{2.2}$$

als Approximation der Lagrange-Multiplikatoren $\hat{\mu}_j$ zu verwenden.
Es gilt:

Satz 2.1.6

Seien f und h_j, $j = 1, \dots, p$, stetig differenzierbar und $\{x^{[k]}\}_{k\in\mathbb{N}}$ eine durch das Penalty-Verfahren erzeugte Folge mit $x^{[k]} \longrightarrow \hat{x}$ für $k \longrightarrow \infty$. Die Gradienten $\nabla h_j(\hat{x})$, $j = 1, \dots, p$, seien linear unabhängig, und $\{\mu^{[k]}\}_{k\in\mathbb{N}}$ sei durch (2.2) gegeben. Dann gelten:

(a) Die Folge $\{\mu^{[k]}\}_{k\in\mathbb{N}}$ konvergiert gegen einen Vektor $\hat{\mu}$.

(b) $(\hat{x}, \hat{\mu})$ ist ein KKT-Punkt des Ausgangsproblems.

Beweis: (vgl. [GK02], S. 211)

(a) Es seien $J_k := \frac{\partial h(x^{[k]})}{\partial x}$ und $\hat{J} := \frac{\partial h(\hat{x})}{\partial x}$ die Jacobimatrizen von $h = (h_1, \dots, h_p)$ in $x^{[k]}$ bzw. $\hat{x}$. Aus der Stetigkeit der Jacobimatrizen folgt $J_k \to \hat{J}$. Da die Gradienten $\nabla h_j(\hat{x})$ linear unabhängig sind, ist die Matrix $J_k J_k^\top$ regulär und es gilt $\left(J_k J_k^\top\right)^{-1} \to \left(\hat{J}\hat{J}^\top\right)^{-1}$. Da $x^{[k]}$ ein Minimum von $P(x; \eta_k)$ ist, gilt

$$\begin{aligned} 0 &= \nabla_x P(x^{[k]}; \eta_k) \\ &= \nabla f(x^{[k]}) + \eta_k \sum_{j=1}^{p} h_j(x^{[k]}) \nabla h_j(x^{[k]}) \\ &= \nabla f(x^{[k]}) + \sum_{j=1}^{p} \mu_j^{[k]} \nabla h_j(x^{[k]}) \\ &= \nabla f(x^{[k]}) + J_k^\top \mu^{[k]}. \end{aligned}$$

Multiplikation von links mit J_k liefert $J_k J_k^\top \mu^{[k]} = -J_k \nabla f(x^{[k]})$ bzw.

$$\mu^{[k]} = -\left(J_k J_k^\top\right)^{-1} J_k \nabla f(x^{[k]}) \to -\left(\hat{J}\hat{J}^\top\right)^{-1} \hat{J} \nabla f(\hat{x}) =: \hat{\mu}.$$

(b) Folgt aus

$$0 = \nabla_x P(x^{[k]}; \eta_k) = \nabla f(x^{[k]}) + \eta_k \sum_{j=1}^{p} h_j(x^{[k]}) \nabla h_j(x^{[k]})$$

und $\mu_j^{[k]} = \eta_k h_j(x^{[k]}) \to \hat{\mu}$.

□

2.2 Multiplikator-Penalty-Verfahren

Multiplikator-Penalty-Verfahren ähneln den Penalty-Verfahren. Allerdings arbeiten sie mit einer exakten und differenzierbaren Penalty-Funktion – der erweiterten Lagrangefunktion.
Wir betrachten wieder das gleichungsrestringierte Problem 2.1.2, d.h.

$$\text{Minimiere} \qquad f(x) \qquad \text{u.d.N.} \qquad h(x) = 0. \tag{2.3}$$

Darin seien die Funktionen $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ und $h = (h_1, \ldots, h_p)^\top : \mathbb{R}^n \longrightarrow \mathbb{R}^p$ zweimal stetig differenzierbar.

Sei $\hat{x}$ lokales Minimum des Problems. Dann ist $\hat{x}$ für $\eta > 0$ auch ein lokales Minimum von

$$\text{Minimiere} \qquad f(x) + \frac{\eta}{2}\|h(x)\|^2 \qquad \text{u.d.N.} \qquad h(x) = 0.$$

Die Lagrangefunktion für dieses Problem lautet

$$L_a(x, \mu; \eta) := f(x) + \frac{\eta}{2}\|h(x)\|^2 + \mu^\top h(x)$$

und heißt **erweiterte Lagrangefunktion** (**augmented Lagrangian**) oder **Multiplikator-Penalty-Funktion**.
Es zeigt sich, dass der Gewichtungsparameter η für L_a nicht gegen ∞ streben muss, um ein lokales Minimum des Ausgangsproblems zu erreichen.

Hilfsatz 2.2.1

Sei $(\hat{x}, \hat{\mu})$ KKT-Punkt von (2.3). Desweiteren sei die hinreichende Bedingung zweiter Ordnung (1.6) erfüllt. Dann existiert ein endliches $\bar{\eta} > 0$, so dass $\hat{x}$ für jedes $\eta \geq \bar{\eta}$ ein striktes lokales Minimum von $L_a(\cdot, \hat{\mu}; \eta)$ ist.

Beweis: siehe Geiger und Kanzow [GK02], S. 229 □

Auf Grund dieses Hilfssatzes kann man versuchen, das Ausgangsproblem (2.3) indirekt zu lösen, indem die erweiterte Lagrangefunktion minimiert wird:

$$\text{Minimiere} \qquad L_a(x, \hat{\mu}; \eta) \qquad \text{u.d.N.} \qquad x \in \mathbb{R}^n.$$

Der Penalty-Parameter η muß jetzt, anders als bei den Penalty-Verfahren, nicht mehr gegen ∞ streben. Darüber hinaus ist L_a differenzierbar, so dass bekannte Verfahren aus der unrestringierten Optimierung eingesetzt werden können.

Problem: Der optimale Lagrangemultiplikator $\hat{\mu}$ ist unbekannt.

Wir versuchen nun, $\hat{\mu}$ geeignet zu approximieren. Sei η hinreichend groß und $x^{[k+1]}$ stationärer Punkt des Problems

$$\text{Minimiere} \qquad L_a(x, \mu^{[k]}; \eta) \qquad \text{u.d.N.} \qquad x \in \mathbb{R}^n.$$

Dann gilt notwendig

$$0 = \nabla_x L_a(x^{[k+1]}, \mu^{[k]}; \eta) = \nabla f(x^{[k+1]}) + \sum_{j=1}^{p} \left(\mu_j^{[k]} + \eta h_j(x^{[k+1]}) \right) \nabla h_j(x^{[k+1]}).$$

Andererseits gilt in einem KKT-Punkt $(\hat{x}, \hat{\mu})$ von (2.3) notwendig

$$0 = \nabla_x L(\hat{x}, \hat{\mu}) = \nabla f(\hat{x}) + \sum_{j=1}^{p} \hat{\mu}_j \nabla h_j(\hat{x}).$$

Ein Vergleich beider Ausdrücke liefert die naheliegende Aufdatierungsvorschrift

$$\mu^{[k+1]} := \mu^{[k]} + \eta h(x^{[k+1]}).$$

Insgesamt entsteht das Multiplier-Penalty-Verfahren:

Algorithmus 2.2.2 (Multiplikator-Penalty-Verfahren)

(i) *Wähle* $x^{[0]} \in \mathbb{R}^n$, $\mu^{[0]} \in \mathbb{R}^p$, $\eta_0 > 0$, $\sigma \in (0,1)$ *und setze* $k = 0$.

(ii) *Ist* $(x^{[k]}, \mu^{[k]})$ *KKT-Punkt von (2.3), STOP.*

(iii) *Bestimme* $x^{[k+1]}$ *als Lösung von*

$$\textit{Minimiere} \qquad L_a(x, \mu^{[k]}; \eta_k) \qquad \textit{u.d.N.} \qquad x \in \mathbb{R}^n.$$

(iv) *Setze* $\mu^{[k+1]} := \mu^{[k]} + \eta_k h(x^{[k+1]})$.

(v) *Ist* $\|h(x^{[k+1]})\| \geq \sigma \|h(x^{[k]})\|$, *so setze* $\eta_{k+1} := 10\eta_k$. *Andernfalls setze* $\eta_{k+1} := \eta_k$.

(vi) *Setze* $k := k + 1$ *und gehe zu (ii).*

2.2.1 Anwendung auf Ungleichungen

Das Standard-Optimierungsproblem

$$\text{Minimiere} \qquad f(x) \qquad \text{u.d.N.} \qquad g(x) \le 0,\ h(x) = 0 \tag{2.4}$$

ist durch Einführung von Schlupfvariablen $s = (s_1, \dots, s_m)^\top \in \mathbb{R}^m$ äquivalent mit dem gleichungsrestringierten Problem

$$\begin{array}{lll} \text{Minimiere} & f(x) & \\ \text{bzgl.} & (x,s) \in \mathbb{R}^{n+m} & \\ \text{u.d.N.} & g_i(x) + s_i^2 = 0, & i = 1, \dots, m, \\ & h_j(x) = 0, & j = 1, \dots, p. \end{array}$$

Die erweiterte Lagrange-Funktion hierfür lautet

$$\bar{L}_a(x,s,\lambda,\mu;\eta) = f(x) + \frac{\eta}{2}\|h(x)\|^2 + \mu^\top h(x) + \sum_{i=1}^m \left(\lambda_i \left(g_i(x) + s_i^2\right) + \frac{\eta}{2}\left(g_i(x) + s_i^2\right)^2\right).$$

Für festes x kann die Minimierung bzgl. s explizit ausgeführt werden und man erhält

$$\hat{s}_i = \left(\max\left\{0, -\left(\frac{\lambda_i}{\eta} + g_i(x)\right)\right\}\right)^{1/2}, \quad i = 1, \dots, m.$$

Einsetzen in die erweiterte Lagrange-Funktion liefert

$$\begin{aligned} \bar{L}_a(x,\lambda,\mu;\eta) &= f(x) + \mu^\top h(x) + \frac{\eta}{2}\|h(x)\|^2 \\ &\quad + \frac{1}{2\eta}\sum_{i=1}^m \left(\left(\max\{0, \lambda_i + \eta g_i(x)\}\right)^2 - \lambda_i^2\right) \\ &= f(x) + \sum_{j=1}^p \left(\mu_j h_j(x) + \frac{\eta}{2} h_j(x)^2\right) \\ &\quad + \sum_{i=1}^m \begin{cases} \lambda_i g_i(x) + \frac{\eta}{2} g_i(x)^2, & \text{falls } \lambda_i + \eta g_i(x) \ge 0, \\ -\frac{\lambda_i^2}{2\eta}, & \text{sonst.} \end{cases} \end{aligned}$$

Beachte, dass diese Funktion nur noch stetig differenzierbar ist.
Für die Multiplikatoren ergeben sich die Aufdatierungsformeln

$$\begin{aligned} \mu^{[k+1]} &:= \mu^{[k]} + \eta h(x^{[k+1]}), \\ \lambda_i^{[k+1]} &:= \max\left\{0, \lambda_i^{[k]} + \eta g_i(x^{[k+1]})\right\}, \qquad i = 1, \dots, m. \end{aligned}$$

Wir schließen diesen Abschnitt mit einem Testbeispiel, bei dem die exakte Lösung zur Kontrolle auch direkt berechnet werden kann.

Beispiel 2.2.3

Gegeben sei das nichtlineare Optimierungsproblem

$$\text{Minimiere} \quad x^2 + y^2 \quad \text{u.d.N.} \quad x - y - 1 = 0\ .$$

Die erweiterte Lagrange-Funktion lautet

$$L_a(x,y,\mu,\eta) = x^2 + y^2 + \frac{\eta}{2}(x-y-1)^2 + \mu(x-y-1) \ .$$

Anwendung der Multiplier-Penalty-Methode mit Startwerten

$$(x^{[0]}, y^{[0]}) = (0,0) \ , \ \mu^{[0]} = 0 \ , \ \eta_0 = 1 \ , \ \sigma = 0.1$$

liefert das folgende Resultat:

K	X	Y	LAMBDA	ETA	F(X,Y)	|H(X,Y)|
0	0.00000E+00	0.00000E+00	0.00000E+00	0.10000E+01	0.00000E+00	0.10000E+01
1	0.25000E+00	-0.25000E+00	-0.50000E+00	0.10000E+02	0.12500E+00	0.50000E+00
2	0.47727E+00	-0.47727E+00	-0.95455E+00	0.10000E+02	0.45558E+00	0.45455E-01
3	0.49793E+00	-0.49793E+00	-0.99587E+00	0.10000E+02	0.49588E+00	0.41322E-02
4	0.49981E+00	-0.49981E+00	-0.99962E+00	0.10000E+02	0.49962E+00	0.37566E-03
5	0.49998E+00	-0.49998E+00	-0.99997E+00	0.10000E+02	0.49997E+00	0.34151E-04
6	0.50000E+00	-0.50000E+00	-0.10000E+01	0.10000E+02	0.50000E+00	0.31046E-05
7	0.50000E+00	-0.50000E+00	-0.10000E+01	0.10000E+02	0.50000E+00	0.28224E-06
8	0.50000E+00	-0.50000E+00	-0.10000E+01	0.10000E+02	0.50000E+00	0.25658E-07
9	0.50000E+00	-0.50000E+00	-0.10000E+01	0.10000E+02	0.50000E+00	0.23325E-08

■

Kapitel 3

SQP-Verfahren

Die sequentielle quadratische Programmierung (SQP) wird, z.B., in [Han77], [Pow78], [GMW81], [Sto85], [Sch81], [Sch83], [Alt02], [GK02] behandelt. Es existieren diverse Implementierungen, z.B. [Sch85], [Kra88], [GMSW98], [GMS02]. Zunächst diskutieren wir das lokale SQP-Verfahren (mit Schrittweite 1) und erweitern das lokale Verfahren dann durch eine Globalisierungsstrategie, die auf dem Armijo-Verfahren basiert.

3.1 Das lokale SQP-Verfahren

Zur Motivation des SQP-Verfahrens erinnern wir uns an das Lagrange-Newton-Verfahren. Das Lagrange-Newton-Verfahren eignet sich zur Lösung des gleichungsrestringierten Optimierungsproblems

$$\text{Minimiere} \qquad f(x) \qquad \text{u.d.N.} \qquad h(x) = 0,$$

wobei $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ und $h : \mathbb{R}^n \longrightarrow \mathbb{R}^p$ zweimal stetig differenzierbare Funktionen seien und $L(x, \mu) = f(x) + \mu^\top h(x)$ die Lagrange-Funktion bezeichnet. Das Lagrange-Newton-Verfahren entsteht durch Anwendung des Newton-Verfahrens auf die KKT-Bedingungen

$$\nabla_x L(x, \mu) = 0 \qquad \text{und} \qquad h(x) = 0$$

und lautet wie folgt:

Algorithmus 3.1.1 (Lagrange-Newton-Verfahren)

(i) Wähle Startschätzungen $x^{[0]} \in \mathbb{R}^n$ *und* $\mu^{[0]} \in \mathbb{R}^p$, $\varepsilon > 0$ *und setze* $k = 0$.

(ii) Falls $\max\{\|\nabla_x L(x^{[k]}, \mu^{[k]})\|, \|h(x^{[k]})\|\} \leq \varepsilon$, *STOP.*

(iii) Löse das lineare Gleichungssystem

$$\begin{pmatrix} \nabla^2_{xx} L(x^{[k]}, \mu^{[k]}) & h'(x^{[k]})^\top \\ h'(x^{[k]}) & 0 \end{pmatrix} \begin{pmatrix} d \\ v \end{pmatrix} = - \begin{pmatrix} \nabla_x L(x^{[k]}, \mu^{[k]}) \\ h(x^{[k]}) \end{pmatrix} \tag{3.1}$$

und setze

$$x^{[k+1]} := x^{[k]} + d, \qquad \mu^{[k+1]} := \mu^{[k]} + v. \tag{3.2}$$

(iv) Setze $k := k+1$ und gehe zu (ii).

Wir starten zunächst mit einer Beobachtung. Das lineare Gleichungssystem (3.1) in (iii) des Lagrange-Newton-Verfahrens entsteht auch auf andere Art. Wir erinnern uns an das Newton-Verfahren für unrestringierte Optimierungsprobleme. Dort hatten wir das Newtonverfahren auf zwei Arten motiviert: 1. Anwendung des Newtonverfahrens auf die notwendigen Bedingung $\nabla f = 0$ (indirekter Ansatz); 2. lokale Approximation der Zielfunktion durch eine quadratische Funktion (direkter Ansatz). Beide Ansätze lieferten das gleiche Verfaren.
Zur Abkürzung setzen wir im Folgenden

$$Q_k := \nabla^2_{xx} L(x^{[k]}, \mu^{[k]}).$$

Wir betrachten nun wieder das gleichungsrestringierte Optimierungsproblem und approximieren es lokal im Punkt $(x^{[k]}, \mu^{[k]})$ durch das **quadratische Optimierungsproblem**

$$\begin{aligned} &\text{Minimiere} && \frac{1}{2} d^\top Q_k d + \nabla f(x^{[k]})^\top d \\ &\text{bzgl.} && d \in \mathbb{R}^n \\ &\text{u.d.N.} && h(x^{[k]}) + h'(x^{[k]}) d &= 0. \end{aligned}$$

Die Lagrange-Funktion für das quadratische Optimierungsproblem ist gegeben durch

$$L_{QP}(d, \eta) := \frac{1}{2} d^\top Q_k d + \nabla f(x^{[k]})^\top d + \eta^\top \left(h(x^{[k]}) + h'(x^{[k]}) d \right).$$

Auswertung der KKT-Bedingungen führt auf das lineare Gleichungssystem

$$\begin{aligned} Q_k d + \nabla f(x^{[k]}) + h'(x^{[k]})^\top \eta &= 0, \\ h(x^{[k]}) + h'(x^{[k]}) d &= 0, \end{aligned}$$

bzw.

$$\begin{pmatrix} Q_k & h'(x^{[k]})^\top \\ h'(x^{[k]}) & 0 \end{pmatrix} \begin{pmatrix} d \\ \eta \end{pmatrix} = - \begin{pmatrix} \nabla f(x^{[k]}) \\ h(x^{[k]}) \end{pmatrix}. \tag{3.3}$$

Subtraktion von $h'(x^{[k]})^\top \mu^{[k]}$ auf beiden Seiten der ersten Gleichung in (3.3) liefert das lineare Gleichungssystem

$$\begin{pmatrix} Q_k & h'(x^{[k]})^\top \\ h'(x^{[k]}) & 0 \end{pmatrix} \begin{pmatrix} d \\ \eta - \mu^{[k]} \end{pmatrix} = - \begin{pmatrix} \nabla_x L(x^{[k]}, \mu^{[k]}) \\ h(x^{[k]}) \end{pmatrix}. \tag{3.4}$$

Ein Vergeich von (3.4) mit (3.1) zeigt, dass diese zwei Gleichungssysteme identisch sind, wenn man noch $v := \eta - \mu^{(k)}$ definiert. Die neuen Iterierten in (3.2) lassen sich damit wie folgt berechnen:

$$x^{[k+1]} = x^{[k]} + d, \qquad \mu^{[k+1]} = \mu^{[k]} + v = \eta.$$

Zusammenfassung:
Für gleichungsrestringierte Optimierungsprobleme ist das Lagrange-Newton-Verfahren identisch mit dem oben hergeleiteten sukzessiven quadratischen Optimierungsverfahren, wenn der Multiplikator η des quadratischen Hilfsproblems als neue Approximation für den Multiplikator μ des Ausgangsproblems verwendet wird.

Diese Beobachtung motiviert die folgende Erweiterung des quadratischen Hilfsproblems für Standard-Optimierungsprobleme mit **Gleichungs- und Ungleichungsrestriktionen**:

Problem 3.1.2 **(QP Problem $QP(x^{[k]}, \lambda^{[k]}, \mu^{[k]})$)**

$$\begin{aligned} &\textit{Minimiere} && \frac{1}{2} d^\top Q_k d + \nabla f(x^{[k]})^\top d \\ &\textit{bzgl.} && d \in \mathbb{R}^n \\ &\textit{u.d.N.} && g(x^{[k]}) + g'(x^{[k]}) d \;\leq\; 0, \\ & && h(x^{[k]}) + h'(x^{[k]}) d \;=\; 0. \end{aligned}$$

Sukzessive quadratische Approximation liefert das lokale SQP Verfahren:

Algorithmus 3.1.3 **(Lokales SQP Verfahren)**

(i) *Wähle Startwerte* $(x^{[0]}, \lambda^{[0]}, \mu^{[0]}) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p$ *und setze* $k = 0$.

(ii) *Falls* $(x^{[k]}, \lambda^{[k]}, \mu^{[k]})$ *ein KKT-Punkt des Standard-Optimierungsproblems ist, STOP.*

(iii) *Berechne einen KKT-Punkt* $(d^{[k]}, \lambda^{[k+1]}, \mu^{[k+1]}) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p$ *des quadratischen Optimierungsproblems* $QP(x^{[k]}, \lambda^{[k]}, \mu^{[k]})$.

(iv) *Setze* $x^{[k+1]} := x^{[k]} + d^{[k]}$, $k := k + 1$ *und gehe zu (ii).*

Bemerkung 3.1.4

- *Es ist nicht notwendig, die Indexmenge $A(\hat{x})$ der aktiven Ungleichungsnebenbedingungen im Voraus zu kennen.*
- *Die Iterierten $x^{[k]}$ sind in der Regel nicht zulässig, d.h. es gilt i.A. $x^{[k]} \notin \Sigma$.*

■

Die lokale Konvergenz des SQP Verfahrens wird im folgenden Satz formuliert.

Satz 3.1.5 (Lokale Konvergenz des SQP-Verfahrens)
Voraussetzungen:

(i) $\hat{x}$ ist lokales Minimum des Standard-Optimierungsproblems und $\hat{\lambda}$ und $\hat{\mu}$ bezeichnen die Lagrange-Multiplikatoren.

(ii) Die Funktionen f, g_i, $i = 1, \dots, m$, und h_j, $j = 1, \dots, p$, sind zweimal stetig differenzierbar mit Lipschitz-stetigen zweiten Ableitungen.

(iii) Es gilt die Linear Independence Constraint Qualification (LICQ) in $\hat{x}$.

(iv) Die strikte Komplementaritätsbedingung $\hat{\lambda}_i - g_i(\hat{x}) > 0$ ist für alle $i \in A(\hat{x})$ erfüllt.

(v) Es gilt die hinreichende Bedingung zweiter Ordnung:

$$d^\top \nabla^2_{xx} L(\hat{x}, \hat{\lambda}, \hat{\mu}) d > 0$$

für alle $0 \neq d \in \mathbb{R}^n$ mit

$$\nabla g_i(\hat{x})^\top d = 0, \quad i \in A(\hat{x}), \quad \nabla h_j(\hat{x})^\top d = 0, \quad j = 1, \dots, p.$$

Dann existieren Umgebungen U von $(\hat{x}, \hat{\lambda}, \hat{\mu})$ und V von $(0, \hat{\lambda}, \hat{\mu})$, so dass alle quadratischen Optimierungsprobleme $QP(x^{[k]}, \lambda^{[k]}, \mu^{[k]})$ für beliebige Startwerte

$$(x^{[0]}, \lambda^{[0]}, \mu^{[0]}) \in U$$

in V eine eindeutige lokale Lösung $d^{[k]}$ mit eindeutigen Multiplikatoren $\lambda^{[k+1]}$ und $\mu^{[k+1]}$ besitzen.
Desweiteren konvergiert die Folge $\{(x^{[k]}, \lambda^{[k]}, \mu^{[k]})\}_{k \in \mathbb{N}}$ quadratisch gegen $(\hat{x}, \hat{\lambda}, \hat{\mu})$.

Beweis: Der Beweis kann mithilfe des Sensitivitätssatzes 6.2.11 in Kapitel 6 geführt werden. Dazu interpretiert man $QP(x^{[k]}, \lambda^{[k]}, \mu^{[k]})$ als gestörtes Problem von $QP(\hat{x}, \hat{\lambda}, \hat{\mu})$

und zeigt, dass die Indexmenge der aktiven Ungleichungsrestriktionen sich in der Nähe von $\hat{x}$ nicht ändert. Somit geht das SQP-Verfahren lokal in das Lagrange-Newton-Verfahren über (da man die inaktiven Restriktionen lokal vernachlässigen kann) und erbt die Konvergenzeigenschaften des Lagrange-Newton-Verfahrens (also des Newton-Verfahrens). ∎

Bemerkung 3.1.6 (Approximation der Hessematrix)
Die Verwendung der exakten Hessematrix $Q_k = \nabla^2_{xx} L(x^{[k]}, \lambda^{[k]}, \mu^{[k]})$ im QP-Problem hat zwei Nachteile:

- *In vielen Anwendungen ist die Hessematrix nicht explizit bekannt. Die numerische Approximation durch finite Differenzen ist sehr aufwendig und ungenau.*
- *Die Hessematrix kann indefinit sein. Dies erschwert die Lösung der QP-Hilfsprobleme erheblich. Es ist daher wünschenswert, die Hessematrix durch eine positiv definite Matrix zu ersetzen ($\longrightarrow$ Idee der Quasi-Newton-Verfahren).*

In der Praxis wird die Hessematrix der Lagrange-Funktion Q_k in Iteration k durch eine geeignete Matrix H_k ersetzt. Powell [Pow78] schlug vor, die modifizierte BFGS-Update-Formel

$$H_{k+1} = H_k + \frac{q^{[k]}(q^{[k]})^\top}{(q^{[k]})^\top d^{[k]}} - \frac{H_k d^{[k]} (d^{[k]})^\top H_k}{(d^{[k]})^\top H_k d^{[k]}}, \tag{3.5}$$

mit

$$\begin{aligned}
d^{[k]} &= x^{[k+1]} - x^{[k]}, \\
q^{[k]} &= \theta_k y^{[k]} + (1-\theta_k) H_k d^{[k]}, \\
y^{[k]} &= \nabla_x L(x^{[k+1]}, \lambda^{[k]}, \mu^{[k]}) - \nabla_x L(x^{[k]}, \lambda^{[k]}, \mu^{[k]}), \\
\theta_k &= \begin{cases} 1, & \text{falls } (d^{[k]})^\top y^{[k]} \geq 0.2 (d^{[k]})^\top H_k d^{[k]}, \\ \frac{0.8 (d^{[k]})^\top H_k d^{[k]}}{(d^{[k]})^\top H_k d^{[k]} - (d^{[k]})^\top y^{[k]}}, & \text{sonst} \end{cases}
\end{aligned}$$

zu verwenden. Diese Update-Formel garantiert, dass H_{k+1} symmetrisch und positiv definit bleibt, wenn H_k symmetrisch und positiv definit war. Für $\theta_k = 1$ entsteht die BFGS-Formel, welche schon bei den Quasi-Newton-Verfahren verwendet wurde (allerdings musste dort durch Wahl einer geeigneten Schrittweitenstrategie noch die Bedingung $(d^{[k]})^\top y^{[k]} > 0$ garantiert werden).
Wird die modifizierte BFGS-Update-Formel im SQP-Verfahren verwendet, kann immerhin noch superlineare Konvergenz nachgewiesen werden. ∎

Bemerkung 3.1.7 (Große, dünn besetzte Optimierungsprobleme)
Für große, dünn besetzte Optimierungsprobleme eignet sich der modifizierte BFGS-Update nicht, da die Matrizen H_k in der Regel dicht besetzt sind und sie somit zuviel Speicher und Rechenkapazität beim Lösen der Gleichungssysteme benötigen würden. Die effiziente Berechnung oder Approximation der Hessematrix für große, dünn besetzte Pro-

bleme ist eine Wissenschaft für sich und es gibt verschiedene Ansätze, z.B. Limited-Memory BFGS-Verfahren oder Finite-Differenzen-Approximationen mit Färbungsalgorithmen (Graph-Coloring). Darüber hinaus entstehen bei großen, dünn besetzten Optimierungsproblemen noch weitere Herausforderungen, um die hohen Dimensionen (Millionen von Variablen und Restriktionen sind keine Seltenheit) beherrschen zu können. Diese Herausforderungen betreffen in erster Linie Lösungsverfahren für große, dünn besetzte lineare Gleichungssysteme. ∎

3.2 Globalisierung des SQP-Verfahrens

Das Konvergenzresultat zeigt, dass das SQP-Verfahren für alle Startwerte, die in einer Umgebung eines lokalen Minimums liegen, konvergent ist. In der Praxis ist diese Umgebung jedoch unbekannt und kann sehr klein sein. Daher ist es notwendig, das SQP-Verfahren zu globalisieren, so dass es (unter geeigneten Bedingungen) für beliebige Startwerte konvergiert. Wie im unrestringierten Fall wird dies durch Einführung einer Schrittweite $\alpha_k > 0$ erreicht. Die neue Iterierte ist gegeben durch

$$x^{[k+1]} = x^{[k]} + \alpha_k d^{[k]},$$

wobei $d^{[k]}$ wie zuvor ein quadratisches Hilfsproblem löst. Zur Bestimmung der Schrittweite α_k wird wieder eine eindimensionale **Liniensuche** in Richtung $d^{[k]}$ durchgeführt. Im Unterschied zur unrestringierten Optimierung tritt jetzt allerdings das folgende **Problem** auf:

Wann ist $x^{[k+1]}$ „besser“ als $x^{[k]}$?

Im unrestringierten Fall konnte diese Frage leicht durch einen Vergleich der Zielfunktionswerte beantwortet werden: $x^{[k+1]}$ ist besser als $x^{[k]}$, wenn $f(x^{[k+1]}) < f(x^{[k]})$ gilt.
Im restringierten Fall ist dies nicht mehr so einfach, da die Iterierten $x^{[k]}$ des SQP-Verfahrens i.a. unzulässig sind. Eine Verbesserung kann also sowohl an Hand der Zielfunktionswerte als auch an Hand der Verletzungen der Nebenbedingungen gemessen werden. Dies sind i.A. zwei miteinander konkurrierende Kriterien, da man einen besseren Zielfunktionswert leicht auf Kosten der Zulässigkeit erreichen kann und umgekehrt.
Ein Ansatz, um dieses Dilemma aufzulösen, besteht in der Verwendung von sogenannten **Bewertungsfunktionen** (engl. **merit functions**), die im einfachsten Fall Zielfunktion und Verletzung der Nebenbedingungen gewichtet in einer skalarwertigen Funktion vereinen ($\longrightarrow$ Idee der Penalty-Funktion).
Mithilfe der Bewertungsfunktion ist es möglich zu entscheiden, ob die neue Iterierte $x^{[k+1]}$ „besser“ ist als die alte Iterierte $x^{[k]}$. Dabei ist die neue Iterierte besser als die alte,

falls entweder ein hinreichender Abstieg in der Zielfunktion f oder eine weniger starke Verletzung der Nebenbedingungen erreicht wird, wobei sich das jeweils andere Kriterium nicht substantiell verschlechtern darf.
Eine allgemeine Klasse von Bewertungsfunktionen wird durch

$$P_r(x;\eta) := f(x) + \eta \cdot r(x) \tag{3.6}$$

definiert (vgl. (2.1) in Kapitel 2), wobei $\eta > 0$ einen Gewichtungsparameter und $r : \mathbb{R}^n \longrightarrow [0,\infty)$ eine stetige Funktion mit der Eigenschaft

$$r(x) \begin{cases} = 0, & \text{falls } x \in \Sigma, \\ > 0, & \text{falls } x \notin \Sigma \end{cases}$$

bezeichnen.

Beispiel 3.2.1 (Bewertungsfunktion)
Eine typische Bewertungsfunktion für das Standard-Optimierungsproblem, die auf der 1-Norm basiert, ist die ℓ_1-Bewertungsfunktion:

$$\ell_1(x;\eta) := f(x) + \eta \left(\sum_{i=1}^{m} \max\{0, g_i(x)\} + \sum_{j=1}^{p} |h_j(x)| \right), \qquad \eta > 0.$$

Beachte, dass unzulässige Punkte $x \notin \Sigma$ durch die Terme

$$\sum_{i=1}^{m} \max\{0, g_i(x)\} + \sum_{j=1}^{p} |h_j(x)| > 0$$

bestraft werden. Desweiteren ist ℓ_1 Lipschitz-stetig (falls f, g_i und h_j differenzierbar sind), aber nicht differenzierbar.
Allgemeinere Bewertungsfunktionen basieren auf der q-Norm:

$$\ell_q(x;\eta) := f(x) + \eta \left(\sum_{i=1}^{m} (\max\{0, g_i(x)\})^q + \sum_{j=1}^{p} |h_j(x)|^q \right)^{1/q}, \ 1 \le q < \infty,$$

und

$$\ell_\infty(x;\eta) := f(x) + \eta \max\{0, g_1(x), \ldots, g_m(x), |h_1(x)|, \ldots, |h_p(x)|\}.$$

■

Von besonderem Interesse sind die sogenannten exakten Bewertungsfunktionen, da für diese Bewertungsfunktionen lokale Minima des restringierten Ausgangsproblems auch lokale Minima der unrestringierten Bewertungsfunktion sind und der Gewichtungsparameter η dabei endlich gewählt werden kann.

Definition 3.2.2 **(Exakte Bewertungsfunktion)**

Die Bewertungsfunktion $P_r(x;\eta)$ in (3.6) heißt **exakt in einem lokalen Minimum $\hat{x}$ des Standard-Optimierungsproblems**, *falls es einen* **endlichen** *(!) Parameter $\hat{\eta} > 0$ gibt, so dass $\hat{x}$ ein lokales Minimum von $P_r(\cdot;\eta)$ für alle $\eta \geq \hat{\eta}$ ist.*

Beispiel 3.2.3

Abbildung 3.1 zeigt die ℓ_1-Bewertungsfunktion für verschiedene Werte von η für das Optimierungsproblem mit den Daten

$$\begin{aligned} f(x,y) &= (x-2)^2 + (y-3)^2, \\ h(x,y) &= y + \frac{x}{2} - \frac{1}{2}, \\ g_1(x,y) &= y + 2x^2 - 2, \\ g_2(x,y) &= x^2 - y - 1. \end{aligned}$$

Die optimale Lösung ist gegeben durch $\hat{x} = (3/5, 1/5)^\top$, $\hat{\lambda} = (0,0)^\top$ und $\hat{\mu} = 28/5$. Die Restriktionen g_1 und g_2 sind nicht aktiv in $\hat{x}$.

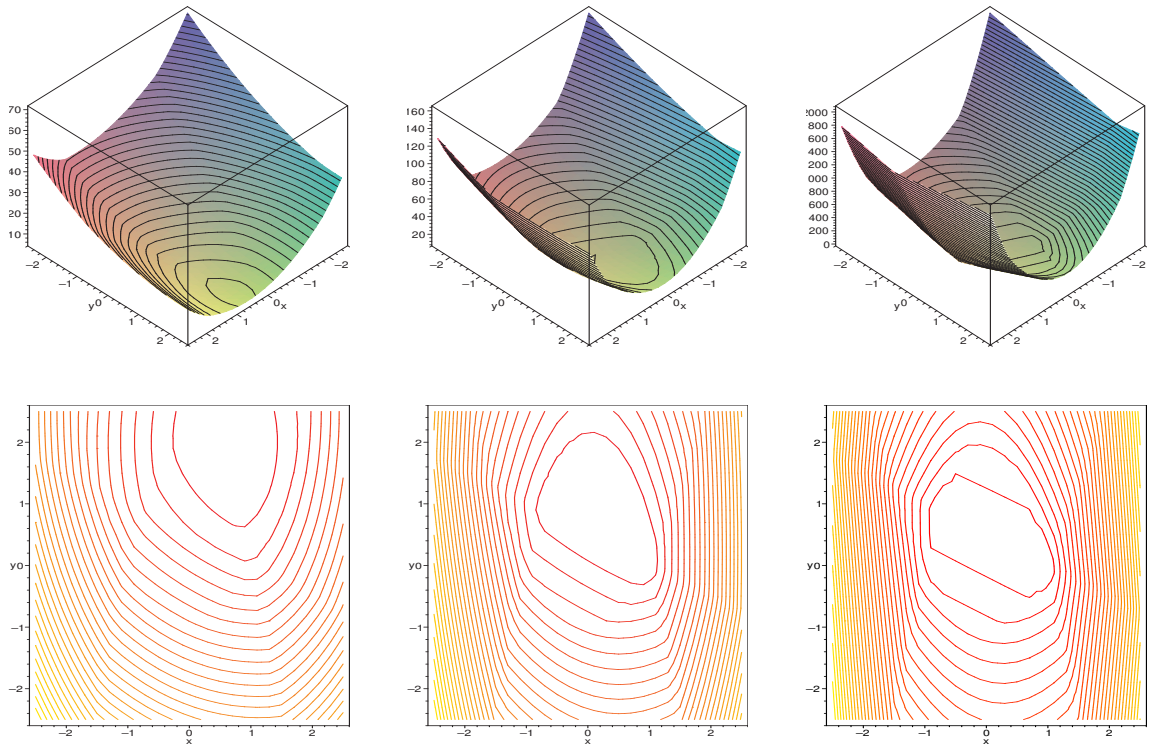


Abbildung 3.1: 3D-Darstellung (oben) und Höhenlinien (unten) der l_1-Bewertungsfunktion für $\eta = 1$ (links), $\eta = 28/5$ (Mitte) und $\eta = 100$ (rechts).

■

Es wäre wünschenswert, eine differenzierbare exakte Bewertungsfunktion zu haben. Dummerweise kann gezeigt werden, dass Bewertungsfunktionen der Form $P_r(x;\eta)$ aus (3.6) in einem lokalen Minimum $\hat{x}$ stets **nicht differenzierbar** sind, falls sie exakt sind und $\nabla f(\hat{x}) \neq 0$ gilt (letzteres ist der Normalfall in der restringierten Optimierung).
Der folgende Satz sagt aus, dass die Bewertungsfunktionen ℓ_q für $1 \leq q \leq \infty$ exakt sind, wenn eine Regularitätsbedingung gilt.

Satz 3.2.4

Sei $\hat{x} \in \Sigma$ ein isoliertes lokales Minimum des Standard-Optimierungsproblems, welches die LICQ erfüllt. Dann ist ℓ_q exakt für $1 \leq q \leq \infty$.

Beweis: Geiger and Kanzow [GK02, S. 225] zeigen diesen Satz unter der schwächeren Annahme, dass anstatt LICQ nur die Mangasarian-Fromowitz-Bedingung gilt. ■

Im Folgenden beschränken wir uns auf die ℓ_1-Bewertungsfunktion. Es liegt nun nahe, das restringierte Standard-Optimierungsproblem für hinreichend großes $\eta > 0$ durch das unrestringierte Minimierungsproblem

$$\text{Minimiere} \qquad \ell_1(x;\eta) \qquad \text{u.d.N.} \qquad x \in \mathbb{R}^n$$

zu ersetzen. Diese Idee wird im SQP-Verfahren ausgenutzt, um eine Schrittweite α mittels eindimensionaler Liniensuche ($\longrightarrow$ Armijo-Verfahren) für die Funktion

$$\varphi(\alpha) := \ell_1(x^{[k]} + \alpha d^{[k]};\eta)$$

durchzuführen. Wie oben erwähnt, ist die exakte ℓ_1-Bewertungsfunktion nicht differenzierbar. Allerdings ist sie immerhin noch richtungsdifferenzierbar, d.h. der Grenzwert

$$\ell_1'(x;d;\eta) := \lim_{\alpha \downarrow 0} \frac{\ell_1(x+\alpha d;\eta) - \ell_1(x;\eta)}{\alpha}$$

existiert für alle $x \in \mathbb{R}^n$ und alle Richtungen $d \in \mathbb{R}^n$ (vgl. Geiger und Kanzow [GK02, S. 252]) und man kann zeigen, dass ein KKT-Punkt $(d^{[k]}, \lambda^{[k+1]}, \mu^{[k+1]})$ mit $d^{[k]} \neq 0$ des QP-Problems $QP(x^{[k]}, \lambda^{[k]}, \mu^{[k]})$ die Abschätzung

$$\ell_1'(x^{[k]}; d^{[k]}; \eta) \leq -(d^{[k]})^\top Q_k d^{[k]} < 0$$

erfüllt, falls

- Q_k symmetrisch und positiv definit ist (was bei Verwendung des modifizierten BFGS-Updates H_k erfüllt ist) und

- der Gewichtungsparameter η die Bedingung

$$\eta \geq \max\{\lambda_1^{[k+1]}, \dots, \lambda_m^{[k+1]}, |\mu_1^{[k+1]}|, \dots, |\mu_p^{[k+1]}|\} \tag{3.7}$$

erfüllt. Hierin bezeichnen $\lambda_i^{[k+1]}$, $i = 1, \dots, m$, und $\mu_j^{[k+1]}$, $j = 1, \dots, p$, die Lagrange-Multiplikatoren des QP-Problems.

Fazit:
Eine Liniensuche mit dem Armijo-Verfahren kann durchgeführt werden, wenn die Hessematrix Q_k positiv definit ist oder alternativ der modifizierte BFGS-Update H_k im QP verwendet wird und wenn der Gewichtungsparameter hinreichend groß gewählt wird, was man durch iteratives Anpassen z.B. gemäß der Formel

$$\eta_{k+1} := \max\{\eta_k, \max\{\lambda_1^{[k+1]}, \dots, \lambda_m^{[k+1]}, |\mu_1^{[k+1]}|, \dots, |\mu_p^{[k+1]}|\} + \varepsilon\}, \tag{3.8}$$

erreichen kann ($\varepsilon \geq 0$ ist ein Parameter).

Insgesamt erhalten wir das globalisierte SQP-Verfahren:

Algorithmus 3.2.5 (Globalisiertes SQP-Verfahren)

(i) *Wähle Startwerte* $(x^{[0]}, \lambda^{[0]}, \mu^{[0]}) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p$, $H_0 \in \mathbb{R}^{n \times n}$ *symmetrisch und positiv definit,* $\beta \in (0,1)$, $\sigma \in (0,1)$ *und setze* $k = 0$.

(ii) *Falls* $(x^{[k]}, \lambda^{[k]}, \mu^{[k]})$ *ein KKT-Punkt des Standard-Optimierungsproblems ist, STOP.*

(iii) **QP-Hilfsproblem:**
Berechne einen KKT-Punkt $(d^{[k]}, \lambda^{[k+1]}, \mu^{[k+1]}) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p$ *des quadratischen Hilfsproblems* $QP(x^{[k]}, \lambda^{[k]}, \mu^{[k]})$, *wobei die Hessematrix* Q_k *durch die modifizierte BFGS-Update-Matrix* H_k *ersetzt ist.*

(iv) *Wähle* $\eta^{[k]}$ *hinreichend groß, z.B. gemäß (3.8).*

(v) **Armijo-Regel:**
Bestimme eine Schrittweite $\alpha_k = \max\{\beta^j \mid j = 0, 1, 2, \dots\}$ *mit*

$$\ell_1(x^{[k]} + \alpha_k d^{[k]}; \eta^{[k]}) \leq \ell_1(x^{[k]}; \eta^{[k]}) + \sigma \alpha_k \ell_1'(x^{[k]}; d^{[k]}; \eta^{[k]}).$$

(vi) **Modifizierter BFGS-Update:**
Berechne H_{k+1} *gemäß der Update-Formel (3.5).*

(vii) *Setze* $x^{[k+1]} := x^{[k]} + \alpha_k d^{[k]}$, $k := k + 1$ *und gehe zu (ii).*

Beispiel 3.2.6

Minimiere

$$f(x) = -x_2x_6 + x_1x_7 - x_3x_7 - x_5x_8 + x_4x_9 + x_3x_8$$

bzgl. $x \in \mathbb{R}^9$ *unter den Nebenbedingungen*

$$x_1 \geq 0,\ -1 \leq x_3 \leq 1,\ x_5 \geq 0,\ x_6 \geq 0,\ x_7 \geq 0,\ x_8 \leq 0,\ x_9 \leq 0$$

und

$$x_2 - x_1 \geq 0, \quad x_3 - x_2 \geq 0, \quad x_3 - x_4 \geq 0, \quad x_4 - x_5 \geq 0$$

und

$$\begin{aligned}
x_1^2 + x_6^2 &\leq 1,\\
(x_2 - x_1)^2 + (x_7 - x_6)^2 &\leq 1,\\
(x_3 - x_1)^2 + x_6^2 &\leq 1,\\
(x_1 - x_4)^2 + (x_6 - x_8)^2 &\leq 1,\\
(x_1 - x_5)^2 + (x_6 - x_9)^2 &\leq 1,\\
x_2^2 + x_7^2 &\leq 1,\\
(x_3 - x_2)^2 + x_7^2 &\leq 1,\\
(x_4 - x_2)^2 + (x_8 - x_7)^2 &\leq 1,\\
(x_2 - x_5)^2 + (x_7 - x_9)^2 &\leq 1,\\
(x_4 - x_3)^2 + x_8^2 &\leq 1,\\
(x_5 - x_3)^2 + x_9^2 &\leq 1,\\
x_4^2 + x_8^2 &\leq 1,\\
(x_4 - x_5)^2 + (x_9 - x_8)^2 &\leq 1,\\
x_5^2 + x_9^2 &\leq 1.
\end{aligned}$$

Startschätzung:

$$x^{[0]} = (0.1, 0.125, 2/3, 0.142857, 1/9, 0.2, 0.25, -0.2, -0.25)^\top.$$

Die Startschätzung der Multiplikatoren ist Null.

Ausgabe des SQP-Verfahrens:

```
---------------------------------------- SQP VERSION 1.1 (C) Matthias Gerdts, University of Bayreuth, 2004 ----------------------------------------

      NUMBER OF VARIABLES             :         9
      NUMBER OF CONSTRAINTS           :        18
      METHOD                          :  SEQUENTIAL QUADRATIC PROGRAMMING (SQP)
      MERIT FUNCTION                  :  L1-PENALTY FUNCTION
      MULTIPLIER UPDATE RULE          :  POWELL
      OPTIMALITY TOLERANCE            :  0.149E-07
      FEASIBILITY TOLERANCE           :  0.100E-11
      LINE SEARCH PARAMETER           :  SIGMA=  0.100E+00 BETA=  0.900E+00
      MAXIMUM NUMBER OF ITERATIONS    :     10000
      INFINITY                        :  0.100E+21
      ROUNDOFF TOLERANCE              :  0.300E-12
      REAL WORK SPACE PROVIDED        :       5338  NEEDED :       5338
      INTEGER WORK SPACE PROVIDED     :         82  NEEDED :         82
```

```
------------------------------------------------------------------------------------------------------------------------------
 ITER    QPIT     ALPHA                OBJ                NB          KKT         PEN         |D|         DELTA       RDELTA        F/G
------------------------------------------------------------------------------------------------------------------------------
   0       0    0.0000E+00    -0.3134920277777778E+00   0.0000E+00  0.5667E+00  0.4243E+01  0.0000E+00  0.0000E+00  0.1000E+01     1/   1    i
   1       7    0.1000E+01    -0.1593788677043482E+01   0.7834E+00  0.1217E+01  0.2138E+01  0.8139E+00  0.0000E+00  0.1000E+01     2/   2    i
   2      10    0.1000E+01    -0.1520709520745054E+01   0.3302E+00  0.2969E+00  0.1236E+01  0.4374E+00  0.3790E+00  0.1000E+01     3/   3    i
   3       3    0.1000E+01    -0.1388511910024717E+01   0.7326E-01  0.7589E-01  0.7931E+00  0.1180E+00  0.1801E+00  0.1000E+01     4/   4    i
   4       3    0.1000E+01    -0.1352359503836702E+01   0.4336E-02  0.9657E-02  0.6253E+00  0.4386E-01  0.3948E-01  0.1000E+01     5/   5    i
   5       3    0.1000E+01    -0.1349928698009139E+01   0.2504E-04  0.4006E-02  0.5547E+00  0.7860E-02  0.2493E-02  0.1000E+01     6/   6    i
   6       3    0.1000E+01    -0.1350018091962160E+01   0.7467E-04  0.3736E-02  0.5284E+00  0.1358E-01  0.1478E-04  0.1000E+01     7/   7    i
   7       3    0.1000E+01    -0.1349983850538642E+01   0.2758E-04  0.1499E-02  0.5174E+00  0.6417E-02  0.6179E-04  0.1000E+01     8/   8    i
   8       3    0.1000E+01    -0.1349963012462895E+01   0.2394E-06  0.2028E-03  0.5128E+00  0.4120E-03  0.2106E-04  0.1000E+01     9/   9    i
   9       3    0.1000E+01    -0.1349962886083232E+01   0.8269E-09  0.1035E-04  0.5106E+00  0.4345E-04  0.1275E-06  0.1000E+01    10/  10    i
  10       3    0.1000E+01    -0.1349962885862414E+01   0.3006E-11  0.3058E-06  0.5096E+00  0.2714E-05  0.2240E-09  0.1000E+01    11/  11    i
  11       3    0.1000E+01    -0.1349962885860211E+01   0.2665E-14  0.4422E-08  0.5091E+00  0.5736E-07  0.2205E-11  0.1000E+01    12/  12    i
------------------------------------------------------------------------------------------------------------------------------
KKT CONDITIONS SATISFIED (IER= 0)!
SOLUTION:

OBJ =       -0.1349962885860211E+01
KKT =        0.4421887098724018E-08
CON =        0.2664535259100376E-14
X   =
             0.6094665336054564E-01
             0.5976493035302869E+00
             0.1000000000000000E+01
             0.5976493034306842E+00
             0.6094665324738306E-01
             0.3437714533890817E+00
             0.5000000000868919E+00
            -0.4999999999131094E+00
            -0.3437714530799649E+00
CONSTRAINT    LB                    VALUE                 UB                    STATUS      LAMBDA
   1           0.0000000000000000E+00       0.6094665336054564E-01       0.1000000000000000E+21    IA       0.0000000000000000E+00
   2          -0.1000000000000000E+21       0.5976493035302869E+00       0.1000000000000000E+21    IA       0.0000000000000000E+00
   3          -0.1000000000000000E+01       0.1000000000000000E+01       0.1000000000000000E+01    UB       0.6875429052209211E+00
   4          -0.1000000000000000E+21       0.5976493034306842E+00       0.1000000000000000E+21    IA       0.0000000000000000E+00
   5           0.0000000000000000E+00       0.6094665324738306E-01       0.1000000000000000E+21    IA       0.0000000000000000E+00
   6           0.0000000000000000E+00       0.3437714533890817E+00       0.1000000000000000E+21    IA       0.0000000000000000E+00
   7           0.0000000000000000E+00       0.5000000000868919E+00       0.1000000000000000E+21    IA       0.0000000000000000E+00
   8          -0.1000000000000000E+21      -0.4999999999131094E+00       0.0000000000000000E+00    IA       0.0000000000000000E+00
   9          -0.1000000000000000E+21      -0.3437714530799649E+00       0.0000000000000000E+00    IA       0.0000000000000000E+00
CONSTRAINT    LB                    VALUE                 UB                    STATUS      LAMBDA
  10           0.0000000000000000E+00       0.5367026501697413E+00       0.1000000000000000E+21    IA       0.0000000000000000E+00
  11           0.0000000000000000E+00       0.4023506964697131E+00       0.1000000000000000E+21    IA       0.0000000000000000E+00
  12           0.0000000000000000E+00       0.4023506965693158E+00       0.1000000000000000E+21    IA       0.0000000000000000E+00
  13           0.0000000000000000E+00       0.5367026501833012E+00       0.1000000000000000E+21    IA       0.0000000000000000E+00
  14          -0.1000000000000000E+21       0.1218933067210921E+00       0.1000000000000000E+01    IA       0.0000000000000000E+00
  15          -0.1000000000000000E+21       0.3124570935025335E+00       0.1000000000000000E+01    IA       0.0000000000000000E+00
  16          -0.1000000000000000E+21       0.1000000000000001E+01       0.1000000000000000E+01    UB       0.8318406155170190E-01
  17          -0.1000000000000000E+21       0.1000000000000001E+01       0.1000000000000000E+01    UB       0.3202624887781933E+00
  18          -0.1000000000000000E+21       0.4727152482359042E+00       0.1000000000000000E+01    IA       0.0000000000000000E+00
  19          -0.1000000000000000E+21       0.6071846900971289E+00       0.1000000000000000E+01    IA       0.0000000000000000E+00
  20          -0.1000000000000000E+21       0.4118860830365552E+00       0.1000000000000000E+01    IA       0.0000000000000000E+00
  21          -0.1000000000000000E+21       0.1000000000000003E+01       0.1000000000000000E+01    UB       0.1992983286519636E+00
  22          -0.1000000000000000E+21       0.1000000000000002E+01       0.1000000000000000E+01    UB       0.3202624872860010E+00
  23          -0.1000000000000000E+21       0.4118860829429231E+00       0.1000000000000000E+01    IA       0.0000000000000000E+00
  24          -0.1000000000000000E+21       0.1000000000000001E+01       0.1000000000000000E+01    UB       0.8318406609821495E-01
  25          -0.1000000000000000E+21       0.6071846898042915E+00       0.1000000000000000E+01    IA       0.0000000000000000E+00
  26          -0.1000000000000000E+21       0.3124570935593750E+00       0.1000000000000000E+01    IA       0.0000000000000000E+00
  27          -0.1000000000000000E+21       0.1218933064947673E+00       0.1000000000000000E+01    IA       0.0000000000000000E+00
```

■

Bemerkung 3.2.7

- *Für einen Konvergenzbeweis sei auf Han [Han77] verwiesen. Im günstigsten Fall geht dabei das globale SQP-Verfahren nach endlich vielen Schritten in das lokale über, welches (unter entsprechenden Voraussetzungen) mindestens superlinear konvergiert. Dies bedeutet, dass dann die Schrittweite $\alpha_k = 1$ bei der Armijo-Regel akzeptiert wird.*

  Es gibt jedoch Beispiele bei denen die Schrittweite $\alpha_k = 1$ nicht akzeptiert wird und somit die superlineare Konvergenz verhindert wird. Dieser Effekt geht auf Maratos zurück und heißt daher **Maratos-Effekt**. *Strategien zur Vermeidung des Effekts sind in Geiger und Kanzow [GK02] ab S. 258 beschrieben.*

- *Es gibt auch differenzierbare exakte Bewertungsfunktionen, diese sind allerdings nicht von der Gestalt in (3.6). Eine häufig benutzte differenzierbare exakte Bewertungsfunktion für das Standard-Optimierungsproblem ist die* **erweiterte Lagrange-**

Funktion

$$\begin{aligned} L_a(x,\lambda,\mu;\eta) &= f(x)+\mu^\top h(x)+\frac{\eta}{2}\|h(x)\|^2 \\ &\quad +\frac{1}{2\eta}\sum_{i=1}^{m}\left((\max\{0,\lambda_i+\eta g_i(x)\})^2-\lambda_i^2\right) \\ &= f(x)+\sum_{j=1}^{p}\left(\mu_j h_j(x)+\frac{\eta}{2}h_j(x)^2\right) \\ &\quad +\sum_{i=1}^{m}\begin{cases}\lambda_i g_i(x)+\frac{\eta}{2}g_i(x)^2, & \text{falls } \lambda_i+\eta g_i(x)\geq 0,\\ -\frac{\lambda_i^2}{2\eta}, & \text{sonst.}\end{cases} \end{aligned} \tag{3.9}$$

Ein SQP-Verfahren unter Verwendung der erweiterten Lagrange-Funktion wird in Schittkowsi [Sch81, Sch83] diskutiert.

- *In praktischen Anwendungen wird anstatt eines einzelnen Gewichtungsparameters η jeder Summand der Strafterme in der Bewertungsfunktion individuell gewichtet, etwa durch η_i, $i=1,\dots,m$ und $\hat\eta_j$, $j=1,\dots,p$. Powell [Pow78] schlug folgende Update-Formel vor:*

$$\begin{aligned} \eta_i^{[k+1]} &:= \max\{|\lambda_i^{[k+1]}|,\frac{1}{2}(\eta_i^{[k]}+|\lambda_i^{[k+1]}|)\}, \qquad i=1,\dots,m,\\ \hat\eta_j^{[k+1]} &:= \max\{|\mu_j^{[k+1]}|,\frac{1}{2}(\hat\eta_j^{[k]}+|\mu_j^{[k+1]}|)\}, \qquad j=1,\dots,p. \end{aligned}$$

  Diese Formel hat sich in der Praxis bewährt.

- *Anstatt eine eindimensionale Liniensuche für eine Bewertungsfunktion durchzuführen, kann auch der Trust-Region-Ansatz mit dem SQP-Ansatz gekoppelt werden. Dies führt zu* **Trust-Region-SQP-Verfahren**.

- *Eine aktuelle Entwicklung sind die sogenannten* **Filter-SQP-Verfahren**, *siehe Fletcher und Leyffer [FL02] und Fletcher et al. [FLT02]. Diese ersetzen die Liniensuche für eine Bewertungsfunktion durch ein Verfahren, welches gute Suchrichtungen nach einem geeigneten Kriterien aus einer Menge von möglichen Suchrichtungen „herausfiltert“.*

■

3.3 Inkonsistentes QP Problem

Bisher haben wir stets vorausgesetzt, dass das QP Problem eine Lösung besitzt. Dies muss aber nicht der Fall sein, wie das folgende Beispiel zeigt.

Beispiel 3.3.1

Betrachte die Nebenbedingung

$$g(x)=1-x^2\leq 0$$

und $x^{[0]} = 0$. Im QP Problem erhalten wir die Nebenbedingung

$$g(x^{[0]}) + g'(x^{[0]}) \cdot d = 1 \leq 0.$$

Offensichtlich ist diese niemals erfüllt. ∎

Powell [Pow78] schlug vor, die Nebenbedingungen des QP Problems zu relaxieren, so dass das relaxierte QP Problem zulässig ist. Das ursprüngliche QP Problem wird ersetzt durch ein relaxiertes QP Problem.

Problem 3.3.2 **(Relaxiertes QP Problem $RQP(x^{[k]}, \lambda^{[k]}, \mu^{[k]})$)**

$$\begin{aligned} \min_{d\in\mathbb{R}^n,\ \delta\in[0,1]} \quad & \frac{1}{2} d^\top H_k d + \nabla f(x^{[k]})^\top d + \frac{\eta}{2}\delta^2 \\ \text{unter} \quad & g_i(x^{[k]})(1-\sigma_i\delta) + \nabla g_i(x^{[k]})^\top d \leq 0, \quad i = 1,\dots,m, \\ & h_j(x^{[k]})(1-\delta) + \nabla h_j(x^{[k]})^\top d = 0, \quad j = 1,\dots,p. \end{aligned}$$

Hierin ist

$$\sigma_i = \begin{cases} 0, & \text{falls } g_i(x^{[k]}) < 0, \\ 1, & \text{sonst,} \end{cases} \qquad i = 1,\dots,m.$$

Der Punkt $d = 0$ und $\delta = 1$ ist stets zulässig für $RQP(x^{[k]}, \lambda^{[k]}, \mu^{[k]})$. Erfüllt die optimale Lösung (d, δ) von $RQP(x^{[k]}, \lambda^{[k]}, \mu^{[k]})$ die Beziehung $\delta = 0$, dann ist d auch optimal für das ursprüngliche QP Problem $QP(x^{[k]}, \lambda^{[k]}, \mu^{[k]})$. Um tatsächlich $\delta = 0$ zu erreichen, muss der Gewichtungsparameter η, der auch in der Bewertungsfunktion auftritt, hinreichend groß sein.

3.4 Quadratische Optimierung

Wir widmen uns nun einem Verfahren zur Lösung des wohl einfachsten nichtlinearen Optimierungsproblems – dem quadratischen Optimierungsproblem. Zunächst beschränken wir uns auf den Fall mit Gleichungsrestriktionen.

Problem 3.4.1 **(Quadratisches Optierungsproblem mit Gleichungsbeschränkungen)**

Für eine symmetrische Matrix $W \in \mathbb{R}^{n\times n}$, eine Matrix $B \in \mathbb{R}^{p\times n}$ und Vektoren $c \in \mathbb{R}^n$ und $v \in \mathbb{R}^p$ minimiere

$$f(x) := \frac{1}{2} x^\top W x + c^\top x$$

unter der Nebenbedingung

$$h(x) := Bx - v \;=\; 0.$$

Die KKT-Bedingungen in einem Minimum $\hat{x}$ mit Lagrange-Multiplikator $\mu \in \mathbb{R}^p$ lauten

$$\begin{pmatrix} W & B^\top \\ B & 0 \end{pmatrix} \begin{pmatrix} \hat{x} \\ \mu \end{pmatrix} = \begin{pmatrix} -c \\ v \end{pmatrix}. \tag{3.10}$$

Beachte, dass hier wieder die sogenannte KKT-Matrix auftritt. Ist W positiv definit auf dem Kern von B und $\operatorname{Rang}(B) = p$, so ist die Matrix invertierbar ($\longrightarrow$ Übung). Ist W positiv semidefinit, so ist f konvex und jede Lösung des Gleichungssystems ist zugleich globale Lösung des quadratischen Optimierungsproblems.
Im Hinblick auf ein später zu diskutierendes iteratives Verfahren formen wir (3.10) um und setzen dazu $\hat{x} = x^{[k]} + d$, wobei $x^{[k]}$ ein beliebiger zulässiger Punkt mit $h(x^{[k]}) = 0$ sei.
Dann ist (3.10) äquivalent mit

$$\begin{pmatrix} W & B^\top \\ B & 0 \end{pmatrix} \begin{pmatrix} x^{[k]} + d \\ \mu \end{pmatrix} = \begin{pmatrix} -c \\ v \end{pmatrix},$$

bzw. mit

$$\begin{pmatrix} W & B^\top \\ B & 0 \end{pmatrix} \begin{pmatrix} d \\ \mu \end{pmatrix} = \begin{pmatrix} -c \\ v \end{pmatrix} - \begin{pmatrix} W & B^\top \\ B & 0 \end{pmatrix} \begin{pmatrix} x^{[k]} \\ 0 \end{pmatrix} = \begin{pmatrix} -\nabla f(x^{[k]}) \\ 0 \end{pmatrix}.$$

Diese Betrachtungen zeigen

Satz 3.4.2

Ist $x^{[k]} \in \mathbb{R}^n$ zulässig, so erfüllen $x = x^{[k]} + d$ und $\mu \in \mathbb{R}^p$ die KKT-Bedingungen des gleichungsbeschränkten quadratischen Optimierungsproblems, wenn (d, μ) das lineare Gleichungssystem

$$\begin{pmatrix} W & B^\top \\ B & 0 \end{pmatrix} \begin{pmatrix} d \\ \mu \end{pmatrix} = \begin{pmatrix} -\nabla f(x^{[k]}) \\ 0 \end{pmatrix} \tag{3.11}$$

löst.

Wir lassen nun auch Ungleichungen zu und betrachten allgemeine quadratische Optimierungsprobleme:

Problem 3.4.3 **(Quadratisches Optierungsproblem)**
Für eine symmetrische Matrix $W \in \mathbb{R}^{n\times n}$, Vektoren $c \in \mathbb{R}^n$, $a_i \in \mathbb{R}^n$, $i = 1,\dots,m$, $b_j \in \mathbb{R}^n$, $j = 1,\dots,p$, und Zahlen u_i, $i = 1,\dots,m$, $v_j \in \mathbb{R}$, $j = 1,\dots,p$, minimiere

$$f(x) := \frac{1}{2}x^\top W x + c^\top x$$

unter den Nebenbedingungen

$$\begin{aligned} g_i(x) := a_i^\top x - u_i &\le 0, \qquad i \in \mathcal{I} := \{1,\dots,m\},\\ h_j(x) := b_j^\top x - v_j &= 0, \qquad j \in \mathcal{J} := \{1,\dots,p\}. \end{aligned}$$

Mit

$$A := \begin{pmatrix} a_1^\top \\ \vdots \\ a_m^\top \end{pmatrix}, \quad B := \begin{pmatrix} b_1^\top \\ \vdots \\ b_p^\top \end{pmatrix}, \quad v = \begin{pmatrix} v_1 \\ \vdots \\ v_p \end{pmatrix} \quad u = \begin{pmatrix} u_1 \\ \vdots \\ u_m \end{pmatrix}$$

lautet das Problem in Matrixschreibweise

$$\text{Minimiere} \qquad \frac{1}{2}x^\top W x + c^\top x \qquad \text{u.d.N.} \qquad Ax \le u,\ Bx = v.$$

Auswertung der KKT-Bedingungen in einem lokalen Minimum $\hat{x}$ liefert

$$\begin{aligned} \lambda_i &\ge 0, && i = 1,\dots,m,\\ \lambda_i g_i(\hat{x}) &= 0, && i = 1,\dots,m,\\ W\hat{x} + c + \sum_{i=1}^m \lambda_i a_i + \sum_{j=1}^p \mu_j b_j &= 0, &&\\ a_i^\top \hat{x} - u_i &\le 0, && i = 1,\dots,m,\\ b_j^\top \hat{x} - v_j &= 0, && j = 1,\dots,p. \end{aligned}$$

Dieses System von Gleichungen und Ungleichungen lässt sich nicht so einfach lösen wie im gleichungsbeschränkten Fall. Das Problem ist darin begründet, dass die Indexmenge $\mathcal{A}(\hat{x})$ der aktiven Ungleichungsbeschränkungen unbekannt ist. Wäre sie bekannt, so könnten die inaktiven Ungleichungsbeschränkungen weggelassen werden, da sie keinen Einfluss auf das Optimum haben und man erhielte das äquivalente Problem

$$\begin{aligned} &\text{Minimiere} && \tfrac{1}{2}x^\top W x + c^\top x && && \\ &\text{u.d.N.} && g_i(x) = a_i^\top x - u_i &= 0, && i \in \mathcal{A}(\hat{x}),\\ & && h_j(x) = b_j^\top x - v_j &= 0, && j \in \mathcal{J}. \end{aligned} \tag{3.12}$$

Dieses ist ein quadratisches Optimierungsproblem mit Gleichungsbeschränkungen, dessen Lösung durch Satz 3.4.2 charakterisiert ist.
Die Idee der **Strategie der aktiven Mengen** zur Lösung des allgemeinen quadratischen Optimierungsproblems besteht nun darin, die unbekannte Indexmenge $\mathcal{A}(\hat{x})$ in (3.12) durch eine Schätzung $\mathcal{I}_s \subseteq \mathcal{I}$ zu ersetzen und diese iterativ anzupassen.
Sei $x^{[k]}$ der aktuelle Iterationspunkt, $x^{[k]}$ sei zulässig und $\mathcal{I}_s^k \subseteq \mathcal{I}$ sei die aktuelle Schätzung der aktiven Menge. Löse dann das Hilfsproblem

Problem 3.4.4 **(Hilfsproblem)**
Minimiere

$$f(x^{[k]} + d)$$

bzgl. $d \in \mathbb{R}^n$ *unter den Nebenbedingungen*

$$\begin{aligned} a_i^\top d &= 0, \qquad i \in \mathcal{I}_s^k, \\ b_j^\top d &= 0, \qquad j \in \mathcal{J}. \end{aligned}$$

Die Strategie zur Anpassung der Indexmenge $\mathcal{I}_s^k$ hängt nun ab von der Lösung d und den zugehörigen Lagrange-Multiplikatoren λ_i, $i \in \mathcal{I}_s^k$, und μ_j, $j = 1, \ldots, m$, des Hilfsproblems. Folgende Fälle können eintreten:

(a) Besitzt das Hilfsproblem die Lösung $d = 0$, so liefern die KKT-Bedingungen für das Hilfssproblem

$$\nabla f(x^{[k]}) + \sum_{i \in \mathcal{I}_s^k} \lambda_i a_i + \sum_{j=1}^{m} \mu_j b_j = 0.$$

(i) Sind alle $\lambda_i \geq 0$, $i \in \mathcal{I}_s^k$, so wird $x^{[k]}$ als Lösung akzeptiert, da die KKT-Bedingungen für das Ausgangsproblem erfüllt sind, wenn man noch $\lambda_i = 0$, $i \in \mathcal{I} \setminus \mathcal{I}_s^k$ setzt.

(ii) **Deaktivierungsschritt:**
Gibt es einen Index $i \in \mathcal{I}_s^k$ mit $\lambda_i < 0$, so erfüllt $x^{[k]}$ die KKT-Bedingungen des Ausgangsproblems nicht, ist also nicht optimal. Andererseits ist $x^{[k]}$ Minimum von f unter den Nebenbedingungen $\mathcal{I}_s^k \cup \mathcal{J}$. Daher muss der zulässige Bereich vergrößert werden, d.h. die Indexmenge $\mathcal{I}_s^k$ wird verkleinert.
Bestimme dazu denjenigen Index $q \in \mathcal{I}_s^k$ mit

$$\lambda_q = \min_{i \in \mathcal{I}_s^k} \lambda_i < 0$$

und setze

$$\mathcal{I}_s^{k+1} := \mathcal{I}_s^k \setminus \{q\}.$$

(b) Besitzt das Hilfsproblem eine Lösung $d \neq 0$, so ist d eine Abstiegsrichtung von f im Punkt $x^{[k]}$, die die Nebenbedingungen $\mathcal{I}_s^k \cup \mathcal{J}$ erfüllt, d.h. es gilt

$$a_i^\top d = 0, \quad i \in \mathcal{I}_s^k, \qquad b_j^\top d = 0, \quad j \in \mathcal{J}. \tag{3.13}$$

(i) Ist $x^{[k]} + d$ zulässig für das Ausgangsproblem, d.h. gilt

$$a_i^\top (x^{[k]} + d) \leq u_i, \qquad i \in \mathcal{I} \setminus \mathcal{I}_s^k,$$

so setze

$$x^{[k+1]} := x^{[k]} + d, \qquad \mathcal{I}_s^{k+1} := \mathcal{I}_s^k.$$

(ii) **Aktivierungsschritt:**
Ist $x^{[k]} + d$ unzulässig für das Ausgangsproblem, so bestimme eine möglichst große Schrittweite $\alpha_k \geq 0$, so dass $x^{[k]} + \alpha_k d$ zulässig bleibt:

$$a_i^\top (x^{[k]} + \alpha_k d) \leq u_i \qquad \forall i \in \mathcal{I} \setminus \mathcal{I}_s^k$$

bzw.

$$\alpha_k \left(a_i^\top d\right) \leq u_i - a_i^\top x^{[k]} \qquad \forall i \in \mathcal{I} \setminus \mathcal{I}_s^k \tag{3.14}$$

Beachte, dass $x^{[k]} + \alpha d$ für alle α zulässig bleibt für die Nebenbedingungen $\mathcal{I}_s^k \cup \mathcal{J}$, da $x^{[k]}$ zulässig ist und (3.13) gilt.
Da $x^{[k]}$ zulässig ist und $x^{[k]} + \alpha d$ mit $\alpha = 1$ unzulässig ist, muß es in (3.14) einen Index $i \in \mathcal{I} \setminus \mathcal{I}_s^k$ mit $a_i^\top d > 0$ geben. Bestimme also

$$\alpha_k := \min \left\{ \frac{u_i - a_i^\top x^{[k]}}{a_i^\top d} \;\middle|\; i \in \mathcal{I} \setminus \mathcal{I}_s^k,\ a_i^\top d > 0 \right\}.$$

Sei $r \in \mathcal{I} \setminus \mathcal{I}_s^k$ ein (nicht notwendig eindeutiger) Index, für den dieses Minimum angenommen wird. Der Fall $\alpha = 0$ kann auftreten, wenn mehrere Nebenbedingungen gleichzeitig aktiv werden (Entartung).
Setze

$$x^{[k+1]} := x^{[k]} + \alpha_k d, \qquad \mathcal{I}_s^{k+1} := \mathcal{I}_s^k \cup \{r\}.$$

Zusammenfassend erhalten wir den folgenden Algorithmus.

Algorithmus 3.4.5 (Strategie der aktiven Menge)

(i) *Sei $x^{[0]}$ zulässig für das quadratische Optimierungsproblem. Setze $k := 0$ und*

$$\mathcal{I}_s^0 := \{i \in \mathcal{I} \mid a_i^\top x^{[0]} = u_i\}.$$

(ii) *Bestimme eine Lösung $(d, \lambda_{\mathcal{I}_s^k}, \mu)$ des Hilfsproblems durch Lösen des Gleichungssystems*

$$\begin{pmatrix} W & A_{\mathcal{I}_s^k}^\top & B^\top \\ A_{\mathcal{I}_s^k} & 0 & 0 \\ B & 0 & 0 \end{pmatrix} \begin{pmatrix} d \\ \lambda_{\mathcal{I}_s^k} \\ \mu \end{pmatrix} = \begin{pmatrix} -\nabla f(x^{[k]}) \\ 0 \\ 0 \end{pmatrix}$$

(vgl. (3.11)). Hierin sind

$$A_{\mathcal{I}_s^k} := \left(a_i^\top\right)_{i \in \mathcal{I}_s^k}, \qquad \lambda_{\mathcal{I}_s^k} := (\lambda_i)_{i \in \mathcal{I}_s^k}.$$

(iii) *Ist $d = 0$ und $\lambda_{\mathcal{I}_s^k} \geq 0$, so setze $\lambda_i = 0$ für $i \in \mathcal{I} \setminus \mathcal{I}_s^k$ und STOP.*

(iv) *Ist $d = 0$ und $\lambda_q := \min\{\lambda_i \mid i \in \mathcal{I}_s^k\} < 0$, so setze*

$$\mathcal{I}_s^{k+1} := \mathcal{I}_s^k \setminus \{q\}.$$

Setze $k := k + 1$ und gehe zu (ii).

(v) *Gilt $a_i^\top (x^{[k]} + d) \leq u_i$, $i \in \mathcal{I} \setminus \mathcal{I}_s^k$, so setze*

$$x^{[k+1]} := x^{[k]} + d, \qquad \mathcal{I}_s^{k+1} := \mathcal{I}_s^k.$$

Setze $k := k + 1$ und gehe zu (ii).

(vi) *Bestimme $r \in \mathcal{I} \setminus \mathcal{I}_s^k$ mit*

$$\alpha_k := \frac{u_r - a_r^\top x^{[k]}}{a_r^\top d} = \min\left\{ \frac{u_i - a_i^\top x^{[k]}}{a_i^\top d} \;\middle|\; i \in \mathcal{I} \setminus \mathcal{I}_s^k,\ a_i^\top d > 0 \right\}$$

und setze

$$x^{[k+1]} := x^{[k]} + \alpha_k d, \qquad \mathcal{I}_s^{k+1} := \mathcal{I}_s^k \cup \{r\}.$$

Setze $k := k + 1$ und gehe zu (ii).

Bemerkung 3.4.6

- *Ein zulässiger Startpunkt $x^{[0]}$ für das Active-Set-Verfahren kann analog zu Phase I des Simplexverfahrens berechnet werden.*
- *Ist W positiv definit und sind die Vektoren a_i, $i \in \mathcal{I}_s^0$, und b_j, $j \in \mathcal{J}$, linear unabhängig, so ist der Algorithmus wohldefiniert und endet nach endlich vielen Schritten mit der eindeutig bestimmten Lösung.*
- *Es gibt alternative Verfahren zur Lösung quadratischer Optimierungsprobleme:*
  - *Das Verfahren von Goldfarb-Idnani [GI83] für streng konvexe quadratische Optimierungsprobleme ist eine Active-Set-Strategie für das duale Problem. Es hat den Vorteil, dass kein zulässiger Startpunkt berechnet werden muß.*
  - *Der simplexartige Algorithmus von Lemke (vgl. Cottle et al. [CPS92]) zur Lösung linearer Komplementaritätsprobleme kann ebenfalls zur Lösung quadratischer Optimierungsprobleme verwendet werden. Zu beachten ist hierbei, dass die KKT-Bedingungen des quadratischen Optimierungsproblems ein lineares Komplementaritätsproblem darstellen.*
  - *Innere-Punkt-Verfahren (vgl. Vanderbei [Van01]) stellen eine weitere alternative zur Lösung quadratischer Optimierungsprobleme dar.*
  - *Gill et al. [GM78, GMSW91] beschreiben ein Verfahren, welches auch nichtkonvexe quadratische Probleme lösen kann.*

Beispiel 3.4.7

Das folgende Beispiel bezieht sich auf die etwas allgemeinere Aufgabenstellung

$$\min \frac{1}{2} x^\top W x + c^\top x \quad \textit{unter} \quad l \leq \begin{pmatrix} x \\ Ax \end{pmatrix} \leq u,$$

$W \in \mathbb{R}^{n\times n}$, $c, x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, $l, u \in \mathbb{R}^{n+m}$.
Spezielle Daten:

$$W = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad c = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \quad A = \begin{pmatrix} 1 & 1 \\ -1 & 1 \\ 1 & 0 \end{pmatrix}, \quad l = \begin{pmatrix} -\infty \\ -1 \\ 0 \\ -\infty \\ -\infty \end{pmatrix}, u = \begin{pmatrix} 5 \\ 2 \\ 5 \\ 2 \\ 5 \end{pmatrix}.$$

Startschätzung: $x^{[0]} = (5, 0)^\top$.
Resultat des Algorithmus:

```
 ------------------------------------------------------
INITIAL X IS FEASIBLE!
STARTING OPTIMIZATION PHASE ...
------------------------------------------------------
------------------------------------------------------
                  ITERATION  0
------------------------------------------------------
OBJ =        0.22500000000000000E+02
KKT =        0.70000000000000000E+01
CON =        0.00000000000000000E+00
VARIABLE     LB                        VALUE                     UB                        STATUS      LAMBDA
 1          -0.10000000000000000E+21    0.50000000000000000E+01   0.50000000000000000E+01   UB          0.00000000000000000E+00
 2          -0.10000000000000000E+01    0.00000000000000000E+00   0.20000000000000000E+01   IA          0.00000000000000000E+00
CONSTRAINT   LB                        VALUE                     UB                        STATUS      LAMBDA
 3           0.00000000000000000E+00    0.50000000000000000E+01   0.50000000000000000E+01   UB          0.00000000000000000E+00
 4          -0.10000000000000000E+21   -0.50000000000000000E+01   0.20000000000000000E+01   IA          0.00000000000000000E+00
 5          -0.10000000000000000E+21    0.50000000000000000E+01   0.50000000000000000E+01   UB          0.00000000000000000E+00
DEACTIVATING LINEAR DEPENDENT CONSTRAINT  5
DEACTIVATING CONSTRAINT  1 WITH MULTIPLIER VALUE LAMBDA =  -6.
------------------------------------------------------
                  ITERATION  1
------------------------------------------------------
OBJ =        0.22500000000000000E+02
KKT =        0.60000000000000000E+01
CON =        0.00000000000000000E+00
VARIABLE     LB                        VALUE                     UB                        STATUS      LAMBDA
 1          -0.10000000000000000E+21    0.50000000000000000E+01   0.50000000000000000E+01   IA          0.00000000000000000E+00
 2          -0.10000000000000000E+01    0.00000000000000000E+00   0.20000000000000000E+01   IA          0.00000000000000000E+00
CONSTRAINT   LB                        VALUE                     UB                        STATUS      LAMBDA
 3           0.00000000000000000E+00    0.50000000000000000E+01   0.50000000000000000E+01   UB         -0.10000000000000000E+01
 4          -0.10000000000000000E+21   -0.50000000000000000E+01   0.20000000000000000E+01   IA          0.00000000000000000E+00
 5          -0.10000000000000000E+21    0.50000000000000000E+01   0.50000000000000000E+01   IA          0.00000000000000000E+00
ACTIVATING CONSTRAINT  2 OF TYPE UB
------------------------------------------------------
                  ITERATION  2
------------------------------------------------------
OBJ =        0.14500000000000000E+02
KKT =        0.10000000000000000E+01
CON =        0.00000000000000000E+00
VARIABLE     LB                        VALUE                     UB                        STATUS      LAMBDA
 1          -0.10000000000000000E+21    0.30000000000000000E+01   0.50000000000000000E+01   IA          0.00000000000000000E+00
 2          -0.10000000000000000E+01    0.20000000000000000E+01   0.20000000000000000E+01   UB          0.00000000000000000E+00
CONSTRAINT   LB                        VALUE                     UB                        STATUS      LAMBDA
 3           0.00000000000000000E+00    0.50000000000000000E+01   0.50000000000000000E+01   UB         -0.40000000000000000E+01
 4          -0.10000000000000000E+21   -0.10000000000000000E+01   0.20000000000000000E+01   IA          0.00000000000000000E+00
 5          -0.10000000000000000E+21    0.30000000000000000E+01   0.50000000000000000E+01   IA          0.00000000000000000E+00
DEACTIVATING CONSTRAINT  3 WITH MULTIPLIER VALUE LAMBDA =  -5.
------------------------------------------------------
                  ITERATION  3
------------------------------------------------------
OBJ =        0.14500000000000000E+02
KKT =        0.50000000000000000E+01
CON =        0.00000000000000000E+00
VARIABLE     LB                        VALUE                     UB                        STATUS      LAMBDA
 1          -0.10000000000000000E+21    0.30000000000000000E+01   0.50000000000000000E+01   IA          0.00000000000000000E+00
 2          -0.10000000000000000E+01    0.20000000000000000E+01   0.20000000000000000E+01   UB          0.20000000000000000E+01
CONSTRAINT   LB                        VALUE                     UB                        STATUS      LAMBDA
 3           0.00000000000000000E+00    0.50000000000000000E+01   0.50000000000000000E+01   IA          0.00000000000000000E+00
 4          -0.10000000000000000E+21   -0.10000000000000000E+01   0.20000000000000000E+01   IA          0.00000000000000000E+00
 5          -0.10000000000000000E+21    0.30000000000000000E+01   0.50000000000000000E+01   IA          0.00000000000000000E+00
ACTIVATING CONSTRAINT  4 OF TYPE UB
------------------------------------------------------
                  ITERATION  4
------------------------------------------------------
OBJ =        0.40000000000000000E+01
KKT =        0.20000000000000000E+01
CON =        0.00000000000000000E+00
VARIABLE     LB                        VALUE                     UB                        STATUS      LAMBDA
 1          -0.10000000000000000E+21    0.00000000000000000E+00   0.50000000000000000E+01   IA          0.00000000000000000E+00
 2          -0.10000000000000000E+01    0.20000000000000000E+01   0.20000000000000000E+01   UB         -0.30000000000000000E+01
CONSTRAINT   LB                        VALUE                     UB                        STATUS      LAMBDA
 3           0.00000000000000000E+00    0.20000000000000000E+01   0.50000000000000000E+01   IA          0.00000000000000000E+00
 4          -0.10000000000000000E+21    0.20000000000000000E+01   0.20000000000000000E+01   UB          0.00000000000000000E+00
 5          -0.10000000000000000E+21    0.00000000000000000E+00   0.50000000000000000E+01   IA          0.00000000000000000E+00
DEACTIVATING CONSTRAINT  2 WITH MULTIPLIER VALUE LAMBDA =  -5.
------------------------------------------------------
                  ITERATION  5
------------------------------------------------------
OBJ =        0.40000000000000000E+01
KKT =        0.50000000000000000E+01
CON =        0.00000000000000000E+00
VARIABLE     LB                        VALUE                     UB                        STATUS      LAMBDA
 1          -0.10000000000000000E+21    0.00000000000000000E+00   0.50000000000000000E+01   IA          0.00000000000000000E+00
 2          -0.10000000000000000E+01    0.20000000000000000E+01   0.20000000000000000E+01   IA          0.00000000000000000E+00
CONSTRAINT   LB                        VALUE                     UB                        STATUS      LAMBDA
 3           0.00000000000000000E+00    0.20000000000000000E+01   0.50000000000000000E+01   IA          0.00000000000000000E+00
 4          -0.10000000000000000E+21    0.20000000000000000E+01   0.20000000000000000E+01   UB          0.20000000000000000E+01
 5          -0.10000000000000000E+21    0.00000000000000000E+00   0.50000000000000000E+01   IA          0.00000000000000000E+00
ACTIVATING CONSTRAINT  3 OF TYPE LB
------------------------------------------------------
                  ITERATION  6
------------------------------------------------------
OBJ =        0.66613381477509390E-15
KKT =        0.15000000000000000E+01
CON =        0.00000000000000000E+00
VARIABLE     LB                        VALUE                     UB                        STATUS      LAMBDA
 1          -0.10000000000000000E+21   -0.99999999999999980E+00   0.50000000000000000E+01   IA          0.00000000000000000E+00
 2          -0.10000000000000000E+01    0.10000000000000000E+01   0.20000000000000000E+01   IA          0.00000000000000000E+00
CONSTRAINT   LB                        VALUE                     UB                        STATUS      LAMBDA
 3           0.00000000000000000E+00    0.00000000000000000E+00   0.50000000000000000E+01   LB          0.00000000000000000E+00
 4          -0.10000000000000000E+21    0.20000000000000000E+01   0.20000000000000000E+01   UB         -0.49999999999999980E+00
 5          -0.10000000000000000E+21   -0.99999999999999980E+00   0.50000000000000000E+01   IA          0.00000000000000000E+00
DEACTIVATING CONSTRAINT  4 WITH MULTIPLIER VALUE LAMBDA =  -0.5
------------------------------------------------------
                  ITERATION  7
------------------------------------------------------
OBJ =        0.66613381477509390E-15
KKT =        0.50000000000000000E+00
CON =        0.00000000000000000E+00
VARIABLE     LB                        VALUE                     UB                        STATUS      LAMBDA
 1          -0.10000000000000000E+21   -0.99999999999999980E+00   0.50000000000000000E+01   IA          0.00000000000000000E+00
 2          -0.10000000000000000E+01    0.10000000000000000E+01   0.20000000000000000E+01   IA          0.00000000000000000E+00
CONSTRAINT   LB                        VALUE                     UB                        STATUS      LAMBDA
 3           0.00000000000000000E+00    0.00000000000000000E+00   0.50000000000000000E+01   LB         -0.15000000000000000E+01
 4          -0.10000000000000000E+21    0.20000000000000000E+01   0.20000000000000000E+01   IA          0.00000000000000000E+00
 5          -0.10000000000000000E+21   -0.99999999999999980E+00   0.50000000000000000E+01   IA          0.00000000000000000E+00
```

```
-------------------------------------------------------
                  ITERATION  8
-------------------------------------------------------
OBJ =       -0.2499999999999994E+00
KKT =        0.2220446049250313E-15
CON =        0.1110223024625157E-15
VARIABLE     LB                        VALUE                     UB                      STATUS      LAMBDA
 1          -0.1000000000000000E+21     -0.4999999999999999E+00    0.5000000000000000E+01   IA        0.0000000000000000E+00
 2          -0.1000000000000000E+01      0.5000000000000002E+00    0.2000000000000000E+01   IA        0.0000000000000000E+00
CONSTRAINT   LB                        VALUE                     UB                      STATUS      LAMBDA
 3           0.0000000000000000E+00     -0.1110223024625157E-15    0.5000000000000000E+01   LB       -0.1500000000000000E+01
 4          -0.1000000000000000E+21      0.1000000000000000E+01    0.2000000000000000E+01   IA        0.0000000000000000E+00
 5          -0.1000000000000000E+21     -0.4999999999999999E+00    0.5000000000000000E+01   IA        0.0000000000000000E+00

OPTIMAL SOLUTION FOUND AT ITERATION  8
```

Kapitel 4

Innere-Punkte-Verfahren und Barrieremethoden

Innere-Punkt-Verfahren basieren auf der Konstruktion von Näherungslösungen, die sich strikt im Inneren des zulässigen Bereichs Σ befinden und sich aus dem Inneren dem Optimum (welches in der Regel am Rand liegt) nähern. Erreicht wird dies durch Ankopplung von Straftermen, die zulässige Punkte auf dem Rand bestrafen (anders als bei Penalty-Verfahren, wo nur unzulässige Punkte bestraft werden!), vgl. Abbildung 4.1.

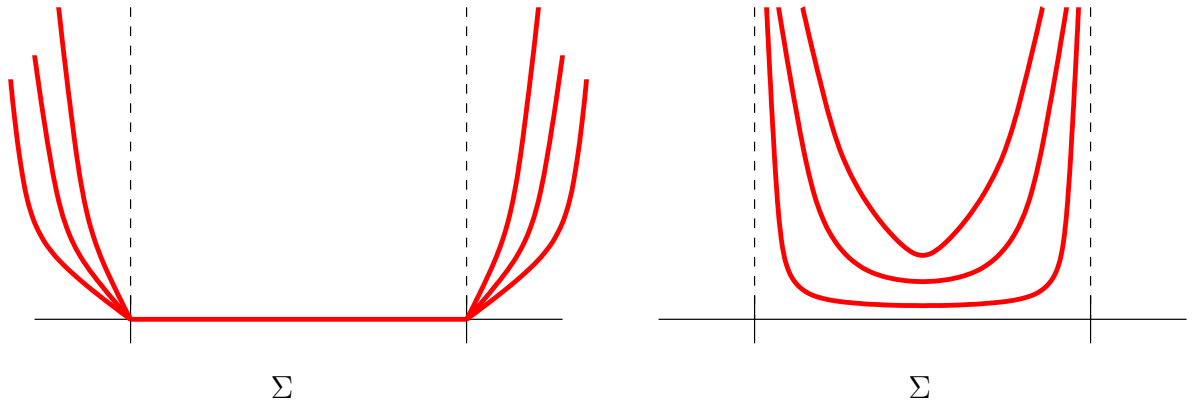


Abbildung 4.1: Qualitativer Unterschied zwischen Penalty-Verfahren (links) und Barriere-Verfahren (rechts). Dargestellt sind die Werte der Penalty-Funktion bzw. der Barriere-Funktion innerhalb und außerhalb des zulässigen Bereichs Σ.

Derartige Ansätze wurden bereits früh von Fiacco und McCormick [FM90] diskutiert, wurden zwischenzeitlich nicht sonderlich beachtet und haben mittlerweile eine Renaissance erfahren. Insbesondere konnte gezeigt werden, dass bestimmte Innere-Punkte-Verfahren – im Gegensatz zum Simplexverfahren – lineare Optimierungsprobleme mit polynomialem Aufwand in Abhängigkeit der Problemdimensionen lösen können, vgl. Satz 4.1.7.

4.1 Lineare Optimierungsprobleme

In der Einführungsvorlesung zur Optimierung haben wir das Simplexverfahren zur Lösung von linearen Optimierungsproblemen kennen gelernt. Die Idee des Simplexverfahren ba-

sierte auf der sukzessiven Berechnung von zulässigen Basislösungen (Ecken), d.h. das Simplexverfahren wählt stets Kanten am Rand des zulässigen Bereichs als Abstiegsrichtungen und wandert von einer Ecke zur nächsten ($\longrightarrow$ Basiswechsel). Innere-Punkte-Verfahren verfolgen eine gänzlich andere Idee und erzeugen Iterierte im Inneren des zulässigen Bereichs.
Wir starten mit dem primalen linearen Optimierungsproblem in Normalform:

Problem 4.1.1 (Primale Normalform)

$$\textit{Minimiere} \qquad c^\top x \qquad \textit{u.d.N.} \quad Ax = b,\ x \geq 0. \tag{4.1}$$

Gegebene Daten: $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$.

Mit der Lagrangefunktion

$$L(x, \lambda, \mu) = c^\top x + \lambda^\top(-x) + \mu^\top(b - Ax)$$

lauten die KKT-Bedingungen

$$\begin{aligned} A^\top\mu + \lambda &= c, && (4.2)\\ Ax &= b, && (4.3)\\ x &\geq 0, && (4.4)\\ \lambda &\geq 0, && (4.5)\\ \lambda_i x_i &= 0, \quad i = 1, \dots, n. && (4.6) \end{aligned}$$

Nun eliminieren wir die Ungleichungen $x \geq 0$ im primalen Problem, indem wir diese durch Strafterme an die Zielfunktion koppeln und erhalten für $\eta > 0$ das folgende Problem:

Problem 4.1.2 ((logarithmisches) Barriere-Problem)

$$\textit{Minimiere} \qquad c^\top x - \eta \sum_{i=1}^{n} \log(x_i) \qquad \textit{u.d.N.} \qquad Ax = b. \tag{4.7}$$

Beachte, dass $\log(x_i) \to -\infty$ für $x_i \downarrow 0$. Daher erzeugt der Term $-\eta \log(x_i)$ bei $x_i = 0$ eine Barriere mit Wert ∞, so dass das Minimum stets für $x_i \neq 0$ angenommen wird. Ziel wird es im Folgenden sein, den Gewichtungsparameter η iterativ anzupassen, so dass eine Folge von zulässigen Lösungen mit $x > 0$ generiert wird, die gegen das gesuchte Minimum von (4.1) konvergiert.

Das Barriere-Problem (4.7) ist bedingt durch die Logarithmusterme ein nichtlineares konvexes Optimierungsproblem. Die KKT-Bedingungen lauten

$$\begin{aligned} c_i - \frac{\eta}{x_i} - (A^\top \mu)_i &= 0, \qquad i = 1, \dots, n, \\ Ax &= b. \end{aligned}$$

Mit den Definitionen $\lambda_i := \frac{\eta}{x_i}$, $i = 1, \dots, n$, $\lambda := (\lambda_1, \dots, \lambda_n)^\top$ lauten sie

$$A^\top \mu + \lambda = c, \tag{4.8}$$

$$Ax = b, \tag{4.9}$$

$$\lambda_i x_i = \eta, \quad i = 1, \dots, n. \tag{4.10}$$

Ein Vergleich mit (4.2)-(4.6) zeigt, dass die KKT-Bedingungen (4.8)- (4.10) für das Barriere-Problem als **Störung** der KKT-Bedingungen (4.2)-(4.6) interpretiert werden können, wenn noch $x > 0$ und $\lambda > 0$ gilt. Die Störung tritt explizit durch den Gewichtungsparameter $\eta > 0$ in der Komplementaritätsbedingung (4.6) bzw. (4.10) auf.
Besitzt das nichtlineare Gleichungssystem (4.8)-(4.10) für jedes $\eta > 0$ eine Lösung

$$(x(\eta), \lambda(\eta), \mu(\eta)),$$

so besteht die Hoffnung, dass diese für $\eta \downarrow 0$ gegen eine Lösung des Ausgangsproblems (4.1) konvergiert. Die Menge

$$\{(x(\eta), \lambda(\eta), \mu(\eta)) \mid \eta > 0\}$$

heißt **zentraler Pfad**.
Da die KKT-Bedingungen für das konvexe Barriere-Problem notwendig[1] und hinreichend sind und mit den zentralen Pfadbedingungen (4.8)-(4.9) übereinstimmen, gilt folgender Hilfssatz.

Hilfsatz 4.1.3

Sei $\eta > 0$. Das Barriere-Problem besitzt genau dann eine Lösung $x > 0$, wenn die zentralen Pfadbedingungen (4.8)-(4.9) eine Lösung (x, μ, λ) mit $x > 0$, $\lambda > 0$ besitzen. ∎

Es stellt sich die Frage, wann die zentralen Pfadbedingungen eine Lösung $x > 0$, $\lambda > 0$ besitzen. Offenbar besitzen sie keine Lösung, wenn die Menge

$$\mathcal{F}^\circ := \{(x, \mu, \lambda) \mid A^\top \mu + \lambda = c,\ Ax = b,\ x > 0,\ \lambda > 0\}$$

[1] Die KKT-Bedingungen gelten i.A. nur, falls eine Constraint Qualification erfüllt ist. Für lineare Nebenbedingungen kann man aber zeigen, dass die sogenannte Abadie-Constraint Qualification automatisch erfüllt ist. Diese besagt, dass der Tangentialkegel gleich dem linearisierenden Kegel ist, was für lineare Nebenbedingungen stets der Fall ist.

leer ist. Andernfalls besitzen sie eine Lösung:

Satz 4.1.4

Sei $\mathcal{F}^\circ \neq \emptyset$. Dann besitzt das Barriere-Problem für jedes $\eta > 0$ eine Lösung $x > 0$ (wegen Hilfssatz 4.1.3 besitzen dann auch die zentralen Pfadbedingungen eine Lösung).

Beweis: Seien $\eta > 0$ und $(\hat{x}, \hat{\mu}, \hat{\lambda}) \in \mathcal{F}^\circ$. Dann gilt $A^\top \hat{\mu} + \hat{\lambda} = c, A\hat{x} = b, \hat{x} > 0, \hat{\lambda} > 0$. Betrachte

$$\phi_\eta(x) := c^\top x - \eta \sum_{i=1}^n \log(x_i).$$

Wir zeigen, dass die Menge

$$L_\eta := \{x \in \mathbb{R}^n \mid Ax = b,\ x \geq 0,\ \phi_\eta(x) \leq \phi_\eta(\hat{x})\}$$

kompakt ist. Da sie offenbar abgeschlossen ist, müssen wir nur noch die Beschränktheit zeigen. Sei nun $x \in L_\eta$. Dann gilt

$$\begin{aligned}
\phi_\eta(x) &= c^\top x - \eta \sum_{i=1}^n \log(x_i) \\
&= c^\top x - \hat{\mu}^\top (Ax - b) - \eta \sum_{i=1}^n \log(x_i) \\
&= c^\top x - x^\top A^\top \hat{\mu} + b^\top \hat{\mu} - \eta \sum_{i=1}^n \log(x_i) \\
&= c^\top x - x^\top (c - \hat{\lambda}) + b^\top \hat{\mu} - \eta \sum_{i=1}^n \log(x_i) \\
&= x^\top \hat{\lambda} + b^\top \hat{\mu} - \eta \sum_{i=1}^n \log(x_i).
\end{aligned}$$

Damit ist $\phi_\eta(x) \leq \phi_\eta(\hat{x})$ äquivalent mit

$$x^\top \hat{\lambda} + b^\top \hat{\mu} - \eta \sum_{i=1}^n \log(x_i) \leq \phi_\eta(\hat{x})$$

bzw.

$$\sum_{i=1}^n \Big(\hat{\lambda}_i x_i - \eta \log(x_i) \Big) \leq \phi_\eta(\hat{x}) - b^\top \hat{\mu} =: \kappa.$$

Die Funktion $x_i \mapsto p(x_i) := \hat{\lambda}_i x_i - \eta \log(x_i)$ hat die Eigenschaften $p(x_i) \to \infty$ für $x_i \to 0$ oder $x_i \to \infty$. Damit ist die Menge $\{x \in \mathbb{R}^n \mid \phi_\eta(x) \leq \phi_\eta(\hat{x})\}$ beschränkt und es folgt automatisch $x > 0$. Also ist auch die Menge L_η beschränkt und somit kompakt. Folglich nimmt die stetige Funktion ϕ_η auf der kompakten Menge L_η ihr Minimum an. ∎

Zur numerischen Lösung der nichtlinearen Gleichungen (4.8)-(4.10) wird das Newton-Verfahren auf die Funktion

$$F_\eta(x,\mu,\lambda) := \begin{pmatrix} A^\top\mu + \lambda - c \\ Ax - b \\ X\Lambda e - \eta e \end{pmatrix}$$

angewendet. Hierbei benutzen wir die Notationen

$$X = diag(x_1,\ldots,x_n), \quad \Lambda = diag(\lambda_1,\ldots,\lambda_n), \quad e = (1,\ldots,1)^\top.$$

Die Jacobimatrix von F_η lautet

$$F'_\eta(x,\mu,\lambda) = \begin{pmatrix} 0 & A^\top & I \\ A & 0 & 0 \\ \Lambda & 0 & X \end{pmatrix}$$

Es gilt

Hilfsatz 4.1.5

Sei $(x,\mu,\lambda) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^n$ ein Vektor mit $x > 0$ und $\lambda > 0$ und es gelte $Rang(A) = m$. Dann ist die Jacobimatrix $F'_\eta(x,\mu,\lambda)$ für jedes $\eta > 0$ invertierbar.

Beweis: Der Beweis verläuft analog zum Beweis zur Regularität der KKT-Matrix ($\longrightarrow$ Übung). ∎

Sei $w^{[k]} := (x^{[k]},\mu^{[k]},\lambda^{[k]})$ ein gegebener Iterationspunkt. Die Newtonkorrektur Δw ist durch das lineare Gleichungssystem

$$F'_\eta(w^{[k]})\Delta w = -F_\eta(w^{[k]})$$

bzw. durch

$$\begin{pmatrix} 0 & A^\top & I \\ A & 0 & 0 \\ \Lambda^{[k]} & 0 & X^{[k]} \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta\mu \\ \Delta\lambda \end{pmatrix} = - \begin{pmatrix} A^\top\mu^{[k]} + \lambda^{[k]} - c \\ Ax^{[k]} - b \\ X^{[k]}\Lambda^{[k]}e - \eta_k e \end{pmatrix} \tag{4.11}$$

gegeben, wobei $X^{[k]} = diag(x_1^{[k]},\ldots,x_n^{[k]})$ und $\Lambda^{[k]} = diag(\lambda_1^{[k]},\ldots,\lambda_n^{[k]})$ sind. Das gedämpfte Newtonverfahren liefert die neue Iterierte

$$w^{[k+1]} := w^{[k]} + t_k\Delta w$$

mit einer Schrittweite $t_k > 0$.

Wir betrachten noch den Spezialfall eines **zulässigen Verfahrens**, welches dadurch ausgezeichnet ist, dass es die Bedingungen

$$\begin{aligned} A^\top \mu^{[k]} + \lambda^{[k]} - c &= 0, \\ Ax^{[k]} - b &= 0 \end{aligned}$$

erfüllt. Aus den ersten beiden Gleichungen in (4.11) folgen dann

$$\begin{aligned} A^\top \Delta\mu + \Delta\lambda &= 0, \\ A\Delta x &= 0. \end{aligned}$$

Für die neue Iterierte ergibt sich damit

$$\begin{aligned} A^\top \mu^{[k+1]} + \lambda^{[k+1]} - c &= A^\top \left(\mu^{[k]} + t_k \Delta\mu\right) + \lambda^{[k]} + t_k \Delta\lambda - c \\ &= A^\top \mu^{[k]} + \lambda^{[k]} - c \\ &= 0, \\ Ax^{[k+1]} - b &= A\left(x^{[k]} + t_k \Delta x\right) - b \\ &= Ax^{[k]} - b \\ &= 0. \end{aligned}$$

Mit anderen Worten: Erfüllt $(x^{[0]}, \mu^{[0]}, \lambda^{[0]})$ die Bedingungen (4.8)-(4.9), so erfüllen alle weiteren Iterierten diese Bedingungen ebenfalls.
Zusammenfassend erhalten wir das folgende (konzeptionelle) Innere-Punkt-Verfahren:

Algorithmus 4.1.6 **((zulässiges) Innere-Punkt-Verfahren)**

(i) Wähle $w^{[0]} := (x^{[0]}, \mu^{[0]}, \lambda^{[0]})$ *mit*

$$Ax^{[0]} = b, \qquad A^\top \mu^{[0]} + \lambda^{[0]} = c, \qquad x^{[0]} > 0, \qquad \lambda^{[0]} > 0,$$

$\varepsilon > 0$ *und setze* $k = 0$.

(ii) Ist $\zeta_k := \frac{(x^{[k]})^\top \lambda^{[k]}}{n} \leq \varepsilon$, *STOP.*

(iii) Wähle $\sigma_k \in [0,1]$ *und berechne* $\Delta w = (\Delta x, \Delta\mu, \Delta\lambda)$ *als Lösung von*

$$\begin{pmatrix} 0 & A^\top & I \\ A & 0 & 0 \\ \Lambda^{[k]} & 0 & X^{[k]} \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta\mu \\ \Delta\lambda \end{pmatrix} = - \begin{pmatrix} 0 \\ 0 \\ X^{[k]}\Lambda^{[k]} e - \sigma_k \zeta_k e \end{pmatrix}. \tag{4.12}$$

(iv) Setze $w^{[k+1]} := w^{[k]} + t_k \Delta w$, *wobei* $t_k > 0$ *so gewählt ist, dass* $x^{[k+1]} > 0$ *und* $\lambda^{[k+1]} > 0$ *gelten.*

(v) Setze $k := k + 1$ *und gehe zu (ii).*

Erläuterungen:

- Die Iterierten $x^{[k]}$ sind primal zulässig, da per Konstruktion stets

$$Ax^{[k]} = b, \qquad x^{[k]} > 0$$

gilt.

- Hilfssatz 4.1.5 garantiert die Durchführbarkeit des Algorithmus, falls noch $\text{Rang}(A) = m$ gefordert wird.

- Die einzige Bedingung in den KKT-Bedingungen (4.2)-(4.6), die von $(x^{[k]}, \mu^{[k]}, \lambda^{[k]})$ i.A. nicht erfüllt wird, ist die Komplementaritätsbedingung $(\lambda^{[k]})^\top x^{[k]} = 0$. Wir müssen also dafür sorgen, dass die Größe $(\lambda^{[k]})^\top x^{[k]}$ bzw.

$$\zeta_k := \frac{(x^{[k]})^\top \lambda^{[k]}}{n}$$

klein wird. Dies motiviert das Abbruchkriterium in Schritt (ii).

- Der Algorithmus enthält noch zwei Freiheitsgrade, die nicht näher spezifiziert sind: die **Schrittweite** $t_k > 0$ und den **Centering-Parameter** $\sigma_k > 0$ (beachte, dass der Term $\sigma_k \zeta_k$ die Rolle des Penalty-Parameters η übernimmt). Je nach Wahl dieser Parameter ergeben sich verschiedene Verfahren. Für Details verweisen wir auf Geiger und Kanzow [GK02], Jarre und Stoer [JS04], Wright [Wri97] und Vanderbei [Van01].

Der folgende Konvergenzsatz besagt, dass das Innere-Punkte-Verfahren eine ε-optimale Lösung mit polynomialem Aufwand findet (beim Simplexverfahren ist der Aufwand im schlechtesten Fall exponentiell).

Satz 4.1.7 (Konvergenzsatz)

Seien $\varepsilon \in (0,1)$ *beliebig und* $\{(x^{[k]}, \mu^{[k]}, \lambda^{[k]})\}_{k \in \mathbb{N}_0}$ *eine durch den Algorithmus erzeugte Folge. Es gelte*

$$\zeta_{k+1} \leq \left(1 - \frac{\delta}{n^s}\right) \zeta_k, \qquad k = 0, 1, 2, \ldots \tag{4.13}$$

mit gewissen Parametern $\delta > 0$ und $s > 0$. Der Startvektor $(x^{[0]}, \mu^{[0]}, \lambda^{[0]})$ erfülle die Bedingung

$$\zeta_0 \leq \frac{1}{\varepsilon^\kappa}, \qquad \kappa > 0.$$

Dann existiert ein Index $K \in \mathbb{N}$ mit $K = \mathcal{O}(n^s |\log(\varepsilon)|)$ und $\zeta_k \leq \varepsilon$ für alle $k \geq K$.

Beweis: Anwendung des Logarithmus als monotone Funktion auf (4.13) liefert

$$\log(\zeta_{k+1}) \leq \log\left(1 - \frac{\delta}{n^s}\right) + \log(\zeta_k), \qquad k = 0, 1, 2, \ldots$$

Rekursive Anwendung führt auf

$$\log(\zeta_k) \leq k \log\left(1 - \frac{\delta}{n^s}\right) + \log(\zeta_0) \leq k \log\left(1 - \frac{\delta}{n^s}\right) + \kappa \log\left(\frac{1}{\varepsilon}\right).$$

Wegen $\log(1 + \beta) \leq \beta$ für $\beta > -1$ gilt

$$\log(\zeta_k) \leq k\left(-\frac{\delta}{n^s}\right) + \kappa \log\left(\frac{1}{\varepsilon}\right).$$

Das Abbruchkriterium $\zeta_k \leq \varepsilon$ ist erfüllt, wenn gilt

$$k\left(-\frac{\delta}{n^s}\right) + \kappa \log\left(\frac{1}{\varepsilon}\right) \leq \log(\varepsilon).$$

Umformung liefert die Bedingung

$$k \geq (1 + \kappa)\frac{n^s}{\delta} |\log(\varepsilon)| = \mathcal{O}(n^s |\log(\varepsilon)|).$$

□

dass die Bedingung (4.13) erfüllbar ist, zeigt

Hilfsatz 4.1.8

Sei $(\Delta x, \Delta\mu, \Delta\lambda)$ Lösung von (4.12) und

$$\begin{aligned} (x^{[k]}(t), \mu^{[k]}(t), \lambda^{[k]}(t)) &= (x^{[k]}, \mu^{[k]}, \lambda^{[k]}) + t(\Delta x, \Delta\mu, \Delta\lambda), \\ \zeta_k(t) &= \frac{(x^{[k]}(t))^\top \lambda^{[k]}(t)}{n}. \end{aligned}$$

Dann gelten:

$$\begin{aligned} \Delta x^\top \Delta\lambda &= 0, \\ \zeta_k(t) &= (1 - t(1 - \sigma_k))\,\zeta_k. \end{aligned}$$

Beweis: Da $(\Delta x, \Delta\mu, \Delta\lambda)$ Gleichung (4.12) löst, gelten

$$\begin{aligned} A^\top \Delta\mu + \Delta\lambda &= 0, \\ A\Delta x &= 0, \\ \Lambda^{[k]}\Delta x + X^{[k]}\Delta\lambda &= -X^{[k]}\Lambda^{[k]}e + \sigma_k\zeta_k e. \end{aligned}$$

Multiplikation der ersten Gleichung von links mit $\Delta x^\top$ liefert $\Delta x^\top A^\top \Delta\mu + \Delta x^\top \Delta\lambda = 0$. Ausnutzen der zweiten Gleichung zeigt $\Delta x^\top \Delta\lambda = 0$.
Multiplikation der dritten Gleichung mit $e^\top$ von links liefert

$$(\lambda^{[k]})^\top \Delta x + (x^{[k]})^\top \Delta\lambda = -(x^{[k]})^\top \lambda^{[k]} + \sigma_k\zeta_k n = -(1-\sigma_k)(x^{[k]})^\top \lambda^{[k]}.$$

Zusammen mit $\Delta x^\top \Delta\lambda = 0$ folgt

$$\begin{aligned} n\zeta_k(t) &= (x^{[k]})^\top \lambda^{[k]} + t\left((\lambda^{[k]})^\top \Delta x + (x^{[k]})^\top \Delta\lambda\right) + t^2 \Delta x^\top \Delta\lambda \\ &= (x^{[k]})^\top \lambda^{[k]} \left(1 - t(1-\sigma_k)\right) \\ &= n\zeta_k \left(1 - t(1-\sigma_k)\right). \end{aligned}$$

■

Konkrete Realisierungen des Algorithmus versuchen, die Schrittweite t_k so zu wählen, dass die Iterierten $(x^{[k]}, \mu^{[k]}, \lambda^{[k]})$ in der Nähe des zentralen Pfades verbleiben und heißen **Pfadverfolgungsverfahren**. Dabei gibt es zulässige und unzulässige Varianten.
In der Regel werden die Umgebungen

$$N_2(\theta) := \{(x,\mu,\lambda) \in \mathcal{F}^\circ \mid \|X\Lambda e - \zeta e\|_2 \le \theta\zeta\}$$

und

$$N_{-\infty}(\gamma) := \{(x,\mu,\lambda) \in \mathcal{F}^\circ \mid x_i\lambda_i \ge \gamma\zeta,\ i = 1,\dots,n\}$$

mit $\zeta = x^\top\lambda/n$ und

$$\mathcal{F}^\circ := \{(x,\mu,\lambda) \mid A^\top\mu + \lambda = c,\ Ax = b,\ x > 0,\ \lambda > 0\}$$

verwendet.
Es zeigt sich, dass die Umgebung $N_2(\theta)$ in der Regel kleiner ist als $N_{-\infty}(\gamma)$, so dass Verfahren, die auf der Umgebung $N_2(\theta)$ basieren, in der Praxis langsamer konvergieren und kleinere Schrittweiten zulassen als diejenigen Verfahren, die auf der Umgebung $N_{-\infty}(\gamma)$ basieren. Die entsprechenden Verfahren heißen daher auch **short-step-Verfahren** bzw. **long-step-Verfahren**. Aus theoretischer Sicht besitzen die short-step-Verfahren häufig jedoch bessere Konvergenzeigenschaften.

4.1.1 Nichtlineare Optimierungsprobleme

Wir wollen hier nur die grundlegenden Ideen zur Erweiterung der Innere-Punkte-Verfahren auf nichtlineare Optimierungsprobleme darstellen, da es sich hierbei um ein sehr aktives Forschungsgebiet handelt. Das folgende Verfahren stammt von Byrd et al. [BHN99].
Wir betrachten wieder das Standard-Optimierungsproblem.

Finde $x \in \mathbb{R}^n$, so dass $f(x)$ minimal wird unter den Nebenbedingungen

$$\begin{aligned} g_i(x) &\leq 0, \quad i = 1, \dots, m, \\ h_j(x) &= 0, \quad j = 1, \dots, p. \end{aligned}$$

Ähnlich wie im linearen Fall eliminieren wir die Ungleichungsrestriktionen, indem wir Schlupfvariablen $s > 0$ einführen und diese über Logarithmusterme mit Gewichtungsfaktor $\eta > 0$ an die Zielfunktion koppeln:

Problem 4.1.9 (Barriere-Problem)
Finde $x \in \mathbb{R}^n$ und $s = (s_1, \dots, s_m)^\top \in \mathbb{R}^m$, so dass

$$f(x) - \eta \sum_{i=1}^{m} \log(s_i)$$

minimal wird unter den Nebenbedingungen

$$\begin{aligned} g_i(x) + s_i &= 0, \qquad i = 1, \dots, m, \\ h_j(x) &= 0, \qquad j = 1, \dots, p. \end{aligned}$$

Die Lagrangefunktion des Barriere-Problems lautet

$$L(x, s, \lambda, \mu) = f(x) - \eta \sum_{i=1}^{m} \log(s_i) + \sum_{i=1}^{m} \lambda_i (g_i(x) + s_i) + \sum_{j=1}^{p} \mu_j h_j(x).$$

Mit $S = diag(s_1, \dots, s_m)$ lauten die KKT-Bedingungen des Barriere-Problems wie folgt:

$$0 = \nabla_x L(x, s, \lambda, \mu) = \nabla f(x) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x) + \sum_{j=1}^{p} \mu_j \nabla h_j(x), \tag{4.14}$$

$$0 = \nabla_s L(x, s, \lambda, \mu) = -\eta S^{-1} e + \lambda, \tag{4.15}$$

$$0 = g_i(x) + s_i, \qquad i = 1, \dots, m, \tag{4.16}$$

$$0 = h_j(x), \qquad j = 1, \dots, p. \tag{4.17}$$

Gleichung (4.15) lautet nach Umformung

$$\eta e = S\lambda \qquad \Longleftrightarrow \qquad \eta = s_i\lambda_i, \qquad i = 1, \dots, m.$$

Zusammen mit $s_i = -g_i(x) < 0$, $i = 1, \dots, m$ aus (4.16) folgt die mit $-\eta$ **gestörte Komplementaritätsbedingung**

$$-\eta = \lambda_i g_i(x), \qquad i = 1, \dots, m. \tag{4.18}$$

Beim Innere-Punkte-Verfahren sorgt man nun wie im linearen Fall dafür, dass $s_i > 0$ gilt, was wegen (4.16) gleichbedeutend ist mit $g_i(x) < 0$. Wegen (4.18) und $-\eta < 0$ gilt dann zwangsläufig $\lambda_i > 0$ für $i = 1, \dots, m$. Die von η abhängige Lösung $(x(\eta), s(\eta), \lambda(\eta), \mu(\eta))$ bestimmt wieder den zentralen Pfad (die Existenz der Lösung sei vorausgesetzt).
Wie im linearen Fall wird das nichtlineare Gleichungssystem (4.14)-(4.17) in den Variablen x, s, λ, μ mit dem Newtonverfahren gelöst. Diese Vorgehensweise entspricht dem Lagrange-Newton-Verfahren. Da nur Gleichungsrestriktionen im Barriere-Problem auftreten, ist das Lagrange-Newton-Verfahren identisch mit dem (lokalen) SQP-Verfahren. Die Newton-Richtungen sind somit durch Lösen von quadratischen Hilfsproblemen gegeben.
Wir begnügen uns hier mit diesem Konzept eines Algorithmus und verweisen für die zahlreichen Details des Algorithmus auf Byrd et al. [BHN99]. Unter anderem muss geklärt werden, wie der Parameter η gegen Null streben soll und es muss dafür gesorgt werden, dass $s_i^{[k]} > 0$ für $i = 1, \dots, m$ erfüllt ist.

Beispiel 4.1.10

Wir möchten die Funktion

$$f(x_1, x_2) = x_1 + x_2$$

minimieren unter den Nebenbedingungen

$$\begin{aligned} g_1(x_1, x_2) = x_1^2 - x_2 &\leq 0, \\ g_2(x_1, x_2) = -x_1 &\leq 0. \end{aligned}$$

Das zugehörige Barriere-Problem lautet

$$\begin{aligned} &\textit{Minimiere} && x_1 + x_2 - \eta(\log(s_1) + \log(s_2)) \\ &\textit{unter} && x_1^2 - x_2 + s_1 = 0, \\ & && -x_1 + s_2 = 0. \end{aligned}$$

Durch Elimination der Schlupfvariablen ist es äquivalent mit der unrestringierten Minimierung der Funktion

$$\phi(x_1, x_2) := x_1 + x_2 - \eta(\log(-x_1^2 + x_2) + \log(x_1)).$$

Ein lokales Minimum von ϕ erfüllt notwendig die Bedingungen

$$\begin{aligned} 0 &= 1 - \eta\frac{-2x_1}{-x_1^2+x_2} - \eta\frac{1}{x_1}, \\ 0 &= 1 - \eta\frac{1}{-x_1^2+x_2}. \end{aligned}$$

Die zweite Gleichung impliziert $1 = \eta\frac{1}{-x_1^2+x_2}$ und damit lautet die erste Gleichung nach einigen Umformungen

$$0 = x_1^2 + \frac{1}{2}x_1 - \frac{\eta}{2}.$$

Diese quadratische Gleichung besitzt die Lösungen

$$x_1 = -\frac{1}{4} \pm \sqrt{\frac{1}{16} + \frac{\eta}{2}}.$$

Wegen $\eta > 0$ erfüllt nur die Lösung mit „+“ die Bedingung $x_1 > 0$. Somit folgt

$$x_1(\eta) = -\frac{1}{4} + \sqrt{\frac{1}{16} + \frac{\eta}{2}}.$$

Schließlich folgt hiermit die Beziehung

$$x_2(\eta) = \eta + x_1(\eta)^2 = \frac{3}{2}\eta + \frac{1}{8} - \frac{1}{2}\sqrt{\frac{1}{16} + \frac{\eta}{2}}.$$

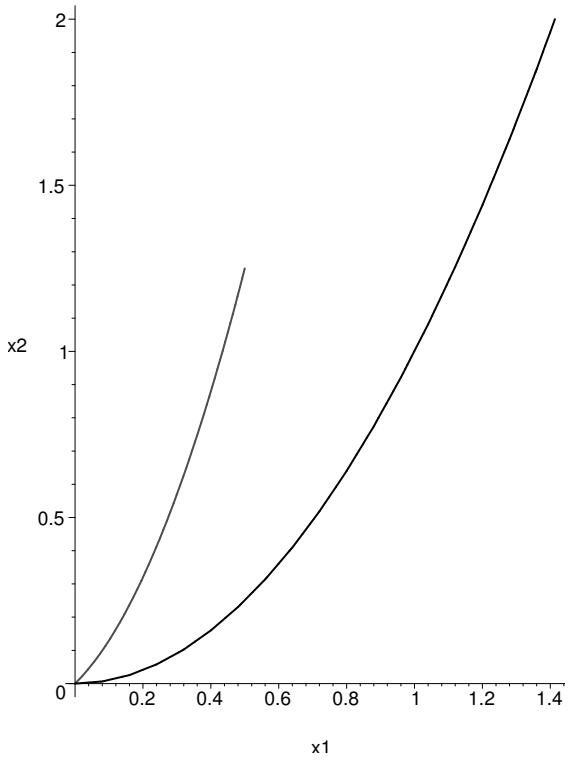
Es gilt

$$\begin{aligned} g_1(x_1(\eta), x_2(\eta)) = -\eta &< 0, \\ g_2(x_1(\eta), x_2(\eta)) = -x_1(\eta) &< 0. \end{aligned}$$

Damit sind $(x_1(\eta), x_2(\eta))$ strikt zulässig. Da das Barriere-Problem konvex ist, ist $(x_1(\eta), x_2(\eta))$ ein lokales Minimum. Grenzübergang $\eta \to 0$ liefert

$$x_1(\eta) \to 0, \qquad x_2(\eta) \to 0.$$

Der Punkt $(0,0)$ ist gerade das Minimum des Ausgangsproblems.
Die Abbildung zeigt den zentralen Pfad $\{(x_1(\eta), x_2(\eta)) \mid \eta > 0\}$.



Kapitel 5

Semiglatte Newtonverfahren für Komplementaritätsprobleme

Während SQP-Verfahren auf einer sequentiellen lokalen Approximation und Innere-Punkte-Verfahren auf einer Störung der KKT-Bedingungen für das Standard-Optimierungsproblem 1.1.1 basieren, verfolgen semiglatte Newtonverfahren die Philosophie, das KKT-System in Satz 8.4.2 **direkt** durch Anwendung eines newtonartigen Verfahrens zu lösen. Für Probleme ohne Ungleichungsrestriktionen führt dies sofort zum Lagrange-Newton-Verfahren. Treten jedoch Ungleichungsrestriktionen auf, so enthalten die KKT-Bedingungen auch die **Komplementaritätsbedingungen**

$$\lambda_i \geq 0, \qquad g_i(x) \leq 0, \qquad \lambda_i g_i(x) = 0 \qquad (i = 1, \ldots, m)$$

bzw. in Matrixnotation

$$\lambda \geq 0, \qquad g(x) \leq 0, \qquad \lambda^\top g(x) = 0.$$

Diese Bedingungen enthalten eine kombinatorische Komponente, da man a priori nicht weiß, welche Ungleichungen in der Lösung aktiv sind (dafür gilt $g_i(\hat{x}) = 0$) und welche inaktiv sind (dafür gilt dann $\lambda_i = 0$). Wüsste man die aktiven Restriktionen, so könnte man die inaktiven weglassen und wieder ein Gleichungssystem mit dem Newtonverfahren lösen. Leider ist diese Information i.A. nicht verfügbar, so dass es im Folgenden darum geht, wie man die KKT-Bedingungen aus Satz 8.4.2, also die nichtlinearen Bedingungen

$$0 = \nabla_x L(x, \lambda, \mu), \tag{5.1}$$

$$0 = h(x), \tag{5.2}$$

$$\lambda \geq 0, \quad g(x) \leq 0, \quad \lambda^\top g(x) = 0, \tag{5.3}$$

für das Standard-Optimierungsproblem 1.1.1 lösen kann. Wegen der guten lokalen Konvergenzeigenschaften (quadratische Konvergenz!) möchten wir dafür gerne ein Newtonverfahren verwenden. Leider sind Newtonverfahren zwar zur Bestimmung von Nullstellen einer Funktion geeignet, aber nicht, um auch die Ungleichungen in (5.3) behandelt zu können.

Idee:
Transformiere die Komplementaritätsbedingungen in (5.3) äquivalent um, so dass nur noch Gleichungen auftreten.

Gute Idee, aber wie kann man die Bedingungen (5.3) äquivalent als Gleichung schreiben? Dazu verwendet man sogenannte **NCP-Funktionen** $\psi : \mathbb{R} \longrightarrow \mathbb{R}$ (engl. nonlinear complementarity functions), die die Eigenschaft haben, dass für Zahlen $a, b \in \mathbb{R}$ die folgende Äquivalenz gilt:

$$\psi(a,b) = 0 \qquad \Longleftrightarrow \qquad a \geq 0,\ b \geq 0,\ a \cdot b = 0.$$

Solche Funktionen gibt es tatsächlich:

Beispiel 5.0.11

- **Min-Funktion:**

$$\psi_{min}(a,b) := \min\{a,b\}.$$

Diese Funktion ist eine NCP-Funktion. Denn:

$\Rightarrow$ *Falls* $0 = \psi_{min}(a,b)$ *gilt, so folgt* $a \geq 0$ *und* $b \geq 0$. *Da* $\min\{a,b\} = 0$ *gilt, muss* a *oder* b *Null sein, so dass das Produkt* $a \cdot b$ *ebenfalls Null ergibt.*

$\Leftarrow$ *Gilt* $a \geq 0$, $b \geq 0$ *und* $a \cdot b = 0$, *so folgt, dass* a *oder* b *Null sein muss, woraus sofort* $\min\{a,b\} = 0$ *folgt.*

ψ_{min} *ist Lipschitz-stetig, aber für* $a = b$ *nicht differenzierbar.*

- **Fischer-Burmeister-Funktion:**

$$\psi_{FB}(a,b) = \sqrt{a^2 + b^2} - a - b.$$

Diese Funktion ist auch eine NCP-Funktion. Denn:

$\Rightarrow$ *Falls* $0 = \psi_{FB}(a,b)$ *gilt, so gilt*

$$0 = \sqrt{a^2 + b^2} - a - b \qquad \Longrightarrow \qquad a^2 + b^2 = (a+b)^2 = a^2 + 2ab + b^2.$$

Folglich gilt $a \cdot b = 0$. *Sei o.B.d.A.* $b = 0$, *so dass* $0 = \sqrt{a^2} - a = |a| - a$ *folgt. Diese Gleichung ist nur erfüllt, wenn* $a \geq 0$ *gilt.*

$\Leftarrow$ *Gilt* $a \geq 0$, $b \geq 0$ *und* $a \cdot b = 0$, *so folgt, dass* a *oder* b *Null sein muss. Sei o.B.d.A.* $b = 0$, *woraus wegen* $a \geq 0$ *folgt, dass* $\psi_{FB}(a,b) = \psi_{FB}(a,0) = \sqrt{a^2} - a = |a| - a = a - a = 0$ *gilt.*

ψ_{FB} *ist Lipschitz-stetig, aber in* $(a,b) = (0,0)$ *nicht differenzierbar.*

■

Durch komponentenweise Anwendung einer NCP-Funktion ψ auf die Komplementaritätsbedingung (5.3) werden diese in äquivalente nichtlineare Gleichungen überführt:

$$0 = \psi(\lambda_i, -g_i(x)), \qquad i = 1, \dots, m.$$

In Vektornotation schreiben wir hierfür

$$0 = \vec{\psi}(\lambda, -g(x)) := \begin{pmatrix} \psi(\lambda_1, -g_1(x)) \\ \vdots \\ \psi(\lambda_m, -g_m(x)) \end{pmatrix}.$$

Kombiniert man diese Gleichungen nun mit dem Rest der KKT-Bedingungen, so erhält man das zu (5.1)-(5.3) **äquivalente Gleichungssystem**

$$F(x, \lambda, \mu) = 0 \tag{5.4}$$

mit der Funktion $F : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \longrightarrow \mathbb{R}^{n+m+p}$ gemäß

$$F(x, \lambda, \mu) := \begin{pmatrix} \nabla_x L(x, \lambda, \mu) \\ h(x) \\ \vec{\psi}(\lambda, -g(x)) \end{pmatrix}. \tag{5.5}$$

Wir haben also das Problem, einen KKT-Punkt mit (5.1)-(5.3) zu finden, transformiert auf die Aufgabe, eine Nullstelle für die Funktion F in (5.5) zu bestimmen. Dies ist ein Spezialfall des folgenden allgemeinen Nullstellenproblems.

Problem 5.0.12 **(Nullstellenproblem)**

Gesucht ist eine Nullstelle $z^* = (z_1^*, \dots, z_N^*)^\top \in \mathbb{R}^N$ *der Funktion* $F = (F_1, \dots, F_N)^\top : \mathbb{R}^N \longrightarrow \mathbb{R}^N$, *d.h. ein* z^* *mit*

$$F(z^*) = 0.$$

Ist die Funktion F mindestens stetig differenzierbar, so kann das bekannte **(klassische) Newtonverfahren** zur numerischen Bestimmung einer Nullstelle verwendet werden:

Algorithmus 5.0.13 **(klassisches Newtonverfahren)**

(0) Wähle Startwert $z^{[0]} \in \mathbb{R}^n$ *geeignet, setze* $k = 0$.

(1) Falls $F(z^{[k]}) = 0$, STOP.

(2) Berechne die Suchrichtung $d^{[k]}$ als Lösung des Gleichungssystems

$$F'(z^{[k]})d = -F(z^{[k]}).$$

(3) Setze $z^{[k+1]} = z^{[k]} + d^{[k]}$, $k := k + 1$, gehe zu (1).

Falls F' existiert und in der Nullstelle z^* invertierbar ist, konvergiert das Newtonverfahren mindestens lokal superlinear.
Leider haben NCP-Funktionen ψ die dumme Eigenschaft, dass sie entweder **nicht differenzierbar** sind (vgl. ψ_{min} und ψ_{FB}), womit das klassische Newtonverfahren nicht anwendbar ist, oder dass sie, falls sie differenzierbar sind, auf eine **nicht invertierbare Jacobimatrix von** F in der Nullstelle führen, was sich negativ auf die Durchführbarkeit und die Konvergenzeigenschaften des klassischen Newtonverfahrens auswirkt.
Wie können wir das Newtonverfahren retten?

Ansatz:
Modifiziere das Newtonverfahren, so dass es auf nicht differenzierbare Probleme angewendet werden kann, wobei die guten lokalen Konvergenzeigenschaften des klassischen Newtonverfahrens erhalten bleiben sollen. Ersetze dazu die Jacobimatrix F' im Newtonverfahren durch eine **verallgemeinerte Ableitung**.

5.1 Verallgemeinerte Ableitungen

Wir setzen ab jetzt voraus, dass F lokal Lipschitz-stetig ist, d.h. für alle $z \in \mathbb{R}^N$ gibt es $r > 0$ und eine von z abhängige Konstante $L(z)$ mit

$$\|F(z_1) - F(z_2)\| \leq L(z)\|z_1 - z_2\| \qquad \forall z_1, z_2 \in U_r(z).$$

Nach einem Satz von Rademacher ist eine lokal Lipschitz-stetige Funktion fast überall differenzierbar, z.B. ist die Fischer-Burmeister-Funktion mit Ausnahme des Nullpunkts überall differenzierbar.
Wir bezeichnen die Menge aller Punkte, an denen F differenzierbar ist, mit D_F, d.h.

$$D_F := \{z \in \mathbb{R}^N \mid F \text{ ist in } z \text{ differenzierbar}\}.$$

Dies rechtfertigt die folgende Definition, vgl. Abbildung 5.1:

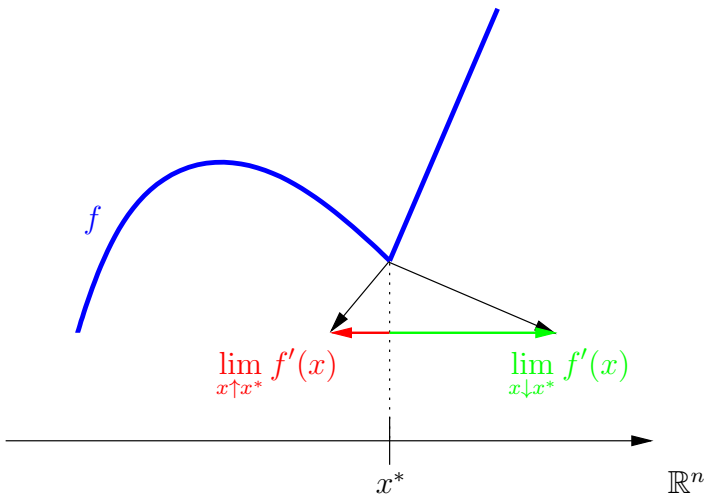


Abbildung 5.1: Illustration des B-Differentials

Definition 5.1.1 **(B(ouligand)-Differential, Clarke's verallgemeinerte Jacobimatrix)**

(a) *Das* **B(ouligand)-Differential** $\partial_B F(z)$ **von** F **in** z *ist definiert als die Menge*

$$\partial_B F(z) \;:=\; \left\{ V \in \mathbb{R}^{N\times N} \;\middle|\; V = \lim_{\substack{z_i \in D_F \\ z_i \to z}} F'(z_i) \right\}.$$

(b) *Die* **verallgemeinerte Jacobimatrix nach Clarke von** F **in** z *ist definiert als die Menge*

$$\partial F(z) := \mathit{conv}\,(\partial_B F(z)).$$

(conv bezeichnet die konvexe Hülle)

Eigenschaften (vgl. Clarke [Cla83, Prop. 2.6.2]):

- $\partial_B F(z)$ ist nichtleer und kompakt.
- $\partial F(z)$ ist nichtleer, konvex und kompakt.
- $\partial F(\cdot)$ ist oberhalbstetig, d.h. zu jedem $\varepsilon > 0$ gibt es ein $\delta > 0$ mit

$$\partial F(z) \subseteq \partial F(z^*) + U_\varepsilon(0) \quad \forall \|z - z^*\| < \delta.$$

  $\partial_B F(\cdot)$ ist ebenfalls oberhalbstetig.

- Ist F stetig differenzierbar in einer Umgebung von z, so gilt $\partial_B F(z) = \partial F(z) = \{F'(z)\}$.
- Es gilt die **Kettenregel** (vgl. Clarke [Cla83, Theorem 2.6.6]): Für eine stetig differenzierbare Funktion G und eine lokal Lipschitz-stetige Funktion H ist die Komposition $F = H \circ G$ gemäß $z \mapsto F(z) = H(G(z))$ lokal Lipschitz-stetig und es gilt
$$\partial F(z) \subseteq \text{conv } \{V \cdot G'(z) \mid V \in \partial H(G(z))\}.$$

Beispiel 5.1.2

Für die Fischer-Burmeister-Funktion

$$\psi_{FB}(a,b) = \sqrt{a^2+b^2} - a - b$$

gilt

$$\partial_B \psi_{FB}(a,b) = \begin{cases} \left\{ \left(\dfrac{a}{\sqrt{a^2+b^2}} - 1, \dfrac{b}{\sqrt{a^2+b^2}} - 1 \right) \right\}, & \text{falls } (a,b) \neq (0,0), \\ \left\{ (r,t) \;\middle|\; (r+1)^2 + (t+1)^2 = 1 \right\}, & \text{falls } (a,b) = (0,0). \end{cases}$$

Denn: Betrachte Grenzwerte

$$\psi'_{FB}(a,b) = \left(\frac{a}{\sqrt{a^2+b^2}} - 1, \frac{b}{\sqrt{a^2+b^2}} - 1 \right) \qquad (a,b) \to (0,0),\ (a,b) \neq (0,0).$$

Offenbar gilt $\|\psi'_{FB}(a,b) + (1,1)\| = 1$ *für alle* $(a,b) \neq (0,0)$ *und folglich*

$$\partial_B \psi_{FB}(a,b) \subseteq M := \{(r,t) \mid (r+1)^2 + (t+1)^2 = 1\}.$$

Andererseits liefern die Folgen $(a_i, b_i) = \frac{1}{i}(\cos\alpha, \sin\alpha) \neq (0,0)$ *mit beliebigem* $\alpha \in [0, 2\pi)$

$$\lim_{i\to\infty} \psi'_{FB}(a_i, b_i) = (\cos\alpha - 1, \sin\alpha - 1)$$

und somit $\partial_B \psi_{FB}(0,0) = M$. ∎

5.2 Ein Nichtglattes Newtonverfahren

In diesem Abschnitt entwickeln wir ein Newtonverfahren zur Bestimmung einer Nullstelle einer Lipschitz-stetigen Funktion F. Dabei ersetzen wir die nicht existierende Ableitung F' im klassischen Newtonverfahren durch ein beliebiges Element des B-Differentials $\partial_B F$.

Algorithmus 5.2.1 (Nichtglattes Newtonverfahren)

(0) Wähle Startwert $z^{[0]} \in \mathbb{R}^n$ geeignet, setze $k = 0$.

(1) Falls $F(z^{[k]} = 0$, STOP.

(2) Berechne die Suchrichtung $d^{[k]}$ als Lösung des Gleichungssystems

$$V_k d = -F(z^{[k]})$$

mit beliebigem $V_k \in \partial_B F(z^{[k]})$.

(3) Setze $z^{[k+1]} = z^{[k]} + d^{[k]}$, $k := k + 1$, gehe zu (1).

Es stellt sich natürlich sofort die Frage nach Durchführbarkeit und Konvergenz des Verfahrens.

Definition 5.2.2 **(Regularität)**
$\partial_B F(z)$ heißt **regulär**, *falls alle $V \in \partial_B F(z)$ invertierbar sind.*

Die Regularität sichert zumindest die lokale Durchführbarkeit:

Satz 5.2.3
Sei $\partial_B F(z^*)$ regulär. Dann gibt es ein $r > 0$ und eine Konstante $C > 0$, so dass $\partial_B F(z)$ für alle $\|z - z^*\| < r$ regulär ist und dort gilt

$$\|V^{-1}\| \leq C \qquad \forall V \in \partial_B F(z).$$

Beweis: Zum Beweis des Satzes nutzt man die Oberhalbstetigkeit und die Kompaktheit von $\partial_B F$ aus. ∎

Zur Konvergenz des Verfahrens wird eine weitere Eigenschaft benötigt – die Semiglattheit.

Definition 5.2.4 **(Semiglattheit)**
Sei $F : \mathbb{R}^N \longrightarrow \mathbb{R}^N$ lokal lipschitzstetig.

(i) F heißt **semiglatt in** *$z \in \mathbb{R}^N$, falls*

$$\sup_{V \in \partial F(z+h)} \|F(z+h) - F(z) - Vh\| = o(\|h\|) \qquad \text{für } \|h\| \to 0$$

gilt. F heißt semiglatt, wenn F semiglatt für alle $z \in \mathbb{R}^N$ ist.

(ii) F heißt **semiglatt von der Ordnung q in** $z \in \mathbb{R}^N$ *mit* $0 < q \leq 1$, *falls*

$$\sup_{V \in \partial F(z+h)} \|F(z+h) - F(z) - Vh\| = O(\|h\|^{1+q}) \qquad \text{für } \|h\| \to 0$$

gilt. F heißt semiglatt von der Ordnung q, wenn F semiglatt von der Ordnung q für alle $z \in \mathbb{R}^N$ ist.

Beispiel 5.2.5

Die folgenden Funktionen sind semiglatt:

- *stetig differenzierbare Funktionen, $\partial F(z) = \{F'(z)\}$*
- *konvexe Funktionen*
- *Summe, Produkt und Verkettung von semiglatten Funktionen (falls F richtungsdifferenzierbar)*
- *Fischer-Burmeister-Funktion $\psi_{FB}(a,b) = \sqrt{a^2+b^2} - a - b$, Minimumfunktion $\psi_{min}(a,b) = \min\{a,b\}$*

Die folgenden Funktionen sind semiglatt von der Ordnung 1:

- *stetig differenzierbare Funktionen mit lipschitzstetiger Ableitung*
- *Summe, Produkt und Verkettung von semiglatten Funktionen der Ordnung 1*
- *Fischer-Burmeister-Funktion $\psi_{FB}(a,b) = \sqrt{a^2+b^2} - a - b$*

■

Damit lässt sich lokale Konvergenz des nichtglatten (bzw. semiglatten) Newtonverfahrens nachweisen:

Satz 5.2.6 (Lokale Konvergenz)

Voraussetzungen:

- *F ist semiglatt in z^*.*
- *z^* ist Nullstelle von F.*

- *$\partial_B F(z^*)$ ist regulär.*

Dann gibt es ein $r > 0$, so dass das nichtglatte Newtonverfahren für alle $\|z^{[0]} - z^*\| < r$ wohldefiniert ist und die Folge $\{z^{[k]}\}_{k\in\mathbb{N}_0}$ mindestens superlinear gegen z^* konvergiert. Ist F darüber hinaus sogar semiglatt von der Ordnung q, so ist die Konvergenzordnung $1+q$.

Beweis: Die Durchführbarkeit folgt aus der Regularität von $\partial_B F(z^*)$: Es gibt $r_1 > 0$ mit

$$\|V^{-1}\| \le C \qquad \text{für alle } \|z - z^*\| < r_1,\ V \in \partial_B F(z).$$

Zur Konvergenz:
Aus der Semiglattheit in z^* folgt: Für jedes $\varepsilon > 0$ gibt es ein $r_2 > 0$, so dass für alle $\|z - z^*\| \le r_2$ und alle $V \in \partial F(z)$ gilt

$$\|F(z) - F(z^*) - V(z - z^*)\| \le \varepsilon \|z - z^*\|.$$

Wähle nun $\varepsilon := \frac{1}{2C}$ und $r = \min\{r_1, r_2\}$. Für $\|z^{[0]} - z^*\| \le r$ folgt

$$\begin{aligned}
\|z^{[1]} - z^*\| &= \|z^{[0]} - V_0^{-1} F(z^{[0]}) - z^*\| \\
&= \|V_0^{-1}\left(F(z^{[0]}) - F(z^*) - V_0(z^{[0]} - z^*)\right)\| \\
&\le \|V_0^{-1}\| \cdot \|F(z^{[0]}) - F(z^*) - V_0(z^{[0]} - z^*)\| \\
&\le C \cdot \|F(z^{[0]}) - F(z^*) - V_0(z^{[0]} - z^*)\| \\
&\le C\frac{1}{2C}\|z^{[0]} - z^*\| = \frac{1}{2}\|z^{[0]} - z^*\| \le r.
\end{aligned}$$

Induktiv folgt $\|z^{[k]} - z^*\| \le \frac{1}{2^k}\|z^{[0]} - z^*\| \longrightarrow 0$ für $k \longrightarrow \infty$.
Zur Konvergenzordnung:

$$\begin{aligned}
\|z^{[k+1]} - z^*\| &= \|z^{[k]} - V_k^{-1} F(z^{[k]}) - z^*\| \\
&= \|V_k^{-1}\left(F(z^{[k]}) - F(z^*) - V_k(z^{[k]} - z^*)\right)\| \\
&\le \|V_k^{-1}\| \cdot \|F(z^{[k]}) - F(z^*) - V_k(z^{[k]} - z^*)\| \\
&\le C \cdot \|F(z^{[k]}) - F(z^*) - V_k(z^{[k]} - z^*)\|.
\end{aligned}$$

Die rechte Seite geht superlinear bzw. von der Ordnung $1+q$ gegen Null. ∎

Bemerkung 5.2.7

- *Anstelle des B-Differentials kann man auch die verallgemeinerte Jacobimatrix ∂F nach Clarke verwenden oder eine Obermenge davon, falls die Annahmen (Regularität, Semiglattheit) dafür erfüllt sind.*

- *Das lokale nichtglatte Newtonverfahren kann durch Einführung einer Schrittweite globalisiert werden mit dem Ziel, den Konvergenzbereich zu vergrößern.*
- *Das nichtglatte (bzw. semiglatte) Newtonverfahren kann sehr viel allgemeiner für unendlichdimensionale Probleme untersucht werden, vgl. Kummer [Kum88, Kum91], Ulbrich [Ulb02, Ulb03].*

■

5.2.1 Berechnung eines Elements des B-Differentials

Wir möchten nun das nichtglatte Newtonverfahren auf das Standardoptimierungsproblem bzw. dessen KKT-Bedingungen in (5.4) mit F in (5.5) anwenden, wobei wir die Fischer-Burmeister-Funktion als NCP-Funktion wählen, also $\psi = \psi_{FB}$. Das B-Differential der Fischer-Burmeister-Funktion haben wir bereits in Beispiel 5.1.2 ausgerechnet. Das vollständige B-Differential der Funktion F zu berechnen, ist keine leichte Aufgabe, da die Funktion F in (5.5) die verketteten Funktionen

$$\psi_{FB}(\lambda_i, -g_i(x)), \qquad i = 1, \dots, m,$$

enthält, wobei die g_i, $i = 1, \dots, m$, als stetig differenzierbar vorausgesetzt sind. I.A. kann man mithilfe der Kettenregel lediglich eine Obermenge des B-Differentials (bzw. der Clarke'schen verallgemeinerten Jacobimatrix) angeben. Wenn man allerdings Elemente der Obermenge im Newtonverfahren verwendet, muss erst noch gezeigt werden, dass die Semiglattheit auch für die Obermenge gilt. Dies ist unter geeigneten Annahmen möglich, aber wir begnügen uns hier mit der Berechnung eines konkreten Elements des B-Differentials von F, indem wir einen Grenzwert ausrechnen.

Wir setzen voraus, dass die Funktionen f, g_i, $i = 1, \dots, m$, und h_j, $j = 1, \dots, p$, stetig differenzierbar sind. An den Differenzierbarkeitsstellen von ψ_{FB}, also an allen Punkten $z \in D_F$ mit

$$D_F = \left\{ z = (x, \lambda, \mu)^\top \in \mathbb{R}^{n+m+p} \mid (\lambda_i, g_i(x)) \neq 0, i = 1, \dots, m \right\}$$

ist F in (5.5) differenzierbar mit

$$F'(z) = \begin{pmatrix} \nabla_{xx} L(x, \lambda, \mu) & g'(x)^\top & h'(x)^\top \\ h'(x) & 0 & 0 \\ -R(\lambda, x) g'(x) & S(\lambda, x) & 0 \end{pmatrix}$$

und

$$R(\lambda, x) = \begin{pmatrix} R_1(\lambda_1, x) & & \\ & \ddots & \\ & & R_m(\lambda_m, x) \end{pmatrix}, \quad S((\lambda, x) = \begin{pmatrix} S_1(\lambda_1, x) & & \\ & \ddots & \\ & & S_m(\lambda_m, x) \end{pmatrix}$$

und

$$(S_i(\lambda_i,x),R_i(\lambda_i,x)) = \psi'_{FB}(\lambda_i,-g_i(x)) = \left(\frac{\lambda_i}{\sqrt{\lambda_i^2+g_i(x)^2}} - 1, \frac{-g_i(x)}{\sqrt{\lambda_i^2+g_i(x)^2}} - 1 \right).$$

Für $z=(x,\lambda,\mu)^\top \notin D_F$ definiere die Indexmenge

$$I(\lambda,x) := \{i \in \{1,\dots,m\} \mid (\lambda_i, g_i(x)) = 0\}$$

und die Folge $\{z^\ell\}_{\ell\in\mathbb{N}}$ mit $z^\ell = (x,\lambda^\ell,\mu)$ und

$$\lambda_i^\ell = \begin{cases} \lambda_i, & \text{für } i \notin I(\lambda,x), \\ \frac{1}{\ell}, & \text{für } i \in I(\lambda,x). \end{cases}$$

Dann gilt $z^\ell \longrightarrow z$ für $\ell \longrightarrow \infty$ sowie $z^\ell \in D_F$ für alle $\ell \in \mathbb{N}$. Desweiteren folgt

$$\lim_{\ell\to\infty} F'(z^\ell) = \begin{pmatrix} \nabla_{xx} L(x,\lambda,\mu) & g'(x)^\top & h'(x)^\top \\ h'(x) & 0 & 0 \\ -\hat{R}(\lambda,x)g'(x) & \hat{S}(\lambda,x) & 0 \end{pmatrix} =: \hat{V}(z) \tag{5.6}$$

mit

$$\hat{R}(\lambda,x) = \begin{pmatrix} \hat{R}_1(\lambda_1,x) & & \\ & \ddots & \\ & & \hat{R}_m(\lambda_m,x) \end{pmatrix}, \quad \hat{S}(\lambda,x) = \begin{pmatrix} \hat{S}_1(\lambda_1,x) & & \\ & \ddots & \\ & & \hat{S}_m(\lambda_m,x) \end{pmatrix}$$

und

$$\begin{aligned} (\hat{S}_i(\lambda_i,x),\hat{R}_i(\lambda_i,x)) &= \lim_{\ell\to\infty} (S_i(\lambda_i^\ell,x),R_i(\lambda_i^\ell,x)) \\ &= \begin{cases} \left(\frac{\lambda_i}{\sqrt{\lambda_i^2+g_i(x)^2}} - 1, \frac{-g_i(x)}{\sqrt{\lambda_i^2+g_i(x)^2}} - 1 \right), & \text{falls } i \notin I(\lambda,x), \\ (0,-1), & \text{falls } i \in I(\lambda,x). \end{cases} \end{aligned}$$

Mit $\hat{V}(z)$ in (5.6) haben wir damit ein konkretes Element des B-Differentials für $z \notin D_F$ berechnet, welches wir in Schritt (2) im nichtglatten Newtonverfahren 5.2.1verwenden können.

Beispiel 5.2.8

Minimiere

$$f(x_1,x_2) = (x_1-2)^2 + (x_2-3)^2$$

unter den Nebenbedingungen

$$\begin{aligned} x_2 + \frac{1}{2}x_1 - \frac{1}{2} &= 0, \\ x_2 + 2x_1^2 - 2 &\le 0, \\ x_1^2 - x_2 - 1 &\le 0. \end{aligned}$$

Lagrange-Funktion: ($x = (x_1, x_2)^\top$, $\lambda = (\lambda_1, \lambda_2)^\top$)

$$L(x,\lambda,\mu) = (x_1-2)^2+(x_2-3)^2+\mu\left(x_2+\frac{1}{2}x_1-\frac{1}{2}\right)+\lambda_1\left(x_2+2x_1^2-2\right)+\lambda_2\left(x_1^2-x_2-1\right)$$

Notwendige KKT-Bedingungen:

$$\begin{aligned} 2(x_1-2)+\frac{1}{2}\mu+(4\lambda_1+2\lambda_2)x_1 &= 0,\\ 2(x_2-3)+\mu+\lambda_1-\lambda_2 &= 0,\\ \lambda_1,\lambda_2 &\geq 0,\\ \lambda_1\left(x_2+2x_1^2-2\right) &= 0,\\ \lambda_2\left(x_1^2-x_2-1\right) &= 0. \end{aligned}$$

Nichtglatte Gleichung: ($z = (x_1, x_2, \lambda_1, \lambda_2, \mu)^\top$)

$$F(z) = \begin{pmatrix} 2(x_1-2)+\frac{1}{2}\mu+(4\lambda_1+2\lambda_2)x_1 \\ 2(x_2-3)+\mu+\lambda_1-\lambda_2 \\ x_2+\frac{1}{2}x_1-\frac{1}{2} \\ \psi_{FB}\left(\lambda_1,-\left(x_2+2x_1^2-2\right)\right) \\ \psi_{FB}\left(\lambda_2,-\left(x_1^2-x_2-1\right)\right) \end{pmatrix} = 0.$$

Startschätzung: $(x_1, x_2) = (5, -1)$, $\lambda = (0, 0)$, $\mu = 0$
Parameter: $\beta = 0.9, \sigma = 0.01, tol = 10^{-10}$
optimale Lösung: $(x_1, x_2) = (0.6, 0.2)$, $\lambda = (0, 0)$, $\mu = 28/5$

```
----- NSNEWTON VERSION 1.0 (C) Matthias Gerdts, University of Hamburg, 2006 -----

       NUMBER OF VARIABLES               :          2
       NUMBER OF EQUALITY CONSTRAINTS    :          1
       NUMBER OF INEQUALITY CONSTRAINTS  :          2
       OPTIMALITY TOLERANCE              :  0.100E-09
       LINE SEARCH PARAMETER             :  SIGMA=  0.100E-01 BETA=  0.900E+00
       DESCENT PARAMETER                 :    RHO=  0.100E-01
       MAXIMUM NUMBER OF ITERATIONS      :        100
       ROUNDOFF TOLERANCE                :  0.222E-15
       REAL WORK SPACE PROVIDED          :       1000  NEEDED :         65

-------------------------------------------------------------------------------
  ITER     ALPHA          NB           GL           KKT          |D|
-------------------------------------------------------------------------------
    0    0.0000E+00   0.1065E+03   0.1000E+02   0.1069E+03   0.1112E+02   ns
    1    0.1000E+01   0.2573E+02   0.3126E+01   0.2591E+02   0.3570E+01   ns
    2    0.1000E+01   0.5656E+01   0.3613E+00   0.5668E+01   0.1675E+01   ns
    3    0.1000E+01   0.9059E+00   0.1487E+00   0.9180E+00   0.2537E+00   ns
    4    0.1000E+01   0.3193E+00   0.3267E-01   0.3210E+00   0.2528E+00   ns
    5    0.1000E+01   0.1705E+00   0.4643E-01   0.1767E+00   0.3670E+00   ns
    6    0.1000E+01   0.2410E-01   0.1261E+00   0.1284E+00   0.4789E-01   ns
    7    0.1000E+01   0.1877E-03   0.2919E-02   0.2925E-02   0.2046E-02   ns
    8    0.1000E+01   0.1340E-07   0.1079E-05   0.1079E-05   0.6293E-06   ns
    9    0.1000E+01   0.7457E-16   0.2251E-13   0.2251E-13   0.6293E-06   ns
```

```
--------------------------------------------------------------------------------
 OBJ =        0.9800000000000001E+01
 VARIABLE   VALUE
  1        0.6000000000000090E+00
  2        0.1999999999999955E+00
 CONSTRAINT VALUE                          LAMBDA
  1        0.0000000000000000E+00         0.5600000000000009E+01
  2       -0.1079999999999983E+01        -0.3005334439095832E-16
  3       -0.8399999999999848E+00        -0.6824801378326157E-16
```

■

Kapitel 6

Dualität, Sensitivität und Parametrische Optimierung

Häufig hängen Optimierungsprobleme von Parametern ab, die selbst **keine** Optimierungsvariablen sind. Es stellt sich dann die Frage, wie die Lösung des Optimierungsproblems von den Parametern abhängt bzw. wie sensitiv diese Abhängigkeit ist? Wir werden sehen, dass die Lösung unter geeigneten Voraussetzungen (u.a. hinreichende Bedingung zweiter Ordnung, LICQ) sogar stetig differenzierbar von den Parametern abhängt. Zunächst beschränken wir uns aber auf lineare Optimierungsprobleme, um das sogenannte duale Optimierungsproblem und dessen Bedeutung für die Sensitivitätsanalyse zu untersuchen.

6.1 Lineare Optimierungsprobleme

Dem Standardproblem

$$\text{Minimiere} \qquad c^\top x \qquad \text{u.d.N.} \qquad Ax = b, x \geq 0 \tag{6.1}$$

mit $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, welches im folgenden stets als **primales lineares Optimierungsproblem** oder **Primalproblem** bezeichnet wird, kann ein sogenanntes **duales lineares Optimierungsproblem** oder **Dualproblem** zugeordnet werden.

6.1.1 Dualität

Beispiel 6.1.1

Eine Firma produziert drei Produkte P_1, P_2 und P_3. Der Gewinn pro Einheit der Produkte betrage 10, 5 *bzw.* 5.5 *Geldeinheiten. Die Produktion benötigt Rohmaterialien B_1, B_2, B_3 und B_4, von denen* 1500, 200, 1200 *und* 900 *Einheiten verfügbar sind. Zur Herstellung einer Einheit der jeweiligen Produkte werden die in der folgenden Tabelle angegebenen Mengen an Rohmaterialien benötigt.*

	P_1	P_2	P_3
B_1	30	10	50
B_2	5	0	3
B_3	20	10	50
B_4	10	20	30

Es bezeichne x_i, $i = 1, 2, 3$, die produzierten Mengen der Produkte P_i, $i = 1, 2, 3$. Die Firma möchte ihren Gewinn maximieren und muss daher das folgende primale Problem *lösen:*
Maximiere

$$10x_1 + 5x_2 + 5.5x_3$$

unter den Nebenbedingungen

$$\begin{aligned} 30x_1 + 10x_2 + 50x_3 &\leq 1500, \\ 5x_1 + 3x_3 &\leq 200, \\ 20x_1 + 10x_2 + 50x_3 &\leq 1200, \\ 10x_1 + 20x_2 + 30x_3 &\leq 900, \\ x_1, x_2, x_3 &\geq 0. \end{aligned}$$

Angenommen, eine zweite Firma bietet der ersten Firma an, alle Rohmaterialien aufzukaufen. Die zweite Firma bietet den Preis $y_i \geq 0$ pro Einheit des Rohmaterials B_i, $i = 1, \ldots, 4$. Natürlich möchte die zweite Firma ihre Kosten

$$1500y_1 + 200y_2 + 1200y_3 + 900y_4$$

minimieren. Desweiteren wird die erste Firma das Angebot nur dann akzeptieren, wenn die Preise pro Einheit der Produkte P_j, $j = 1, 2, 3$, größer oder gleich dem (nicht realisierten) Gewinn c_j, $j = 1, 2, 3$, aus dem Verkauf der Produkte sind, d.h. es muss gelten

$$\begin{aligned} 30y_1 + 5y_2 + 20y_3 + 10y_4 &\geq 10, \\ 10y_1 + 10y_3 + 20y_4 &\geq 5, \\ 50y_1 + 3y_2 + 50y_3 + 30y_4 &\geq 5.5. \end{aligned}$$

Insgesamt muss die zweite Firma also das folgende sogenannte duale Problem *lösen:*
Minimiere

$$1500y_1 + 200y_2 + 1200y_3 + 900y_4$$

unter den Nebenbedingungen

$$\begin{aligned} 30y_1 + 5y_2 + 20y_3 + 10y_4 &\geq 10, \\ 10y_1 + 10y_3 + 20y_4 &\geq 5, \\ 50y_1 + 3y_2 + 50y_3 + 30y_4 &\geq 5.5, \\ y_1, y_2, y_3, y_4 &\geq 0. \end{aligned}$$

■

Duale Optimierungsprobleme sind aus folgenden Gründen wichtig:

- Duale Programme liefern untere Schranken für den optimalen Zielfunktionswert des primalen Problems.
- Gelegentlich ist das duale Programm einfacher zu lösen als das primale Problem.
- Das duale Optimierungsproblem erlaubt, Änderungen des Minimalwerts des Primalproblems bei Störungen der Problemdaten abzuschätzen.

Diesen letzten Aspekt, die so genannte Sensitivität des Ausgangsproblems gegenüber Störungen, werden wir später im Beispiel 6.1.7 auch geometrisch illustrieren.

Problem 6.1.2 **(Duales lineares Optimierungsproblem)**
Das dem primalen Optimierungsproblem (6.1) zugeordnete **duale Problem** *lautet:*

$$\textit{Maximiere} \quad b^\top y \quad \textit{unter der Nebenbedingung} \quad A^\top y \leq c. \tag{6.2}$$

Bemerkung 6.1.3
Das duale Problem des dualen Problems liefert wieder das primale Problem (Beweis durch Ausrechnen). ■

Eine erste Eigenschaft des dualen Problems liefert

Satz 6.1.4 **(Schwacher Dualitätssatz)**
Für alle primal zulässigen Punkte x (mit $Ax = b$, $x \geq 0$) und alle dual zulässigen Punkte y (mit $A^\top y \leq c$) gilt

$$b^\top y \leq c^\top x.$$

Beweis: Die Behauptung folgt wegen $b = Ax$ und $x \geq 0$ aus

$$b^\top y = (Ax)^\top y = x^\top A^\top y \leq x^\top c = c^\top x.$$

■

Der schwache Dualitätssatz liefert eine Motivation für das duale Problem. Dual zulässige Werte liefern untere Schranken für den optimalen Zielfunktionswert des primalen Problems. Umgekehrt liefern primal zulässige Werte obere Schranken für den optimalen Zielfunktionswert des dualen Problems. Diese Eigenschaft kann z.B. beim Branch & Bound-Verfahren ür ganzzahlige Probleme zur Bestimmung unterer Schranken ausgenutzt werden.

Darüber hinaus gilt:

Satz 6.1.5

Seien x primal und y dual zulässig.

(i) Gilt $b^\top y = c^\top x$, so ist x optimal für das primale Problem in (6.1) und y ist optimal für das duale Problem 6.1.2.

(ii) $b^\top y = c^\top x$ gilt genau dann, wenn die **Komplementaritätsbedingungen** *gelten (vgl. Kapitel 1):*

$$x^\top \left(A^\top y - c\right) = 0.$$

Beweis: Die erste Behauptung in (i) folgt sofort aus dem schwachen Dualitätssatz. Die zweite Behauptung in (ii) folgt aus der Darstellung

$$c^\top x - b^\top y = c^\top x - y^\top b = c^\top x - y^\top Ax = \left(c - A^\top y\right)^\top x.$$

Gilt $c^\top x = b^\top y$, so muß jeder Summand wegen $x \geq 0$ und $c - A^\top y \geq 0$ einzeln verschwinden, damit die Summe Null ergibt. Gelten umgekehrt die Komplementaritätsbedingungen, so verschwindet jeder Summand in der rechten Summe und es folgt $c^\top x = b^\top y$. ∎

Es stellt sich die Frage, ob $c^\top x = b^\top y$ tatsächlich erreicht werden kann. Sei also x eine primal optimale Basislösung mit Basisindexmenge B, die mit dem Simplexverfahren berechnet wurde, d.h. es gilt $x_B \geq 0$ und $x_N = 0$. Insbesondere gilt dann das Optimalitätskriterium $\zeta_N \leq 0$ mit $\zeta_N^\top = c_B^\top A_B^{-1} A_N - c_N^\top$. Es sei y definiert durch

$$y^\top := c_B^\top A_B^{-1}.$$

Wir zeigen, dass das so definierte y eine Lösung des dualen Problems ist. Zunächst gilt

$$c^\top x = c_B^\top x_B = c_B^\top A_B^{-1} b = y^\top b = b^\top y.$$

Desweiteren ist y auch dual zulässig, denn es gelten $A_B^\top y = c_B$ und

$$A_N^\top y - c_N = A_N^\top \left(A_B^\top\right)^{-1} c_B - c_N = \zeta_N \leq 0.$$

Letzteres ist gerade das primale Optimalitätskriterium. Beides zusammen liefert $A^\top y \leq c$. Damit ist folgendes gezeigt:

Besitzt das primale Problem eine Optimallösung, so besitzt auch das duale Problem eine Optimallösung, wobei eine spezielle duale Optimallösung für die Basisindexmenge B durch $y^\top = c_B^\top A_B^{-1}$ gegeben ist (es kann weitere duale Lösungen geben).

Das folgende Resultat zeigt, dass auch die Umkehrung richtig ist.

Satz 6.1.6 **(Starker Dualitätssatz)**

Das primale Problem in (6.1) besitzt genau dann eine optimale Lösung, wenn das duale Problem 6.1.2 eine optimale Lösung besitzt. Darüber hinaus stimmen dann die Zielfunktionswerte überein, d.h. es gilt $c^\top x = b^\top y$. ∎

6.1.2 Sensitivität und Schattenpreise

Lösungen des dualen Problems besitzen eine für die Wirtschaftswissenschaften wichtige Interpretation. Sie geben – unter gewissen Voraussetzungen – die Sensitivität des primalen Zielfunktionswertes $c^\top x$ bzgl. Störungen im Vektor b an und sind als **Schattenpreise** bekannt. Hintergrund: b bezeichnet häufig Kapazitätsschranken, etwa die Größe eines Lagers. Für einen Unternehmer ist nun interessant, wie sich eine Vergrößerung oder Verkleinerung des Lagers auf die Kosten auswirkt.

Beispiel 6.1.7

Ein Landwirt möchte 40 Hektar mit Zuckerrüben und Weizen bepflanzen. Er hat hierfür 2400 Euro und 312 Arbeitstage zur Verfügung. Für jeden Hektar belaufen sich seine Anpflanzungskosten auf 40 Euro für Zuckerrüben und auf 120 Euro für Weizen. Für Zuckerrüben benötigt er 6 Arbeitstage pro Hektar und für Weizen 12 Arbeitstage pro Hektar. Der Gewinn beläuft sich auf 100 Euro pro Hektar für Zuckerrüben und auf 250 Euro pro Hektar für Weizen.

Der Landwirt möchte 2400 Euro investieren, aber zusätzlich unvorhergesehene Ausgaben in seiner Kalkulation berücksichtigen. Er nimmt daher an, dass $2400 + \delta$ *Euro zur Verfügung stehen, wobei* $\delta \in \mathbb{R}$ *eine durch unvorhergesehene Ausgaben verursachte Störung bezeichnet. Natürlich möchte der Landwirt seinen Gewinn maximieren:*

Minimiere

$$f(x_1, x_2) = -100x_1 - 250x_2$$

unter den Nebenbedingungen

$$\begin{aligned} x_1 + x_2 &\leq 40, \\ 40x_1 + 120x_2 &\leq 2400 + \delta, \\ 6x_1 + 12x_2 &\leq 312, \\ x_1, x_2 &\geq 0. \end{aligned}$$

In Abschnitt **??** *wurde das ungestörte Problem mit* $\delta = 0$ *grafisch gelöst:*

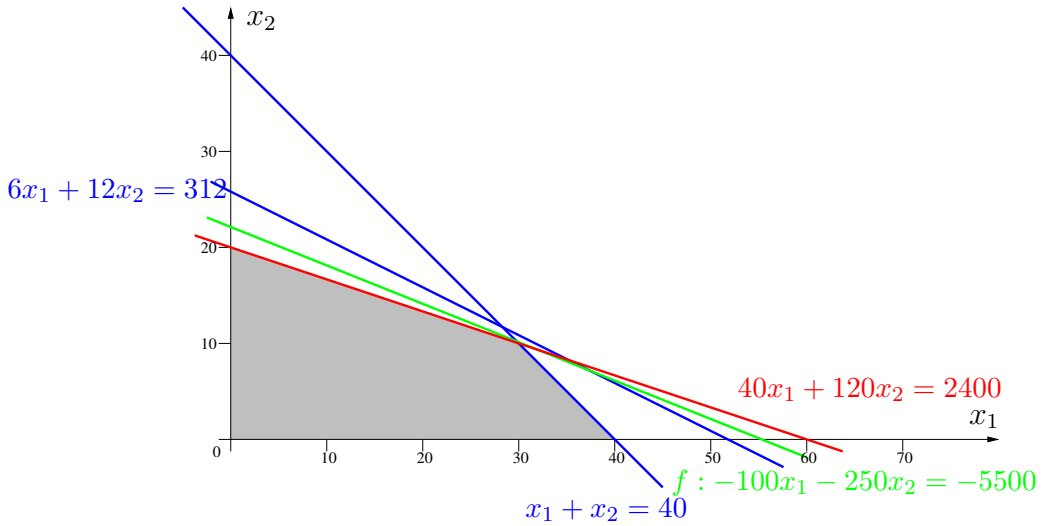


Abbildung 6.1: Lösung des ungestörten Problems mit $\delta = 0$. Die optimale Lösung lautet $(x_1, x_2)^\top = (30, 10)^\top$ mit Zielfunktionswert -5500.

Die optimale Lösung ist eine Ecke der zulässigen Menge und erfüllt die ersten beiden Nebenbedingungen ohne Schlupf, alle übrigen strikt.
Was geschieht, wenn Störungen $\delta \neq 0$ auftreten?
δ beeinflusst nur die zweite Nebenbdingung, wobei deren Steigung nicht verändert wird. Eine Änderung in δ führt also zu einer Parallelverschiebung der Geraden $40x_1 + 120x_2 = 2400$, Abbildung 6.2 zeigt die Situation für $\delta = -600$.

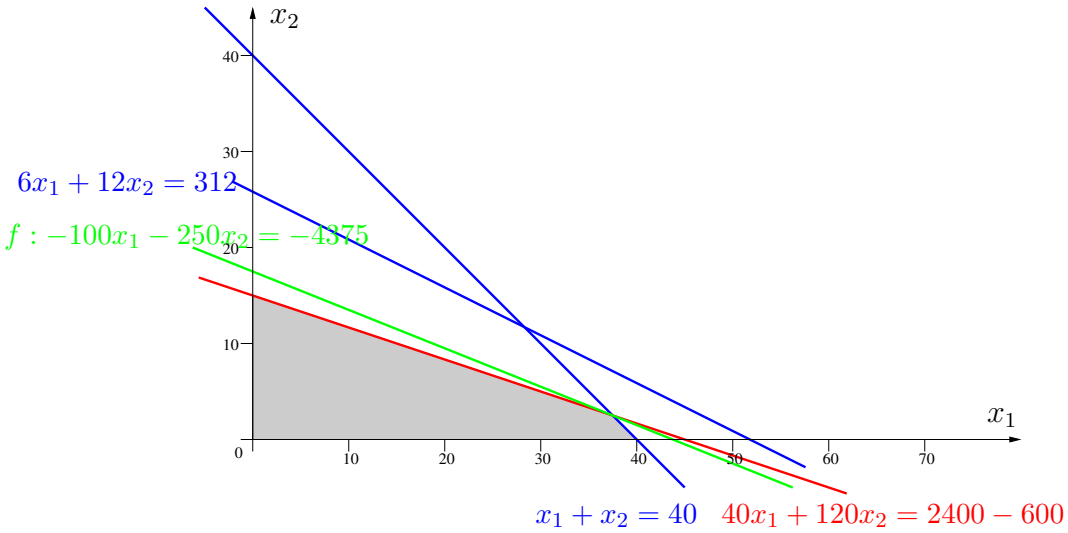


Abbildung 6.2: Lösung des Problems für $\delta = -600$. Die optimale Lösung lautet $(x_1, x_2)^\top = (37.5, 2.5)^\top$ mit Zielfunktionswert -4375.

Offenbar haben sich der zulässige Bereich und die optimale Lösung verändert, aber die optimale Lösung erfüllt immer noch die ersten beiden Nebenbedingungen ohne Schlupf, die übrigen Nebenbedingungen strikt.

Was geschieht, wenn wir δ weiter reduzieren? Abbildung 6.3 zeigt die Situation für $\delta = -1200$.

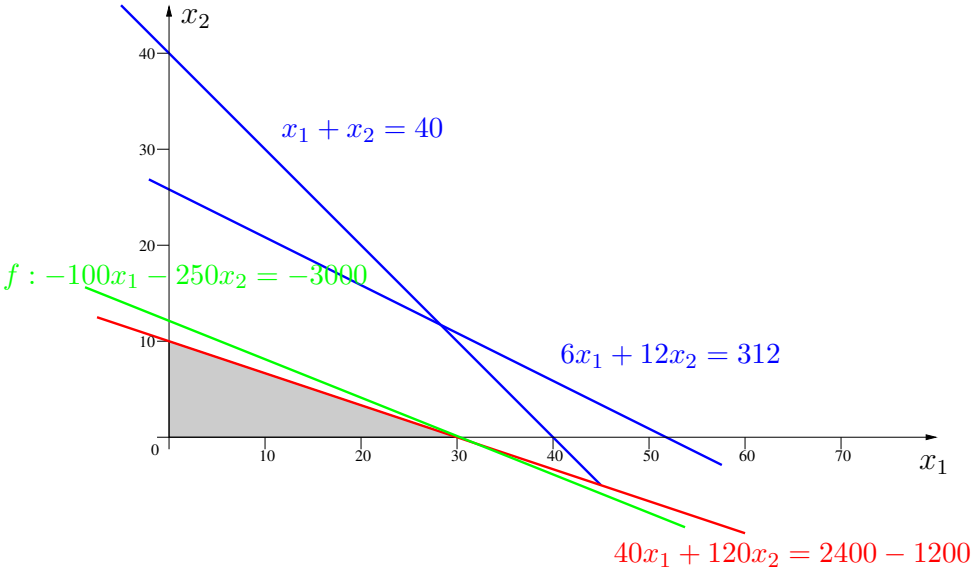


Abbildung 6.3: Lösung des Problems für $\delta = -1200$. Die optimale Lösung lautet $(x_1, x_2)^\top = (30, 0)^\top$ mit Zielfunktionswert -3000.

Jetzt erfüllt die Optimallösung die zweite Nebenbedingung und die zweite Vorzeichenbedingung ohne Schlupf, alle übrigen Nebenbedingungen strikt. Die Struktur der Optimallösung hat sich also geändert, was sich auch im qualitativen Verhalten der Minimalwerte niederschlägt.

Eine grafische Lösung des Problems für alle Werte von δ liefert die Minimallösungen

$$\begin{pmatrix} x_1(\delta) \\ x_2(\delta) \end{pmatrix} = \begin{cases} \begin{pmatrix} 0 \\ 26 \end{pmatrix}, & \textit{falls } 720 \le \delta, \\ \begin{pmatrix} 36 - \frac{1}{20}\delta \\ 8 + \frac{1}{40}\delta \end{pmatrix}, & \textit{falls } 160 \le \delta < 720, \\ \begin{pmatrix} 30 - \frac{1}{80}\delta \\ 10 + \frac{1}{80}\delta \end{pmatrix}, & \textit{falls } -800 \le \delta < 160, \\ \begin{pmatrix} 60 + \frac{1}{40}\delta \\ 0 \end{pmatrix}, & \textit{falls } -2400 \le \delta < -800, \\ \textit{keine Lösung}, & \textit{falls } \delta < -2400, \end{cases}$$

sowie die sogenannte Minimalwertfunktion

$$w(\delta) = c^\top x(\delta) = \begin{cases} -6500, & \textit{falls } 720 \le \delta, \\ -5600 - 1.25\delta, & \textit{falls } 160 \le \delta < 720, \\ -5500 - 1.875\delta, & \textit{falls } -800 \le \delta < 160, \\ -6000 - 2.5\delta, & \textit{falls } -2400 \le \delta < -800, \\ \infty, & \textit{falls } \delta < -2400, \end{cases}$$

welche die optimalen Zielfunktionswerte angibt und offenbar stetig, stückweise linear und konvex ist auf dem Intervall $[-2400, \infty)$.

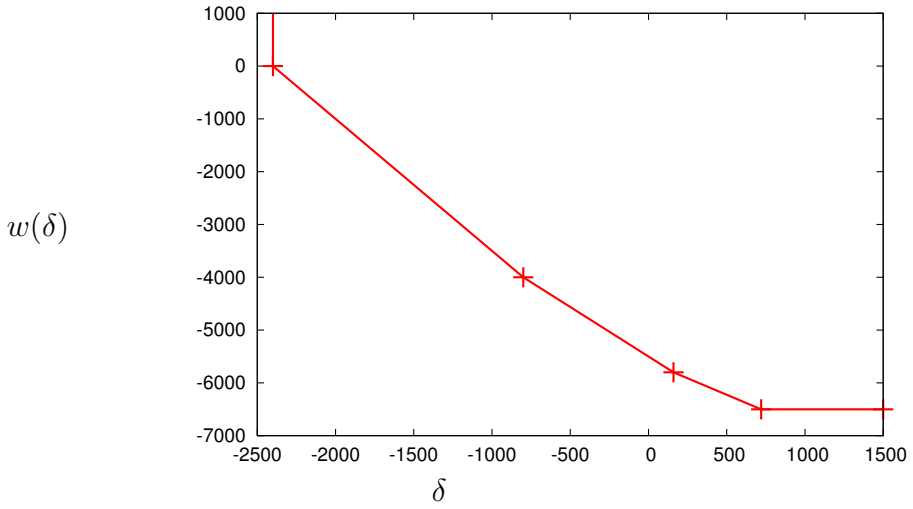


Abbildung 6.4: Minimalwertfunktion

■

Wir beschränken uns im folgenden auf Störungen des Vektors b im Standardproblem (6.1). Störungen in c oder A können ebenfalls betrachtet werden, sind aber komplizierter.

Problem 6.1.8 **(Gestörtes primales lineares Optimierungsproblem)**
Für Störungen $\Delta b \in \mathbb{R}^m$ *lautet das gestörte Standardproblem:*

$$\textit{Minimiere} \quad c^\top x \quad \textit{unter den Nebenbedingungen} \quad Ax = b + \Delta b,\ x \geq 0. \tag{6.3}$$

Offenbar entsteht für $\Delta b = 0$ gerade das Standardproblem (6.1), welches auch als **ungestörtes Problem** oder **Nominalproblem** bezeichnet wird. Jeder Störung Δb kann der zugehörige optimale Zielfunktionswert des Problems 6.1.8 zugeordnet werden (falls er existiert). Die Funktion

$$w(\Delta b) := \inf\{c^\top x \mid Ax = b + \Delta b,\ x \geq 0\}$$

heißt **optimale Wertefunktion** des gestörten Primalproblems 6.1.8. Entsprechend kann jeder Störung die Lösung des gestörten Problems zugeordnet werden:

$$\Delta b \mapsto x(\Delta b).$$

Wir wollen nun untersuchen, wie sich die optimale Lösung des gestörten Problems 6.1.8 in der Nähe der ungestörten Lösung für $\Delta b = 0$ verhält. Wir betrachten hier nur den Fall einer nicht entarteten Lösung mit Basislösung $x_B > 0$. Eine Erweiterung auf entartete Lösungen ist möglich, aber komplizierter.
Das ungestörte Problem mit $\Delta b = 0$ besitze die optimale, nicht entartete Basislösung $x_B(0) = A_B^{-1}b > 0$, $x_N(0) = 0$ mit Basisindexmenge B und Nichtbasisindexmenge N.
Nun betrachten wir kleine Störungen $\Delta b \neq 0$ und definieren

$$x_B(\Delta b) := A_B^{-1}(b + \Delta b), \qquad x_N(\Delta b) := 0. \tag{6.4}$$

Die Werte $x_B(\Delta b)$ und $x_N(\Delta b)$ sind zulässig, solange

$$x_B(\Delta b) = A_B^{-1}(b + \Delta b) = x_B(0) + A_B^{-1}\Delta b \geq 0$$

gilt. Wegen $x_B(0) = A_B^{-1}b > 0$ (keine Entartung!) kann dies für hinreichend kleine Störungen Δb stets erreicht werden.
Interessanterweise haben Störungen Δb in b keine Auswirkungen auf die negativen reduzierten Kosten

$$\zeta_N^\top = c_B^\top A_B^{-1} A_N - c_N^\top$$

im Simplexverfahren. Da $x_B(0)$ optimale Basislösung war, gilt gemäß Simplexverfahren $\zeta_N \leq 0$. Diese Optimalitätsbedingung bleibt auch für $x_B(\Delta b)$ und $x_N(\Delta b) = 0$ in (6.4) erfüllt, so dass (6.4) optimal für das gestörte Problem ist.
Für den optimalen Zielfunktionswert des gestörten Problems ergibt sich

$$w(\Delta b) = c_B^\top x_B(\Delta b) = c_B^\top x_B(0) + c_B^\top A_B^{-1}\Delta b = w(0) + y^\top \Delta b.$$

Also:

Satz 6.1.9 (Sensitivitätssatz (linearer Fall))
Sei Rang$(A) = m$ und das ungestörte Problem besitze eine nicht entartete optimale Basislösung mit Basisindexmenge B und Nichtbasisindexmenge N. Dann ist

$$x_B(\Delta b) = A_B^{-1}(b + \Delta b), \qquad x_N(\Delta b) = 0,$$

für alle Störungen $\Delta b = (\Delta b_1, \ldots, \Delta b_m)^\top$ hinreichend nahe bei 0 optimale zulässige Basislösung für das gestörte Problem und es gilt

$$\Delta w := w(\Delta b) - w(0) = y^\top \Delta b = \sum_{i=1}^{m} y_i \Delta b_i,$$

wobei die duale Lösung $y^\top = c_B^\top A_B^{-1}$ die **Sensitivität der Zielfunktion bzgl. Störungen in** *b angibt.* ∎

Aus dieser Darstellung läßt sich folgendes ablesen: Ist y_i betragsmässig sehr gross, so haben kleine Störungen Δb_i einen verhältnismäßig **großen Einfluss** auf den optimalen Zielfunktionswert des gestörten Problems. Ist y_i betragsmässig sehr klein, so haben kleine Störungen Δb_i einen verhältnismäßig **kleinen Einfluss** auf den optimalen Zielfunktionswert des gestörten Problems. Im Extremfall $y_i = 0$ haben Störungen Δb_i **keinen** Einfluss auf den optimalen Zielfunktionswert des gestörten Problems.
Durch differenzieren nach Δb ergibt sich die

Schattenpreisformel:

$$w'(\Delta b) = y.$$

Beispiel 6.1.10 **(vgl. Beispiel 6.1.7)**

Wir betrachten wieder Beispiel 6.1.7:
Minimiere $f(x_1, x_2) = -100x_1 - 250x_2$ *unter den Nebenbedingungen*

$$x_1 + x_2 \leq 40, \quad 40x_1 + 120x_2 \leq 2400 + \delta, \quad 6x_1 + 12x_2 \leq 312, \quad x_1, x_2 \geq 0.$$

Das Simplexverfahren angewendet auf das ungestörte Problem mit $\delta = 0$ *liefert:*
Starttableau:

	x_1	x_2		
x_3	$\boxed{1}$	*1*	*40*	*40*
x_4	*40*	*120*	*2400*	*60*
x_5	*6*	*12*	*312*	*52*
	100	*250*	*0*	

Pivotzeile- und spalte: $\boxed{p = 3}$, $\boxed{q = 1}$
Tableau 1:

	x_3	x_2		
x_1	*1*	*1*	*40*	*40*
x_4	*-40*	$\boxed{80}$	*800*	*10*
x_5	*-6*	*6*	*72*	*12*
	-100	*150*	*-4000*	

Pivotzeile- und spalte: $\boxed{p = 4}$, $\boxed{q = 2}$
Tableau 2:

	x_3	x_4		
x_1	*1.5*	*-0.0125*	*30*	
x_2	*-0.5*	*0.0125*	*10*	
x_5	*-3*	*-0.075*	*12*	
	-25	*-1.875*	*-5500*	

Tableau 2 ist optimal und die nicht entartete optimale Lösung lautet $x = (30, 10, 0, 0, 12)^\top$. *Die zugehörige duale Lösung ist*

$$y^\top = c_B^\top A_B^{-1} = (-25, -1.875, 0),$$

wobei $B = \{1, 2, 5\}$ *und*

$$c_B = \begin{pmatrix} -100 \\ -250 \\ 0 \end{pmatrix}, \qquad A_B = \begin{pmatrix} 1 & 1 & 0 \\ 40 & 120 & 0 \\ 6 & 12 & 1 \end{pmatrix}, \quad A_B^{-1} = \begin{pmatrix} \frac{3}{2} & -\frac{1}{80} & 0 \\ -\frac{1}{2} & \frac{1}{80} & 0 \\ -3 & -\frac{3}{40} & 1 \end{pmatrix}.$$

Der Sensitivitätssatz besagt:

(i)

$$\underbrace{\begin{pmatrix} x_1(\delta) \\ x_2(\delta) \\ x_5(\delta) \end{pmatrix}}_{=x_B(\delta)} = \underbrace{\begin{pmatrix} x_1(0) \\ x_2(0) \\ x_5(0) \end{pmatrix}}_{=x_B(0)} + A_B^{-1} \begin{pmatrix} 0 \\ \delta \\ 0 \end{pmatrix} = \begin{pmatrix} 30 \\ 10 \\ 12 \end{pmatrix} + \delta \begin{pmatrix} -\frac{1}{80} \\ \frac{1}{80} \\ -\frac{3}{40} \end{pmatrix}$$

ist optimal für δ *hinreichend nahe bei* 0. *Wie groß kann die Störung* δ *sein? Nun,* $x_B(\delta)$ *muss zulässig sein, d.h.*

$$30 - \frac{1}{80}\delta \geq 0, \quad 10 + \frac{1}{80}\delta \geq 0, \quad 12 - \frac{3}{40}\delta \geq 0.$$

Diese drei Ungleichungen sind für $\delta \in [-800, 160]$ *erfüllt. Daher ist* $x_B(\delta)$ *für alle* $\delta \in [-800, 160]$ *optimal.*

(ii)

$$\Delta w = w(\delta) - w(0) = y^\top \begin{pmatrix} 0 \\ \delta \\ 0 \end{pmatrix} = -1.875\delta.$$

Die Änderung des negativ genommenen Profits bzgl. δ *beträgt* -1.875δ.

Vergleiche dies mit der grafischen Untersuchung in Beispiel 6.1.7! ∎

Bemerkung 6.1.11 **(Achtung!)**

Die Annahme, dass die ungestörte Lösung nicht entartet ist, ist wesentlich und kann nicht ohne Weiteres weggelassen werden. Die obige Argumentation funktioniert nur, wenn die Indexmengen B *und* N *sich nicht ändern. Dies kann nur für Störungen* Δb *hinreichend nahe bei* 0 *garantiert werden. Für große Störungen oder im entarteten Fall ändern sich die Indexmengen* B *und* N *üblicherweise und die optimale Wertefunktion ist nicht differenzierbar, vgl. Beispiel 6.1.7.* ∎

6.2 Nichtlineare Optimierungsprobleme

Wir betrachten wieder das Standard-Optimierungsproblem

$$\text{Minimiere} \qquad f(x) \qquad \text{u.d.N.} \qquad x \in S,\ g(x) \leq 0,\ h(x) = 0$$

mit konvexer Menge $S \subseteq \mathbb{R}^n$ und Funktionen $f : \mathbb{R}^n \longrightarrow \mathbb{R}$, $g = (g_1, \dots, g_m)^\top : \mathbb{R}^n \longrightarrow \mathbb{R}^m$, $h = (h_1, \dots, h_p)^\top : \mathbb{R}^n \longrightarrow \mathbb{R}^p$ und bezeichnen dieses Problem im Folgenden als **primales Optimierungsproblem (Primalproblem)**.
Wir wollen das sogenannte duale Problem herleiten bzw. motivieren. Dazu betten wir das Primalproblem ein in eine Schar gestörter Probleme, indem wir die Nebenbedingungen durch Parameter $y = (y_1, \dots, y_m)^\top \in \mathbb{R}^m$ und $z = (z_1, \dots, z_p)^\top \in \mathbb{R}^p$ stören.

Problem 6.2.1 **(gestörtes primales Optimierungsproblem)**

$$\textit{Minimiere} \qquad f(x) \qquad \textit{u.d.N.} \qquad x \in S,\ g(x) \leq y,\ h(x) = z.$$

6.2.1 Dualität

Im Folgenden seien $y = (y_1, \dots, y_m)^\top \in \mathbb{R}^m$ und $z = (z_1, \dots, z_p)^\top \in \mathbb{R}^p$ beliebige Störungsvektoren. Das Primalproblem ergibt sich für $y = 0$ und $z = 0$. Wie im linearen Fall definieren wir:

Definition 6.2.2 **(Minimalwertfunktion)**
Die Funktion

$$\Phi : \mathbb{R}^{m+p} \to \bar{\mathbb{R}} := \mathbb{R} \cup \{\infty, -\infty\}$$

mit

$$\Phi(y, z) := \inf\{f(x) \mid g(x) \leq y,\ h(x) = z,\ x \in S\}$$

heißt **Minimalwertfunktion** *für das gestörte Problem.*

Das duale Problem ist motiviert durch die Aufgabe, den Graphen der Minimalwertfunktion Φ von unten durch eine Hyperebene

$$\lambda^\top y + \mu^\top z + r = \gamma \tag{6.5}$$

mit Normalenvektor $(\lambda, \mu, 1)^\top \in \mathbb{R}^{m+p+1}$ und den Variablen $(y, z, r)^\top \in \mathbb{R}^{m+p+1}$ abzustützen, vgl. Abbildung 6.5. Der Schnittpunkt dieser Hyperebene mit der r-Achse in $y = 0, z = 0$ ist $(0, 0, \gamma)^\top$.

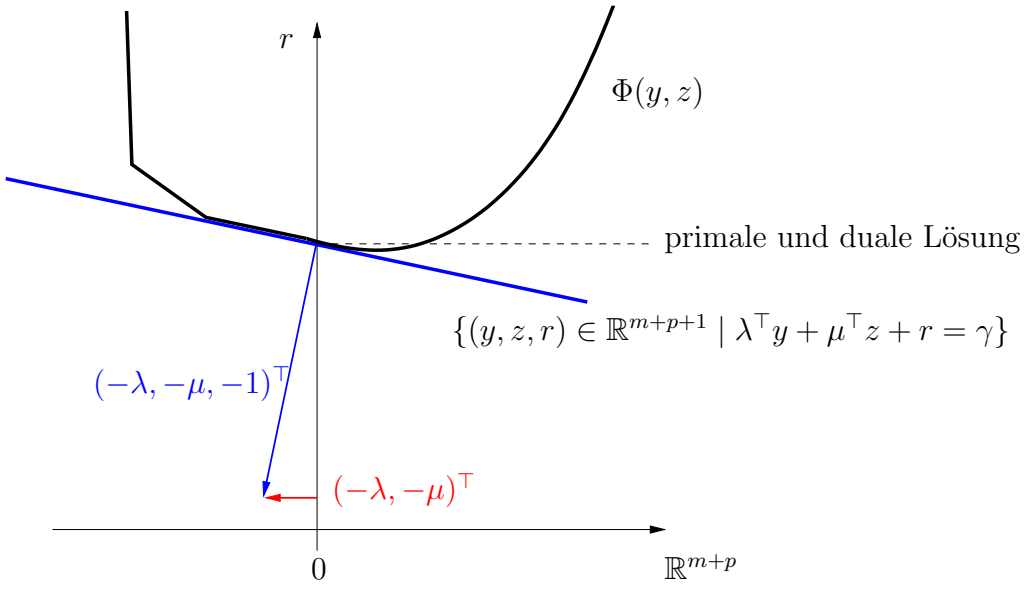


Abbildung 6.5: Grafische Interpretation des Dualproblems: Abstützung des Graphen der Minimalwertfunktion durch eine Hyperbene mit Normalenvektor $(\lambda, \mu, 1)^\top$ bzw. $(-\lambda, -\mu, -1)^\top$.

Das duale Problem läßt sich dann wie folgt formulieren:

$$\begin{array}{ll} \text{Maximiere} & \gamma \\ \text{bzgl.} & \lambda \in \mathbb{R}^m, \mu \in \mathbb{R}^p \\ \text{u.d.N.} & r \le \Phi(y,z) \qquad \forall y \in \mathbb{R}^m, z \in \mathbb{R}^p. \end{array}$$

wobei r gemäß (6.5) durch $r = \gamma - \lambda^\top y - \mu^\top z$ gegeben ist. Damit lautet das Dualprogramm

$$\begin{array}{ll} \text{Maximiere} & \gamma \\ \text{bzgl.} & \lambda \in \mathbb{R}^m, \mu \in \mathbb{R}^p \\ \text{u.d.N.} & \gamma - \lambda^\top y - \mu^\top z \le \Phi(y,z) \qquad \forall y \in \mathbb{R}^m, z \in \mathbb{R}^p. \end{array}$$

Mit der Definition von Φ ist dieses Problem wiederum äquivalent zu

$$\begin{array}{ll} \text{Maximiere} & \gamma \\ \text{bzgl.} & \lambda \in \mathbb{R}^m, \mu \in \mathbb{R}^p \\ \text{u.d.N.} & \gamma \le f(x) + \lambda^\top y + \mu^\top z \qquad \forall y \in \mathbb{R}^m, z \in \mathbb{R}^p,\ x \in S,\ g(x) \le y,\ h(x) = z. \end{array}$$

Wegen $h(x) = z$ ist dieses Problem wiederum äquivalent mit dem Problem

$$\begin{array}{ll} \text{Maximiere} & \gamma \\ \text{bzgl.} & \lambda \in \mathbb{R}^m, \mu \in \mathbb{R}^p \\ \text{u.d.N.} & \gamma \le f(x) + \lambda^\top y + \mu^\top h(x) \qquad \forall y \in \mathbb{R}^m,\ x \in S,\ g(x) \le y. \end{array}$$

Nun unterscheiden wir zwei Fälle. Zunächst untersuchen wir die Nebenbedingung

$$\gamma \leq f(x) + \lambda^\top y + \mu^\top h(x) \qquad \forall y \in \mathbb{R}^m,\ x \in S,\ g(x) \leq y \tag{6.6}$$

und nehmen an, dass es eine Komponente $\lambda_i < 0$ gibt. Damit gilt $\lambda_i y_i \to -\infty$ für $y_i \to \infty$. Für $y_i \to \infty$ ist dann auch $g_i(x) \leq y_i$ für $x \in S$ erfüllt. Es folgt, dass (6.6) nur für $\gamma = -\infty$ erfüllt ist. Da γ maximiert werden soll, ist dieser Fall uninteressant. Wir können also die Bedingung $\lambda \in \mathbb{R}^m$ durch $\lambda \geq 0$ ersetzen. In diesem Fall gilt wegen $g(x) \leq y$ stets $\lambda^\top g(x) \leq \lambda^\top y$ und die Bedingung (6.6) kann durch

$$\gamma \leq f(x) + \lambda^\top g(x) + \mu^\top h(x) \qquad \forall x \in S$$

ersetzt werden. Also ist das duale Problem äquivalent mit

$$\begin{array}{ll} \text{Maximiere} & \gamma \\ \text{bzgl.} & \lambda \geq 0, \mu \in \mathbb{R}^p \\ \text{u.d.N.} & \gamma \leq f(x) + \lambda^\top g(x) + \mu^\top h(x) \qquad \forall x \in S. \end{array}$$

Mit der Lagrangefunktion

$$L(x, \lambda, \mu) = f(x) + \lambda^\top g(x) + \mu^\top h(x)$$

und der **dualen Zielfunktion**

$$\psi(\lambda, \mu) := \inf_{x \in S} L(x, \lambda, \mu)$$

erhält man das zum Primalproblem duale Problem:

Problem 6.2.3 (Dualproblem)

$$\max_{\lambda \geq 0, \mu \in \mathbb{R}^p} \psi(\lambda, \mu) = \max_{\lambda \geq 0, \mu \in \mathbb{R}^p} \inf_{x \in S} L(x, \lambda, \mu).$$

Beispiel 6.2.4

- *(vgl. [GK02], S. 316)*

$$\begin{array}{ll} \textit{Minimiere} & f(x_1, x_2) = x_1^2 - x_2^2 \\ \textit{u.d.N.} & (x_1, x_2) \in \mathbb{R}^2 \\ & g(x_1, x_2) = x_1^2 + x_2^2 - 1 \leq 0. \end{array}$$

Die Lösung dieses Primalproblems ist $f(0, \pm 1) = -1$. Die duale Zielfunktion ist

$$\psi(\lambda) = \inf_{(x_1, x_2) \in \mathbb{R}^2} \{x_1^2 - x_2^2 + \lambda(x_1^2 + x_2^2 - 1)\} = \begin{cases} -\infty, & \textit{falls } 0 \leq \lambda < 1, \\ -\lambda, & \textit{falls } \lambda \geq 1. \end{cases}$$

Die optimale Lösung des Dualproblems $\max_{\lambda \geq 0} \psi(\lambda)$ *ist gegeben durch* $\lambda = 1$ *mit Wert* -1. *Hier besitzen Primal- und Dualproblem den gleichen Zielfunktionswert.*

- *Gegeben sei das primale lineare Optimierungsproblem*

$$\textit{Minimiere} \qquad c^\top x \qquad \textit{u.d.N.} \qquad Ax = b,\ x \in S := \{x \in \mathbb{R}^n \mid x \geq 0\}.$$

Die duale Zielfunktion lautet

$$\begin{aligned} \psi(\lambda) &= \inf_{x \geq 0} \left(c^\top x + \lambda^\top (b - Ax)\right) \\ &= \inf_{x \geq 0} \left(c^\top - \lambda^\top A\right) x + \lambda^\top b \\ &= \begin{cases} \lambda^\top b, & \textit{falls } c^\top - \lambda^\top A \geq 0, \\ -\infty, & \textit{sonst.} \end{cases} \end{aligned}$$

Damit lautet das duale Problem

$$\textit{Maximiere} \qquad b^\top \lambda \qquad \textit{u.d.N.} \qquad A^\top \lambda \leq c.$$

■

Im folgenden werden das primale und das duale Problem zueinander in Beziehung gesetzt.

Satz 6.2.5 **(Schwacher Dualitätssatz)**
Es sei x *ein primal zulässiger Punkt*[1] *und* (λ, μ) *sei dual zulässiger Punkt*[2]. *Dann gilt für den Minimalwert* $w(P)$ *des Primalproblems und den Maximalwert* $w(D)$ *des Dualproblems die Einschließung*

$$\psi(\lambda, \mu) \leq w(D) \leq w(P) \leq f(x).$$

Beweis: Für alle primal und dual zulässigen Punkte x und λ, μ gilt

$$\psi(\lambda, \mu) = \inf_{x \in S} L(x, \lambda, \mu) \leq f(x) + \underbrace{\lambda^\top}_{\geq 0} \underbrace{g(x)}_{\leq 0} + \mu^\top \underbrace{h(x)}_{=0} \leq f(x).$$

Die linke Seite hängt nicht von x ab, während die rechte Seite nicht von λ, μ abhängt. Übergang zum Supremum bzgl. $\lambda \geq 0, \mu$ auf der linken Seite und zum Infimum bzgl. $x \in S, g(x) \leq 0, h(x) = 0$ auf der rechten Seite liefert die Behauptung. ■

Stimmen der primale und der duale Zielfunktionswert überein, so sind beide Probleme bereits optimal gelöst, denn es gilt:

Satz 6.2.6 (Hinreichendes Optimalitätskriterium)
Es gelte $\psi(\hat\lambda,\hat\mu) = f(\hat x)$, wobei $\hat x$ primal zulässig und $(\hat\lambda,\hat\mu)$ dual zulässig seien. Dann ist $\hat x$ optimal für das Primalproblem und $(\hat\lambda,\hat\mu)$ ist optimal für das Dualproblem.
Außerdem gilt die **Komplementaritätsbedingung**

$$\hat\lambda_i = 0, \text{ falls } g_i(\hat x) < 0 \qquad (i = 1,\dots,m).$$

Beweis: Die erste Aussage folgt aus dem schwachen Dualitätssatz. Komplementaritätsbedingung: Angenommen es gilt $g_i(\hat x) < 0$ und $\hat\lambda_i > 0$ für mindestens ein $1 \le i \le m$. Dann folgt

$$\psi(\hat\lambda,\hat\mu) = \inf_{x\in S} L(x,\hat\lambda,\hat\mu) \le f(\hat x) + \underbrace{\hat\lambda^\top g(\hat x)}_{<0} + \underbrace{\hat\mu^\top h(\hat x)}_{=0} < f(\hat x)$$

im Widerspruch zur Voraussetzung $\psi(\hat\lambda,\hat\mu) = f(\hat x)$. ∎

Das folgende Beispiel zeigt, dass tatsächlich der Fall $w(D) < w(P)$ eintreten kann. In diesem Fall spricht man von einer **Dualitätslücke**, vgl. Abbildung 6.6.

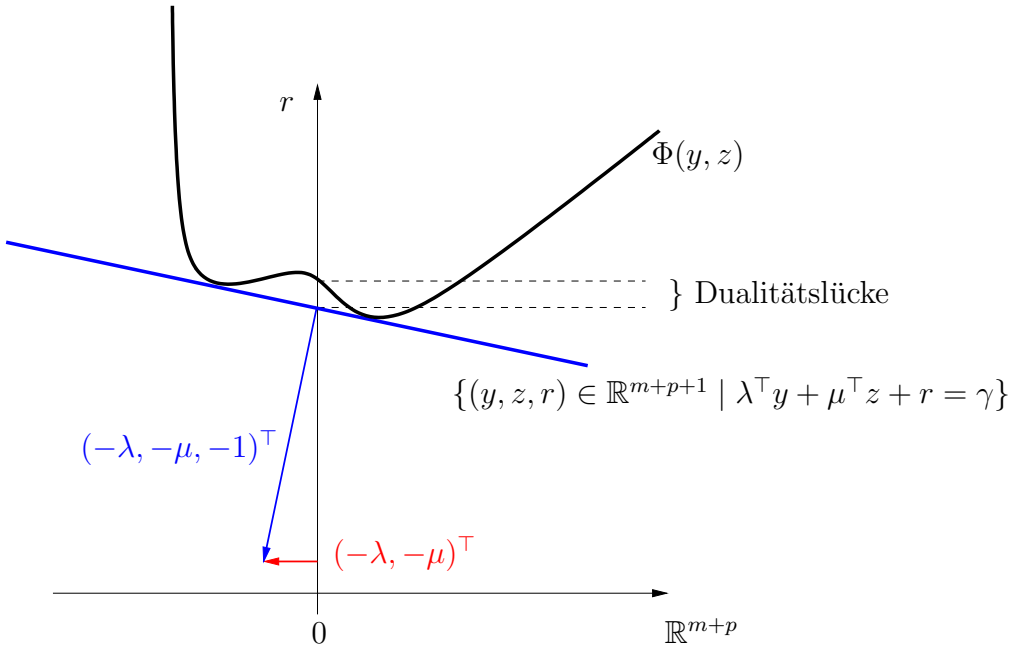


Abbildung 6.6: Grafische Interpretation des Dualproblems: Abstützung des Graphen der Minimalwertfunktion durch eine Hyperbene mit Normalenvektor $(\lambda,\mu,1)^\top$ bzw. $(-\lambda,-\mu,-1)^\top$ und Auftreten einer Dualitätslücke.

Beispiel 6.2.7 (Dualitätslücke (vgl. [BS79], S. 181))

$$\begin{array}{ll} \textit{Minimiere} & f(x_1,x_2) = -2x_1 + x_2 \\ \textit{unter} & (x_1,x_2) \in S = \{(0,0),(0,4),(4,4),(4,0),(1,2),(2,1)\}, \\ & 0 = h(x_1,x_2) = x_1 + x_2 - 3. \end{array}$$

Die Lösung dieses Primalproblems ist $(2,1)$ *mit* $f(2,1) = -3$*. Die duale Zielfunktion ist*

$$\psi(\mu) = \min\{-2x_1+3+\mu(x_1+x_2-3) \mid (x_1,x_2) \in S\} = \begin{cases} -4+5\mu, & \textit{falls } \mu \le -1, \\ -8+\mu, & \textit{falls } -1 \le \mu \le 2, \\ -3\mu, & \textit{falls } \mu \ge 2. \end{cases}$$

Die optimale Lösung des Dualproblems $\max_{\mu\in\mathbb{R}} \psi(\mu)$ *ist gegeben durch* $\mu = 2$ *mit Wert* -6*. Wegen* $-6 < -3$ *besteht eine Dualitätslücke.* ∎

Man kann zeigen, dass für konvexe Probleme keine Dualitätslücke auftritt, wenn noch eine Regularitätsbedingung (Slater Bedingung) gilt. Insbesondere tritt bei linearen Optimierungsproblemen keine Dualitätslücke auf.
Der Vorteil des Dualproblems besteht darin, dass die duale Zielfunktion $\psi(\lambda,\mu)$ auf ihrer **Domäne (wesentlicher Definitionsbereich)**

$$dom(\psi) := \{(\lambda,\mu)^\top \in \mathbb{R}^{m+p} \mid \lambda \ge 0,\ \psi(\lambda,\mu) > -\infty\}$$

konkav ist, d.h. $-\psi$ ist konvex, siehe [GK02], S. 320. Damit ist das zum Dualprogramm äquivalente Problem $\min_{\lambda\ge 0,\mu\in\mathbb{R}^p} -\psi(\lambda,\mu)$ ein konvexes Problem und kann u.U. leichter gelöst werden als das zugehörige Primalproblem (insbesondere falls die duale Zielfunktion ψ leicht berechenbar ist). Insbesondere ist jede Lösung des dualen Problems bereits ein globales Maximum der dualen Zielfunktion. Kann man zudem noch das Auftreten einer Dualitätslücke ausschließen, so besteht ein möglicher Ansatz zur Lösung des Primalproblems darin, das duale Problem zu lösen. Gemäß Satz 6.2.6 hätte man dann auch das Primalproblem gelöst. Kann das Auftreten einer Dualitätslücke nicht ausgeschlossen werden, so liefert das Dualproblem gemäß Satz 6.2.5 zumindest eine untere Schranke des optimalen primalen Zielfunktionswerts (dies kann z.B. bei Branch&Bound Verfahren ausgenutzt werden).

6.2.2 Sensitivität

In praktischen Anwendungen hängen die verwendeten mathematischen Modelle in der Regel von Parametern oder Umwelteinflüssen (z.B. Windrichtung oder Luftdichte während des Fluges) ab. Die Parameter sind dabei häufig nur geschätzt bzw. gemessen und unterliegen Schwankungen.
Wir untersuchen parametrische Optimierungsprobleme der folgenden Form.

Problem 6.2.8 **(Parametrisches Optimierungsproblem** $(NLP(w))$**)**

Für einen gegebenen Parameter $w \in \mathbb{R}^q$ finde $x \in \mathbb{R}^n$, so dass $f(x,w)$ minimal wird unter den Nebenbedingungen

$$\begin{aligned} g_i(x,w) &\leq 0, \quad i = 1,\dots,m, \\ h_j(x,w) &= 0, \quad j = 1,\dots,p. \end{aligned}$$

Beispiel 6.2.9

Betrachte das Optimierungsproblem:

$$\text{Minimiere} \qquad f(x_1,x_2,w) = wx_1 + x_2 \qquad \text{u.d.N.} \qquad h(x_1,x_2,w) = x_1^2 - x_2 - w = 0.$$

Lagrangefunktion:

$$L(x_1,x_2,\mu,w) = wx_1 + x_2 + \mu\left(x_1^2 - x_2 - w\right)$$

KKT-Bedingungen (LICQ ist wegen $\nabla_x h(x_1,x_2,w) = (2x_1,-1)^\top \neq 0$ erfüllt):

$$\nabla_x f(x_1,x_2,w) + \mu \nabla_x h(x_1,x_2,w) = \begin{pmatrix} w + 2\mu x_1 \\ 1 - \mu \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Diese nichtlineare Gleichung für x_1 und μ hat die eindeutige Lösung $x_1 = -\frac{w}{2}$, $\mu = 1$. Einsetzen von x_1 in die Gleichungsrestriktion liefert $0 = \frac{w^2}{4} - x_2 - w$ und somit $x_2 = \frac{w^2}{4} - p$. Insgesamt erhalten wir die von w abhängige Lösung

$$(\hat{x}_1(w), \hat{x}_2(w), \mu(w)) = (-w/2, w^2/4 - w, 1).$$

■

Sei $\hat{w}$ ein fester Parameter, genannt **Nominalparameter**. Wir interessieren uns für das Verhalten der optimalen Lösung $\hat{x}(w)$ als Funktion von w in einer Umgebung des Nominalparameters $\hat{w}$.
Der zulässige Bereich des parametrischen Optimierungsproblems hängt von w ab und lautet

$$\Sigma(w) := \{x \in \mathbb{R}^n \mid g_i(x,w) \leq 0,\ i = 1,\dots,m,\ h_j(x,w) = 0,\ j = 1,\dots,p\}.$$

Ebenso ist auch die Indexmenge der aktiven Ungleichungsrestriktionen von w abhängig:

$$\mathcal{A}(x,w) := \{i \mid g_i(x,w) = 0,\ 1 \leq i \leq m\}.$$

Die Lagrangefunktion lautet entsprechend

$$L(x, \lambda, \mu, w) = f(x, w) + \lambda^\top g(x, w) + \mu^\top h(x, w).$$

Definition 6.2.10 **(Streng reguläre Lösung)**

Ein lokales Minimum $\hat{x}$ von $(NLP(w))$ heißt **streng regulär** *wenn die folgenden Bedingungen gelten:*

- *$\hat{x}$ ist zulässig, d.h. $\hat{x} \in \Sigma(w)$.*
- *In $\hat{x}$ ist die Regularitätsbedingung der linearen Unabhängigkeit (LICQ) erfüllt, d.h. die Gradienten $\nabla_x g_i(\hat{x}, w)$, $i \in \mathcal{A}(\hat{x}, w)$, $\nabla_x h_j(\hat{x}, w)$, $j = 1, \ldots, p$, sind linear unabhängig.*
- *Die KKT-Bedingungen gelten in $(\hat{x}, \hat{\lambda}, \hat{\mu})$.*
- *Die* **strikte Komplementaritätsbedingung**

  $$\hat{\lambda}_i - g_i(\hat{x}, w) > 0 \qquad \forall i = 1, \ldots, m$$

  ist erfüllt.
- *Die hinreichende Bedingung zweiter Ordnung (1.6) ist erfüllt.*

Beachte: Die strikte Komplementaritätsbedingung schließt den Fall aus, dass $\hat{\lambda}_i = 0$ und $g_i(\hat{x}, w) = 0$ **gleichzeitig** gelten. Wegen $\lambda_i g_i(\hat{x}, w) = 0$ gilt also entweder $\lambda_i = 0, g_i(\hat{x}, w) < 0$ oder $\lambda_i > 0, g_i(\hat{x}, w) = 0$.

Das folgende sehr wichtige Resultat findet sich in den Büchern [Fia83], [FM90], [Spe93].

Satz 6.2.11 **(Sensitivitätssatz)**

Seien $f, g_1, \ldots, g_m, h_1, \ldots, h_p : \mathbb{R}^n \times \mathbb{R}^q \to \mathbb{R}$ zweimal stetig differenzierbar und $\hat{w}$ Nominalparameter. Sei $\hat{x}$ ein streng reguläres lokales Minimum von $(NLP(\hat{w}))$, wobei $\hat{\lambda}$, $\hat{\mu}$ die entsprechenden Lagrange-Multiplikatoren bezeichnen. Dann existieren Umgebungen $V_\varepsilon(\hat{w})$ und $U_\delta(\hat{x}, \hat{\lambda}, \hat{\mu})$, so dass $(NLP(w))$ ein eindeutiges, streng reguläres lokales Minimum

$$(x(w), \lambda(w), \mu(w)) \in U_\delta(\hat{x}, \hat{\lambda}, \hat{\mu})$$

für alle $w \in V_\varepsilon(\hat{w})$ besitzt. Die Indexmenge der aktiven Ungleichungen ändert sich in dieser Umgebung nicht, d.h. es gilt $\mathcal{A}(\hat{x}, \hat{w}) = \mathcal{A}(x(w), w)$ für alle $w \in V_\varepsilon(\hat{w})$.

Zusätzlich ist $(x(w),\lambda(w),\mu(w))$ stetig differenzierbar bzgl. w mit

$$\begin{pmatrix} \frac{dx}{dw}(\hat{w}) \\ \frac{d\lambda}{dw}(\hat{w}) \\ \frac{d\mu}{dw}(\hat{w}) \end{pmatrix} = - \begin{pmatrix} \nabla^2_{xx} L & (g'_x)^\top & (h'_x)^\top \\ \hat{\Lambda} \cdot g'_x & \hat{\Gamma} & 0 \\ h'_x & 0 & 0 \end{pmatrix}^{-1} \cdot \begin{pmatrix} \nabla^2_{xw} L \\ \hat{\Lambda} \cdot g'_w \\ h'_w \end{pmatrix}, \tag{6.7}$$

wobei $\hat{\Lambda} = \operatorname{diag}(\hat{\lambda}_1,\dots,\hat{\lambda}_m)$, $\hat{\Gamma} = \operatorname{diag}(g_1,\dots,g_m)$ ist. Alle Funktionen und Ableitungen sind in $(\hat{x},\hat{\lambda},\hat{\mu},\hat{w})$ ausgewertet.

Beweis: Der Beweis basiert auf dem Satz über implizite Funktionen, der auf die KKT-Bedingungen angewendet wird. ∎

Beispiel 6.2.12 (vgl. Beispiel 6.2.9)
Betrachte das Optimierungsproblem:

$$\text{Minimiere} \qquad f(x_1,x_2,w) = wx_1 + x_2 \qquad \text{u.d.N.} \qquad h(x_1,x_2,w) = x_1^2 - x_2 - w = 0.$$

Wir betrachten das Problem für den Nominalparameter $\hat{w} = 2$ und erhalten nach Beispiel 6.2.9 die Lösung

$$\hat{x}_1 = -1, \qquad \hat{x}_2 = -1, \qquad \hat{\mu} = 1.$$

Mit der Lagrangefunktion

$$L(x_1,x_2,\mu,w) = wx_1 + x_2 + \mu\left(x_1^2 - x_2 - w\right)$$

erhält man in diesem Punkt $h'_w(\hat{x}_1,\hat{x}_2,\hat{w}) = -1$ sowie

$$\nabla^2_{xx} L(\hat{x}_1,\hat{x}_2,\hat{\mu},\hat{w}) = \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix}, \quad \nabla^2_{xw} L(\hat{x}_1,\hat{x}_2,\hat{\mu},\hat{w}) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, h'_x(\hat{x}_1,\hat{x}_2,\hat{w}) = (-2,-1).$$

Das Gleichungssystem (6.7) lautet damit

$$\begin{pmatrix} 2 & 0 & -2 \\ 0 & 0 & -1 \\ -2 & -1 & 0 \end{pmatrix} \begin{pmatrix} \frac{dx_1}{dw}(\hat{w}) \\ \frac{dx_2}{dw}(\hat{w}) \\ \frac{d\mu}{dw}(\hat{w}) \end{pmatrix} = - \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}$$

und besitzt die Lösung

$$\frac{dx_1}{dw}(\hat{w}) = -\frac{1}{2}, \quad \frac{dx_2}{dw}(\hat{w}) = 0, \quad \frac{d\mu}{dw}(\hat{w}) = 0.$$

Vergleiche dies mit der von w abhängigen Lösung in Beispiel 6.2.9. ∎

Mithilfe der Sensitivitäten kann man lokal vorhersagen, wie sich die optimale Lösung bei Abweichungen vom Nominalparameter $\hat{w}$ verhalten wird. Die Taylorapproximation

$$x(w) \approx x(\hat{w}) + \frac{dx}{dw}(\hat{w})(w - \hat{w})$$

liefert für $w \approx \hat{w}$ sogar eine Näherung $x(w)$, die sehr kostengünstig zu berechnen ist (Matrix-Vektor-Multiplikation), wenn man die nominale Lösung $x(\hat{w})$ und die Sensitivität $\frac{dx}{dw}(\hat{w})$ bereits vorab berechnet hat. Diese Approximation kann man in Echtzeit berechnen und sie eignet sich daher, wenn man sehr schnell eine Lösung benötigt (Echtzeitoptimierung!), vgl. etwa Büskens und Maurer [BM01] und Büskens und Gerdts [BG01].

Bemerkung 6.2.13

- *Gemäß (6.7) können die* **Sensitivitätsableitungen** $\frac{dx}{dw}(\hat{w})$, $\frac{d\lambda}{dw}(\hat{w})$ *und* $\frac{d\mu}{dw}(\hat{w})$ *durch Lösen eines* **linearen Gleichungssystems** *numerisch berechnet werden. Allerdings werden hierzu die zweiten Ableitungen $\nabla^2_{xx} L$ und $\nabla^2_{xw} L$ benötigt, die häufig nur numerisch berechnet werden können.*

- *Die Größenordnung der Sensitivitätsableitungen erlaubt eine quantitative Aussage darüber, wie stark die Lösung von den Komponenten des Parametervektors $w \in \mathbb{R}^q$ abhängt.*

- *Leider macht der Sensitivitätssatz keine Aussagen darüber, wie groß die Umgebung $V_\varepsilon(\hat{w})$ ist.*

  Problem: Für zu große Abweichungen w vom Nominalwert $\hat{w}$ wird sich im allgemeinen die Indexmenge $\mathcal{A}(\hat{x}, w)$ ändern. Für diese Situation liefert der Satz keine Aussage.

Kapitel 7

Ganzzahlige Optimierung

Bisher konnten die Optimierungsvariable stets reelle Werte innerhalb des zulässigen Bereichs annehmen. In diesem Abschnitt wollen wir auch ganzzahlige Optimierungsvariablen zulassen, die nur ganzzahlige Werte annehmen dürfen. Mit ganzzahligen Variablen lassen sich Wahrheitswerte, Alternativen aus endlich vielen Möglichkeiten, Stückzahlen, Personalbedarf und vieles mehr modellieren. Auf den ersten Blick erscheinen Optimierungsprobleme mit ganzzahligen Variablen einfacher zu sein als Probleme mit reellen Variablen, da in einer kompakten Menge stets nur endlich viele ganzzahlige Kombinationen möglich sind. Allerdings machen Konstrukte wie der Gradient einer Funktion für ganzzahlige Variablen keinen Sinn mehr, so dass gradientenbasierte Verfahren nicht mehr auf ganzzahlige Probleme anwendbar sind. Wir benötigen also andersartige Verfahren.

Gemischt-ganzzahlige Optimierungsprobleme unterscheiden sich von reellen Optimierungsproblemen dadurch, dass neben den üblichen Nebenbedingungen noch Nebenbedingungen des Typs

$$x_j \in \mathbb{Z}$$

auftreten. Schreiben wir alle Nebenbedingungen als implizite Nebenbedingungen, so erhalten wir

Problem 7.0.14 **(Gemischt-ganzzahliges Optimierungsproblem)**
Gegeben sei die Zielfunktion

$$f : \mathbb{R}^m \times \mathbb{Z}^{n-m} \longrightarrow \mathbb{R}$$

und die zulässige Menge

$$\Sigma \subseteq \mathbb{R}^m \times \mathbb{Z}^{n-m} \ .$$

Das gemischt-ganzzahlige Optimierungsproblem lautet:

$$\textit{Minimiere} \qquad f(x) \qquad \textit{u.d.N.} \qquad x \in \Sigma \ .$$

Σ wird im Allgemeinen durch Gleichungen und Ungleichungen beschrieben, und verschiedene Beschreibungen von Σ können verschieden gut brauchbar sein.

Ist $m = 0$, so erhalten wir ein ganzzahliges Optimierungsproblem, ist $m = n$, so erhalten wir ein reelles Optimierungsproblem.
Sind für $x_j \in \mathbb{Z}$ nur Werte aus $\{0, 1\}$ zugelassen, was durch die zusätzlichen expliziten Restriktionen

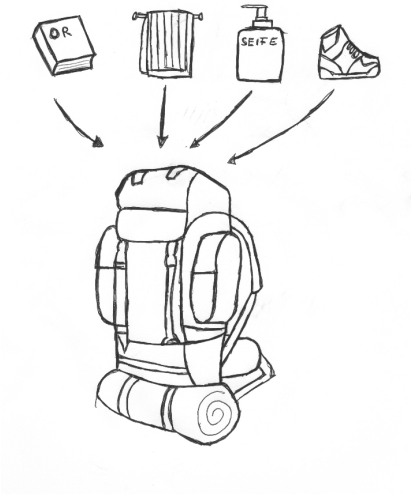
$$0 \leq x_j \leq 1$$

erzwungen werden kann, so heißt x_j **0-1-Variable** oder **Boolesche Variable**. Ein Optimierungsproblem mit lauter Booleschen Variablen heißt auch **kombinatorisches Optimierungsproblem**. Eine sehr ausführliche und aktuelle Darstellung der kombinatorischen Optimierung wird in [KV08] gegeben.
Ganzzahlige bzw. kombinatorische Optimierungsprobleme heißen auch **diskrete Optimierungsprobleme**, reelle Optimierungsprobleme heißen auch **kontinuierliche Optimierungsprobleme**.

7.1 Beispiele für ganzzahlige Optimierungsprobleme

Beispiel 7.1.1 (Rucksackpackproblem)

Maximal N Gegenstände sollen in einen Rucksack gepackt werden.



Jeder Gegenstand $j \in \{1, \ldots, N\}$ besitzt das Gewicht a_j und den Wert c_j. Die Aufgabe besteht im Packen eines Rucksacks mit maximalem Wert, wobei das Gewicht des Rucksacks den Wert A nicht überschreiten darf. Dies führt auf das folgende Optimierungsproblem:

$$\text{Maximiere} \quad \sum_{j=1}^{N} c_j x_j \quad \text{u.d.N.} \quad \sum_{j=1}^{N} a_j x_j \leq A,\ x_j \in \{0,1\} \quad (j = 1, \ldots, N)\ !$$

Dabei bedeutet

$$x_j = \begin{cases} 1, & \textit{Gegenstand } j \textit{ wird eingepackt,} \\ 0, & \textit{Gegenstand } j \textit{ wird nicht eingepackt.} \end{cases}$$

Ein naiver Ansatz besteht in der Enumeration der höchstens 2^N zulässigen Kombinationen. ∎

Beispiel 7.1.2 **(Standortplanung)**

Beim Standortplanungsproblem soll über die Eröffnung von Depots entschieden werden. Sei $I = \{1, \dots, m\}$ eine Menge von Kunden und $J = \{1, \dots, n\}$ eine Menge von potentiellen Depots, die an verschiedenen Standorten eröffnet werden können. Die Eröffnung eines Depots $j \in J$ verursacht feste Instandhaltungskosten c_j. Jeder Kunde $i \in I$ hat einen Bedarf b_i und kann von jedem Depot bedient werden. Jedes Depot $j \in J$ hat eine Lagerkapazität u_j. Die Transportkosten für den Transport einer Einheit von Depot $j \in J$ zu Kunde $i \in I$ betragen h_{ij} Geldeinheiten. Ziel des Standortplanungsproblems ist es zu entscheiden, welche Depots eröffnet werden sollen, so dass der Bedarf der Kunden bei minimalen Gesamtkosten befriedigt werden kann. Mit Hilfe der Variablen $x_j \in \{0, 1\}$ wird entschieden, ob Depot $j \in J$ eröffnet wird ($x_j = 1$) oder nicht ($x_j = 0$). Die Variable y_{ij} bezeichnet die zum Kunden i transportierte Menge von Depot j. Das Standortplanungsproblem führt auf das folgende gemischt-ganzzahlige Optimierungsproblem.

$$
\begin{array}{lll}
\text{Minimiere} & \displaystyle\sum_{j\in J} c_j x_j + \sum_{\substack{i\in I\\ j\in J}} h_{ij} y_{ij} & \\
\text{u.d.N.} & x_j \in \{0,1\} & (j \in J)\ , \\
& y_{ij} \geq 0 & (i \in I,\ j \in J), \\
& \displaystyle\sum_{j\in J} y_{ij} = b_i & (i \in I)\ , \\
& \displaystyle\sum_{i\in I} y_{ij} \leq x_j u_j & (j \in J)\ !
\end{array}
$$

In dieser Problemstellung treten sowohl diskrete Optimierungsvariable $x_j \in \{0, 1\}$ als auch reellwertige Optimierungsvariable $y_{ij} \geq 0$ auf. Die letzte Nebenbedingung erzwingt im Fall $x_j = 0$ (Depot j wird nicht eröffnet), dass $y_{ij} = 0$ für die aus Depot j gelieferten Mengen gilt. ∎

Beispiel 7.1.3 **(Zuweisungsproblem)**

Gegeben seien n Mitarbeiter $E_1, E_2, \dots, E_n$, n Aufgaben $T_1, T_2, \dots, T_n$ und Kosten c_{ij} für die Zuweisung des Mitarbeiters E_i zu Aufgabe T_j. Die Aufgabe eines Projektplaners ist es, den Mitarbeitern eineindeutig Aufgaben zuzuweisen, so dass die Kosten dieser Zuweisung minimal ausfallen. Mathematisch kann diese Aufgabe wie folgt modelliert werden. Es seien $x_{ij} \in \{0, 1\}$ $(i, j = 1, \dots, n)$ definiert durch

$$
x_{ij} = \begin{cases} 0, & \text{falls Mitarbeiter } i \text{ nicht Aufgabe } j \text{ zugewiesen wird,} \\ 1, & \text{falls Mitarbeiter } i \text{ Aufgabe } j \text{ zugewiesen wird.} \end{cases}
$$

Das Zuweisungsproblem lautet dann:

$$\begin{aligned} &\text{Minimiere} && \sum_{i,j=1}^{n} c_{ij}x_{ij} && \\ &\text{u.d.N.} && \sum_{j=1}^{n} x_{ij} = 1 && (i = 1,\dots,n)\ , \\ & && \sum_{i=1}^{n} x_{ij} = 1 && (j = 1,\dots,n)\ , \\ & && x_{ij} \in \{0,1\} && (i,j = 1,\dots,n)\ ! \end{aligned}$$

Wegen $x_{ij} \in \{0,1\}$ garantieren die Nebenbedingungen

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad (i = 1,\dots,n),$$

dass Mitarbeiter i genau eine Aufgabe zugewiesen wird. Entsprechend garantieren

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad (j = 1,\dots,n),$$

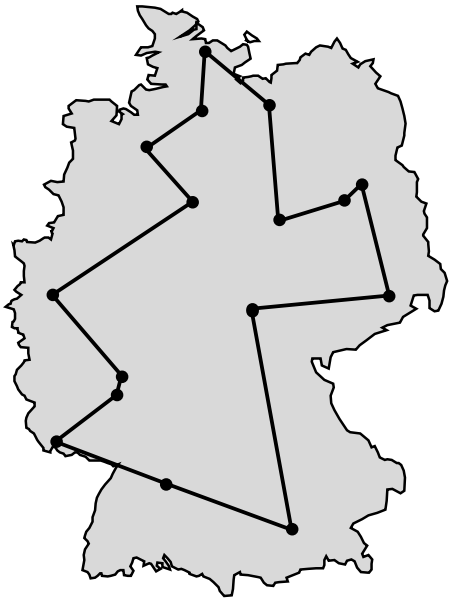
dass jede Aufgabe durch genau einen Mitarbeiter erledigt wird.
Aus theoretischer Sicht scheint dieses Problem einfach zu sein, da es nur endlich viele Möglichkeiten für die Belegung der Variablen x_{ij} $(i,j = 1,\dots,n)$ gibt.
Ein naiver Lösungsansatz besteht daher in der Enumeration aller zulässigen Punkte und der anschließenden Wahl des besten Punktes.
Fasst man die Variablen $x_{ij} \in \{0,1\}$ $(i,j = 1,\dots,n)$ als Einträge einer $n \times n$-Matrix auf, so erzwingen die Nebenbedingungen, dass in jeder Zeile und Spalte dieser Matrix genau ein Eintrag 1 zu finden ist.
Demnach gibt es n Möglichkeiten, die 1 in der ersten Zeile zu platzieren, jeweils $n-1$ Möglichkeiten für die zweite Zeile, usw.
Es gibt also $n! = n(n-1)\cdot(n-2)\cdots 2\cdot 1$ Punkte, die die Nebenbedingungen erfüllen. Der naive Enumerationsalgorithmus erfordert es, die zu minimierende Funktion für jede Kombination auszuwerten. Die folgende Tabelle zeigt, wie schnell die Anzahl der Auswertungen mit der Problemgrösse n ansteigt:

n	10	20	30	50	70
Auswertungen	$3.629 \cdot 10^6$	$2.433 \cdot 10^{18}$	$2.653 \cdot 10^{32}$	$3.041 \cdot 10^{64}$	$1.198 \cdot 10^{100}$

■

Beispiel 7.1.4 (Travelling Salesman Problem, Handlungsreisendenproblem)

Seien Städte $V = \{1, \dots, n\}$ *und Verbindungen* $E \subseteq V \times V$ *zwischen den Städten gegeben, wobei* c_{ij} *die Länge der Verbindung* $(i, j) \in E$ *bezeichne. Eine* Tour *ist ein geschlossener gerichteter Pfad, der jede Stadt genau einmal enthält. Die Aufgabe besteht in der Bestimmung einer Tour minimaler Länge.*



Definiere die Variablen

$$x_{ij} \in \{0, 1\}$$

für alle $(i, j) \in E$ *durch*

$$x_{ij} = \begin{cases} 1, & \textit{falls } (i,j) \textit{ Teil der Tour ist,} \\ 0 & \textit{sonst.} \end{cases}$$

Die Bedingung, dass jede Stadt genau einmal besucht wird, wird durch die Nebenbedingungen

$$\sum_{\{i:(i,j)\in E\}} x_{ij} = 1 \qquad (j \in V), \tag{7.1}$$

$$\sum_{\{j:(i,j)\in E\}} x_{ij} = 1 \qquad (i \in V) \tag{7.2}$$

garantiert. Diese Bedingungen finden sich auch schon im Zuweisungsproblem 7.1.3. Sie ermöglichen jedoch noch unzusammenhängende Subtouren. Um diese auszuschließen, wird für jede disjunkte Partition von V *in nichtleere Mengen*

$$\begin{aligned} U &\subseteq V, \\ U^c &\subseteq V \end{aligned}$$

gefordert, dass es eine Verbindung

$$(i, j) \in E \text{ mit } i \in U,\ j \in U^c$$

und eine Verbindung

$$(k, \ell) \in E \text{ mit } k \in U^c,\ \ell \in U$$

gibt. Hierbei müssen die einelementigen Mengen wegen (7.1) und (7.2) nicht betrachtet werden, so dass die Bedingungen

$$\sum_{\{(i,j)\in E: i\in U, j\in V\setminus U\}} x_{ij} \geq 1 \tag{7.3}$$

für alle $U \subseteq V$ mit $2 \leq |U| \leq |V| - 2$ resultieren. Dabei bezeichnet $|U|$ die Anzahl der Elemente von U.

Insgesamt lautet das Travelling Salesman Problem wie folgt:

Minimiere

$$\sum_{(i,j)\in E} c_{ij} x_{ij}$$

u.d.N.

$$\begin{aligned}
&\sum_{\{i:(i,j)\in E\}} x_{ij} = 1 && (j \in V)\ , \\
&\sum_{\{j:(i,j)\in E\}} x_{ij} = 1 && (i \in V)\ , \\
&\sum_{\{(i,j)\in E: i\in U, j\in V\setminus U\}} x_{ij} \geq 1 && (U \subseteq V,\ 2 \leq |U| \leq |V| - 2)\ , \\
&x_{ij} \in \{0, 1\} && ((i,j) \in E)\ !
\end{aligned}$$

■

Beispiel 7.1.5 (Maschinenbelegungsplaene)

Gegeben sei eine Menge

$$I = \{1, \ldots, m\}$$

von Maschinen und eine Menge

$$J$$

von Jobs. Jeder Job muss von jeder Maschine bearbeitet werden, die Bearbeitungszeit ist

$$p_{ij} \qquad (i \in I,\ j \in J)\ .$$

Jede Maschine kann nur einen Job zur Zeit bearbeiten.
Die Reihenfolge der Abarbeitung liegt für jeden Job fest, d. h. Job j muss bearbeitet werden von den Maschinen

$$i_1 = j(1),\ i_2 = j(2), \ldots, i_m = j(m),$$

in dieser Reihenfolge.
Als Variablen wählen wir die Anfangszeiten

$$t_{ij}$$

des Jobs j auf der Maschine i.
Da die $(r+1)$-te Bearbeitungsstufe des Jobs j nicht beginnen kann, bevor die r-te abgeschlossen ist, erhalten wir die Nebenbedingungen

$$t_{j(r+1),j} \geq t_{j(r),j} + p_{j(r),j} \tag{7.4}$$

für $r = 1, \ldots, m-1$, $j \in J$.
Betrachte jetzt die spezielle Maschine i:
Falls Job j dem Job k vorangeht, so ist

$$t_{ik} \geq t_{ij} + p_{ij} \ , \tag{7.5}$$

falls Job k dem Job j vorangeht, so ist

$$t_{ij} \geq t_{ik} + p_{ik} \ . \tag{7.6}$$

Dies ist eine Nebenbedingung in Form einer *Alternative*, die in dieser Form bislang noch nicht aufgetreten ist.
Sie lässt sich durch einen Trick auf lineare Nebenbedingungen zurückführen, allerdings auf Kosten der Einführung von Booleschen Variablen.
Sei

$$\omega \in \mathbb{R}$$

eine obere Schranke für

$$t_{ij} - t_{ik} + p_{ij} \qquad \text{(für alle } i, j, k) \ .$$

Dann ist

$$t_{ij} - t_{ik} + p_{ij} \leq \omega$$

immer erfüllt. Die Bedingung (7.5) ist äquivalent zu

$$t_{ij} - t_{ik} + p_{ij} \leq \omega x_{ijk} \tag{7.7}$$

mit $x_{ijk} = 0$ oder 1, da sie für $x_{ijk} = 1$ von selbst erfüllt ist.
Die Bedingung (7.6) ist äquivalent zu

$$t_{ik} - t_{ij} + p_{ik} \leq \omega(1 - x_{ijk}) \tag{7.8}$$

mit $x_{ijk} = 0$ oder 1, da sie für $x_{ijk} = 0$ von selbst erfüllt ist.

Aus der Alternative ist damit eine Konjunktion geworden, und wir erhalten das Problem:
Minimiere

$$\sum_{j \in J} t_{j(m),j}$$

unter den Nebenbedingungen *(7.4), (7.7), (7.8)* und

$$\begin{array}{rcll} t_{ij} & \geq & 0 & (i = 1, \ldots, m; j \in J) \ , \\ x_{ijk} & \in & \{0,1\} & (i = 1, \ldots, m; j, k \in J) \ ! \end{array}$$

Dies ist ein gemischt-ganzzahliges Optimierungsproblem mit *sehr vielen* Nebenbedingungen und Entscheidungsvariablen. ∎

7.2 Ganzzahlige lineare Optimierungsprobleme

Wir betrachten zunächst den linearen Fall.

Problem 7.2.1 **(Ganzzahliges lineares Optimierungsproblem)**
Gegeben seien $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$.
Das **ganzzahlige lineare Optimierungsproblem** *lautet:*

$$\textit{Minimiere} \quad c^\top x \quad \textit{u.d.N.} \quad Ax = b, \quad x \geq 0, \quad x \in \mathbb{Z}^n. \tag{7.9}$$

Das folgende Beispiel aus [NM02] verdeutlicht, dass man die Ganzzahligkeitsforderung nicht einfach durch **Rundung** einer kontinuierlichen Lösung auf den nächsten ganzzahligen Vektor bzw. ganzzahligen zulässigen Vektor umgehen kann.

Beispiel 7.2.2 **(erfolgloses Runden)**
Betrachte das ganzzahlige lineare Optimierungsproblem:

Minimiere $-2x_1 - 3x_2$ unter den Nebenbedingungen

$$\begin{array}{rcl} x_1 + 2x_2 & \leq & 8, \\ 2x_1 + x_2 & \leq & 9, \\ x_1, x_2 & \geq & 0, \\ x_1, x_2 & \in & \mathbb{Z} \ . \end{array}$$

Der zulässige Bereich ist gegeben durch die schwarzen Punkte in der Abbildung 7.1.

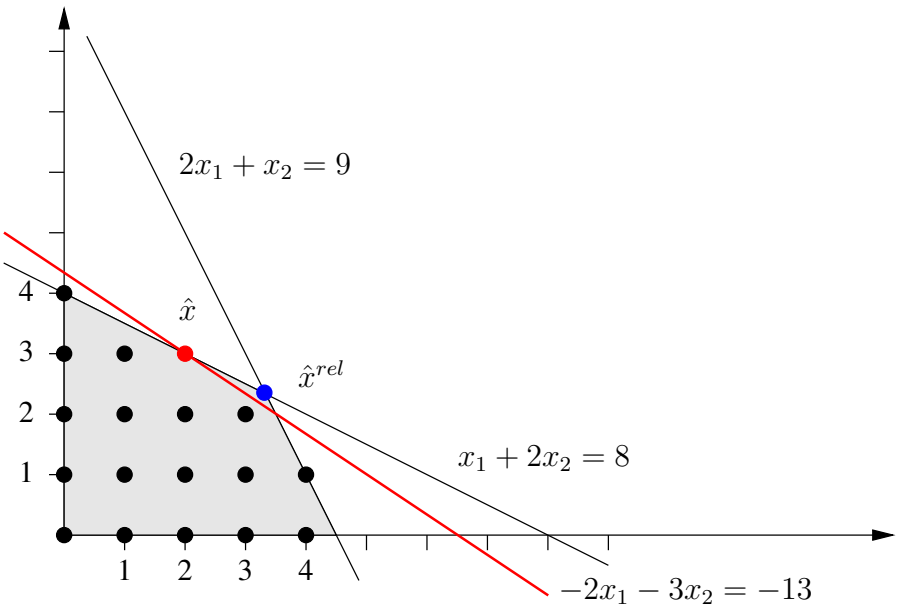


Abbildung 7.1: Erfolgloses Runden

Die optimale Lösung des ganzzahligen Problems ist durch den Punkt $\hat{x} = (2,3)^\top$ *mit Zielfunktionswert* -13 *gegeben. Der Punkt* $\hat{x}^{rel} = (10/3, 7/3)^\top$ *mit Zielfunktionswert* $-13\frac{2}{3}$ *ist die Lösung des reellen linearen Optimierungsproblems, bei dem die Forderung* $x_1, x_2 \in \mathbb{Z}$ *durch* $x_1, x_2 \in \mathbb{R}$ *ersetzt wird. Das resultierende reelle Problem heißt auch* **Relaxation des ganzzahligen Problems** *oder* **relaxiertes ganzzahliges Problem**. *Der schattierte Bereich beschreibt den zulässigen Bereich des relaxierten Problems. Wird der für das ganzzahlige Problem unzulässige Punkt* $\hat{x}^{rel}$ *auf den nächstgelegenen zulässigen ganzzahligen Punkt gerundet, so resultiert der Punkt* $x = (3,2)^\top$ *mit Zielfunktionswert* -12. *Dieser Punkt ist jedoch nicht optimal für das ganzzahlige Optimierungsproblem.*

Naives komponentenweises Runden von nichtganzzahligen Lösungen des relaxierten Problems auf den nächstgelegenen zulässigen ganzzahligen Punkt führt also im Allgemeinen nicht zum Erfolg! ∎

Im folgenden werden wir sogenannte **Relaxierungen** des ganzzahligen Problems 7.2.1 betrachten. Allgemein entsteht ein relaxiertes Problem durch das Weglassen von Beschränkungen. Im konkreten Fall lassen wir die Nebenbedingung $x \in \mathbb{Z}^n$ weg und erhalten das folgende relaxierte ganzzahlige lineare Optimierungsproblem, welches ein lineares Optimierungsproblem in primaler Normalform ist.

Problem 7.2.3 **(Relaxiertes ganzzahliges lineares Optimierungsproblem)**
Gegeben seien $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$.

Das **relaxierte ganzzahlige lineare Optimierungsproblem** *lautet:*

$$\text{Minimiere} \quad c^\top x \quad \text{u.d.N.} \quad Ax = b, \quad x \geq 0. \tag{7.10}$$

Da der zulässige Bereich des relaxierten Problems größer ist als der zulässige Bereich des ganzzahligen Ausgangsproblems (es wurden schließlich Nebenbedingungen weggelassen), liefert jede optimale Lösung des relaxierten Problems eine untere Schranke für den optimalen Funktionswert des Ausgangsproblems. Es gilt:

Satz 7.2.4

Sei $\hat{x}$ optimale Lösung von Problem 7.2.1 und $\hat{x}^{rel}$ optimale Lösung des relaxierten Problems 7.2.3. Dann gilt

$$c^\top \hat{x}^{rel} \leq c^\top \hat{x}.$$

Bemerkung 7.2.5

Es gibt ganzzahlige lineare Optimierungsprobleme, deren Ecken des zulässigen Bereichs stets ganzzahlig sind. Dies ist der Fall, wenn A eine sogenannte total unimodulare Matrix ist und der Vektor b ganzzahlig ist. Eine Matrix heißt dabei total unimodular, wenn die Determinante jeder quadratischen Teilmatrix nur die Werte $0, 1$ oder -1 hat. Solche Matrizen treten in vielen Netzwerkoptimierungsproblemen wie Transportproblemen, Kürzeste-Wege-Problemen oder Maximalflussproblemen auf.

Da das Simplexverfahren – im Falle der Existenz einer optimalen Lösung und unter der üblichen Rangbedingung – stets in einer optimalen Ecke des zulässigen Bereichs terminiert, liefert es dann bei Anwendung auf die Relaxierung eines solchen solchen ganzzahligen linearen Problems stets automatisch eine ganzzahlige Lösung. ∎

7.3 Schnittebenenverfahren nach Gomory

In diesem Abschnitt stellen wir eine Pionierleistung auf dem Gebiet der ganzzahligen linearen Optimierung vor, das Schnittebenenverfahren von Gomory [Gom58, Gom60, Gom63]. Dabei wird die ganzzahlige zulässige Menge reell relaxiert und diese vergrößerte Menge durch „Gomory-Cuts“ sukzessive wieder beschnitten, ohne dass eine ganzzahlige zulässige Lösung verloren geht. Der schematische Ablauf lautet wie folgt:

Algorithmus 7.3.1 (Schnittebenenverfahren nach Gomory)

(1) Löse (mit dem Simplexverfahren) das relaxierte Problem 7.2.3, genannt (R), des ganzzahligen Optimierungsproblems 7.2.1.

Ist die berechnete Lösung x^{rel} ganzzahlig, so ist x^{rel} auch optimal für das Ausgangsproblem. STOP!

Besitzt das relaxierte Problem keine Lösung, so besitzt auch das Ausgangsproblem keine Lösung. STOP!

(2) Konstruiere eine zusätzliche Nebenbedingung mit den folgenden Eigenschaften:

(i) x^{rel} erfüllt die neue Nebenbedingung **nicht**, *d.h. x^{rel} wird durch die neue Nebenbedingung* **abgeschnitten**.

(ii) Jede zulässige Lösung des ganzzahligen Optimierungsproblems 7.2.1 erfüllt die zusätzliche Nebenbedingung.

(3) Füge die neue Nebenbedingung dem Problem (R) hinzu. Das vergrößerte Problem sei (R'). Ersetze (R) durch (R') und gehe zu (1).

Das relaxierte Problem kann in Schritt (1) des Algorithmus mit dem Simplexverfahren gelöst werden. Besitzt es eine Lösung, so liefert das Simplexverfahren die optimale Basislösung

$$x_B^{rel} = A_B^{-1}b - A_B^{-1}A_N x_N^{rel} = \beta_B - \Gamma_B^N x_N^{rel}$$

mit Basisindexmenge B und Nichtbasisindexmenge N. Ist x_B^{rel} ganzzahlig, so ist $x_B^{rel} = \beta_B$, $x_N^{rel} = 0$ optimal für das ganzzahlige Optimierungsproblem 7.2.1 und das Verfahren bricht ab.

Konstruktion eines Schnitts:

Gilt $x_B^{rel} = \beta_B \notin \mathbb{Z}^m$, so muß in Schritt (2) des Algorithmus eine Nebenbedingung konstruiert werden, die die Lösung x^{rel} abschneidet.

Es sei $i \in B$ ein beliebiger Index mit $\beta_i \notin \mathbb{Z}$. Im folgenden bezeichne $\lfloor a \rfloor$ die größte ganze Zahl kleiner oder gleich a, insbesondere gilt dann $\lfloor a \rfloor \leq a$.

Zunächst betrachten wir einen beliebigen zulässigen Punkt x des ganzzahligen Optimierungsproblems 7.2.1. Mit $\Gamma_B^N = (\gamma_{ij})_{i\in B, j\in N}$ und den zu x^{rel} gehörenden Indexmengen B und N folgt

$$x_i = \beta_i - \sum_{j\in N} \gamma_{ij} x_j$$

bzw.

$$\beta_i = x_i + \sum_{j \in N} \gamma_{ij} x_j. \tag{7.11}$$

Wegen $x_j \geq 0$ und $\lfloor \gamma_{ij} \rfloor \leq \gamma_{ij}$ folgt

$$\beta_i \geq x_i + \sum_{j \in N} \lfloor \gamma_{ij} \rfloor x_j. \tag{7.12}$$

Da die Komponenten x_i, $i \in B$, und x_j, $j \in N$, von x ganzzahlig sind, gilt

$$x_i + \sum_{j \in N} \lfloor \gamma_{ij} \rfloor x_j \in \mathbb{Z}.$$

Damit erfüllt jeder zulässige Punkt x des ganzzahligen Problems sogar die im Vergleich zu (7.12) schärfere Ungleichung

$$\lfloor \beta_i \rfloor \geq x_i + \sum_{j \in N} \lfloor \gamma_{ij} \rfloor x_j. \tag{7.13}$$

Wegen (7.11) und (7.13) erfüllt jeder zulässige Punkt x des ganzzahligen Problems die Ungleichung

$$\lfloor \beta_i \rfloor - \beta_i \geq \sum_{j \in N} (\lfloor \gamma_{ij} \rfloor - \gamma_{ij}) x_j. \tag{7.14}$$

Die aktuelle, nichtganzzahlige Basislösung $x_B^{rel} = \beta_B$ erfüllt diese Ungleichung wegen $\beta_i \notin \mathbb{Z}$ nicht, denn es gilt $\lfloor \beta_i \rfloor - \beta_i < 0$ und $x_j^{rel} = 0$, $j \in N$. Daher wird die optimale Lösung x^{rel} des relaxierten Problems durch Hinzunahme der Ungleichung (7.14) wie gewünscht „abgeschnitten“.
Damit können wir Schritt (2) in Algorithmus 7.3.1 präzisieren:

Schritt (2): Konstruktion eines Schnitts
Es bezeichne $x_B^{rel} = \beta_B$, $x_N^{rel} = 0$ die optimale Lösung des relaxierten Problems mit Basisindexmenge B und Nichtbasisindexmenge N. Wähle einen Index $i \in B$ mit $\beta_i \notin \mathbb{Z}$. Führe die zusätzlichen Nebenbedingungen

$$\begin{aligned} x_{n+1} + \sum_{j \in N} (\lfloor \gamma_{ij} \rfloor - \gamma_{ij}) x_j &= \lfloor \beta_i \rfloor - \beta_i, \\ x_{n+1} &\geq 0 \end{aligned}$$

ein (x_{n+1} ist eine Schlupfvariable für (7.14)).

Beispiel 7.3.2 (vgl. Beispiel 7.2.2)
Betrachte das folgende ganzzahlige Optimierungsproblem (x_3, x_4 sind Schlupfvariable):

Minimiere $-2x_1 - 3x_2$ unter den Nebenbedingungen

$$\begin{aligned} x_1 + 2x_2 + x_3 &= 8, \\ 2x_1 + x_2 + x_4 &= 9, \quad x_i \geq 0,\ x_i \in \mathbb{Z},\ i = 1,2,3,4. \end{aligned}$$

Lösung des relaxierten Problems mit dem Simplexverfahren liefert folgende Ausgabe:
Starttableau:

	x_1	x_2	
x_3	*1*	*2*	*8*
x_4	2	*1*	*9*
	2	*3*	*0*

Tableau 1:

	x_4	x_2	
x_3	$-\frac{1}{2}$	$\frac{3}{2}$	$\frac{7}{2}$
x_1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{9}{2}$
	-1	*2*	*-9*

Tableau 2:

	x_4	x_3	
x_2	$-\frac{1}{3}$	$\frac{2}{3}$	$\frac{7}{3}$
x_1	$\frac{2}{3}$	$-\frac{1}{3}$	$\frac{10}{3}$
	$-\frac{1}{3}$	$-\frac{4}{3}$	$-\frac{41}{3}$

Offenbar ist die Komponente $x_1 = \frac{10}{3}$ nicht ganzzahlig. Wir wählen also den Index $1 \in B = \{2,1\}$ zur Konstruktion eines Schnitts. Die Ungleichung (7.14) lautet (mit $\beta_1 = 10/3$, $\gamma_{14} = 2/3$, $\gamma_{13} = -1/3$):

$$-\frac{1}{3} = \left\lfloor \frac{10}{3} \right\rfloor - \frac{10}{3} \geq \left(\left\lfloor \frac{2}{3} \right\rfloor - \frac{2}{3} \right) x_4 + \left(\left\lfloor -\frac{1}{3} \right\rfloor + \frac{1}{3} \right) x_3 = -\frac{2}{3} x_4 - \frac{2}{3} x_3$$

Wir wollen die zur Gerade

$$-\frac{2}{3} x_3 - \frac{2}{3} x_4 = -\frac{1}{3} \tag{7.15}$$

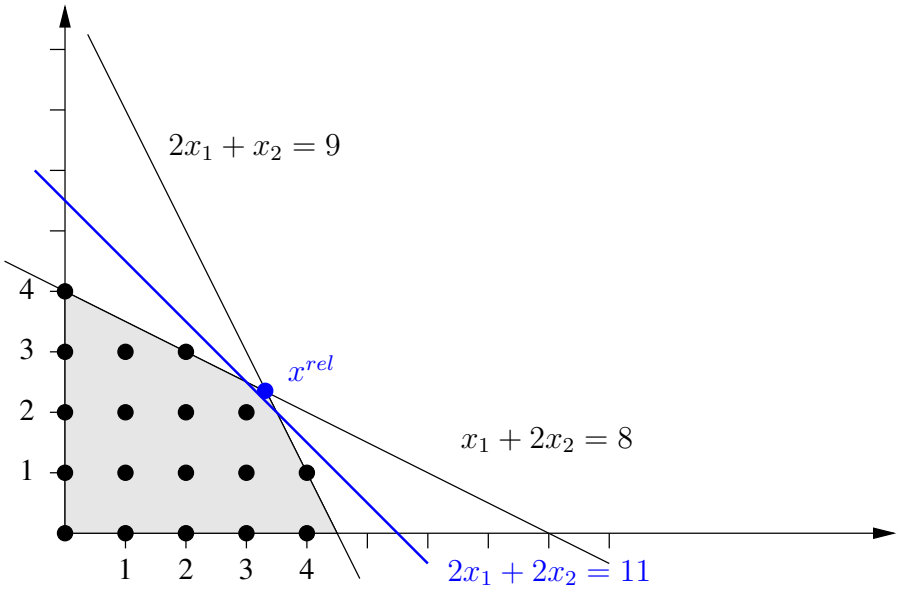
gehörenden Punkte in der (x_1, x_2)-Ebene veranschaulichen. Aus den ursprünglichen Nebenbedingungen erhalten wir

$$\begin{aligned} x_3 &= 8 - x_1 - 2x_2, \\ x_4 &= 9 - 2x_1 - x_2. \end{aligned}$$

Einsetzen in (7.15) liefert die Gerade

$$2x_1 + 2x_2 = 11.$$

Durch die zusätzliche Nebenbedingung (blau) wird die Lösung x^{rel} des relaxierten Problems abgeschnitten:



Im nächsten Schritt des Schnittebenenverfahrens muß das erweiterte Problem gelöst werden:
Minimiere $-2x_1 - 3x_2$ *unter den Nebenbedingungen*

$$\begin{aligned} x_1 + 2x_2 + x_3 &= 8, \\ 2x_1 + x_2 + x_4 &= 9, \\ -\frac{2}{3}x_3 - \frac{2}{3}x_4 + x_5 &= -\frac{1}{3}, \quad x_i \geq 0,\ i = 1, \ldots, 5. \end{aligned}$$

Dieses Problem besitzt die folgende optimale Lösung:

	x_5	x_3	
x_2	$-\frac{1}{2}$	*1*	$\frac{5}{2}$
x_1	*1*	*-1*	*3*
x_4	$-\frac{3}{2}$	*1*	$\frac{1}{2}$
	$-\frac{1}{2}$	*-1*	$-\frac{27}{2}$

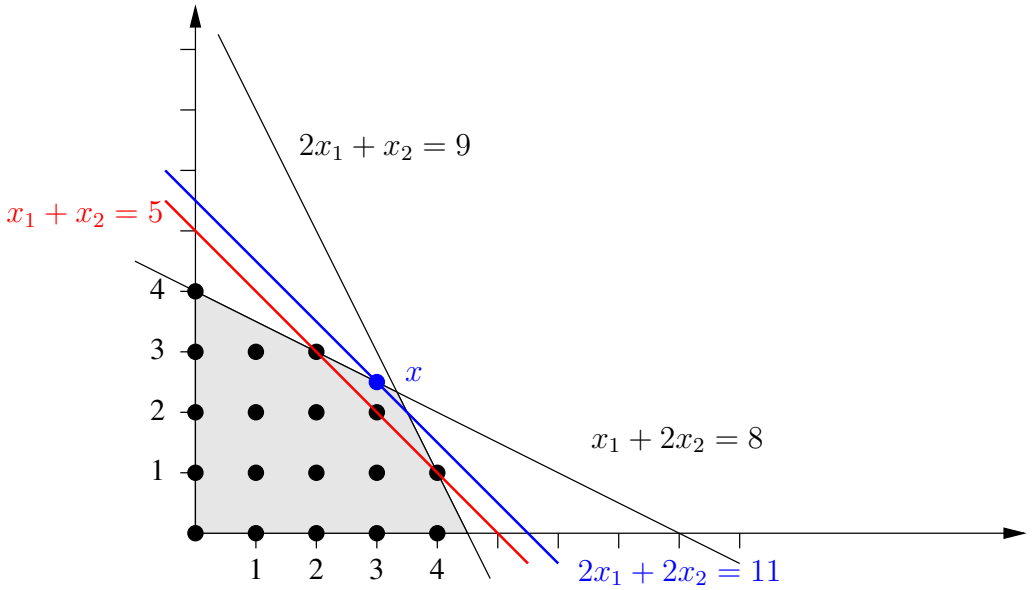
Offenbar ist die Komponente $x_2 = \frac{5}{2}$ *nicht ganzzahlig. Wir wählen also den Index* $2 \in B = \{2, 1, 4\}$ *zur Konstruktion eines weiteren Schnitts. Die Ungleichung (7.14) lautet (mit* $\beta_2 = 5/2$, $\gamma_{25} = -1/2$, $\gamma_{23} = 1$*):*

$$-\frac{1}{2} = \left\lfloor \frac{5}{2} \right\rfloor - \frac{5}{2} \geq \left(\left\lfloor -\frac{1}{2} \right\rfloor + \frac{1}{2} \right) x_5 + (\lfloor 1 \rfloor - 1)\, x_3 = -\frac{1}{2} x_5$$

Die zur Gerade $-1/2 = -1/2x_5$ *gehörenden Punkte in der* (x_1, x_2)*-Ebene sind gegeben durch*

$$x_1 + x_2 = 5.$$

Durch die zusätzliche Nebenbedingung (rot) wird die aktuelle Lösung x *(blauer Punkt) abgeschnitten:*



Im nächsten Schritt des Schnittebenenverfahrens muß das erweiterte Problem gelöst werden:
Minimiere $-2x_1 - 3x_2$ *unter den Nebenbedingungen*

$$\begin{aligned} x_1 + 2x_2 + x_3 &= 8, \\ 2x_1 + x_2 + x_4 &= 9, \\ -\frac{2}{3}x_3 - \frac{2}{3}x_4 + x_5 &= -\frac{1}{3}, \\ -\frac{1}{2}x_5 + x_6 &= -\frac{1}{2}, \quad x_i \geq 0,\ i = 1, \ldots, 6. \end{aligned}$$

Dieses Problem besitzt die folgende optimale Lösung:

	x_6	x_3	
x_2	*-1*	*1*	*3*
x_1	*2*	*-1*	*2*
x_4	*-3*	*1*	*2*
x_5	*-2*	*0*	*1*
	-1	*-1*	*-13*

Diese Lösung ist ganzzahlig und somit optimal für das Ausgangsproblem. ∎

Bemerkung 7.3.3

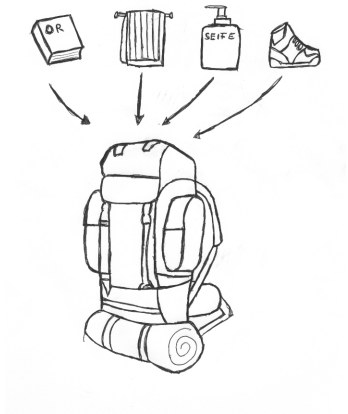
- *Der Nachteil des Schnittebenenverfahrens von Gomory besteht in der wachsenden Dimension der in Schritt (1) zu lösenden Probleme, da mit jeder Nebenbedingung in Schritt (2) eine Ungleichung (bzw. eine Gleichung und eine Schlupfvariable) hinzukommt.*

- *Eine effiziente Implementierung des Verfahrens basiert auf dem dualen Simplexverfahren, bei dem die in Schritt (1) berechneten optimalen Simplex-Tableaus weiter verwendet werden, vgl. Neumann und Morlock [NM02].*

- *Die Endlichkeit des Schnittebenenverfahrens von Gomory wird unter gewissen Voraussetzungen in Gerdts und Lempio [GL11, Hauptsatz 3.3.9] gezeigt. Weitere Aussagen und algorithmische Aspekte finden sich in Wolsey [Wol98]. Hilfreiche Literaturzitate sind in Neumann und Morlock [NM02] auf S. 391 zu finden.*

7.4 Branch&Bound

Eine beliebte Methode zur Lösung linearer *und* nichtlinearer Optimierungsprobleme ist die Branch and Bound-Methode. Wir illustrieren das Prinzip dieser Methode an einem Rucksackpackproblem mit $N = 3$ Gegenständen mit Gewichten $30, 50, 20$ und Werten $3, 5, 2$, vgl. Beispiel 7.1.1.



Die Aufgabe, einen Rucksack mit maximalem Wert und Höchstgewicht 70 zu packen, führt auf ein binäres Optimierungsproblem:

Maximiere

$$3x_1 + 5x_2 + 2x_3$$

unter den Nebenbedingungen

$$30x_1 + 50x_2 + 20x_3 \leq 70, \quad x_i \in \{0, 1\} \quad (i = 1, 2, 3).$$

Es bezeichne

$$S = \{(x_1, x_2, x_3)^\top \ : \ 30x_1 + 50x_2 + 20x_3 \leq 70, \ x_i \in \{0, 1\} \quad (i = 1, 2, 3)\}$$

die zulässige Menge des Problems. Da wir wissen, dass der optimale Wert für x_1 entweder 0 oder 1 ist, können wir das Rucksackpackproblem in zwei kleinere Probleme aufteilen,

indem x_1 auf 0 oder 1 fixiert wird. Dies entspricht einer Partitionierung des zulässigen Bereichs S gemäß

$$S = S_0 \cup S_1, \qquad S_0 \cap S_1 = \emptyset$$

mit

$$\begin{aligned} S_0 &= \{(x_1,x_2,x_3)^\top \,:\, 30x_1+50x_2+20x_3 \le 70,\ x_1=0,\ x_i \in \{0,1\}\ (i=2,3)\}\ , \\ S_1 &= \{(x_1,x_2,x_3)^\top \,:\, 30x_1+50x_2+20x_3 \le 70,\ x_1=1,\ x_i \in \{0,1\}\ (i=2,3)\}\ . \end{aligned}$$

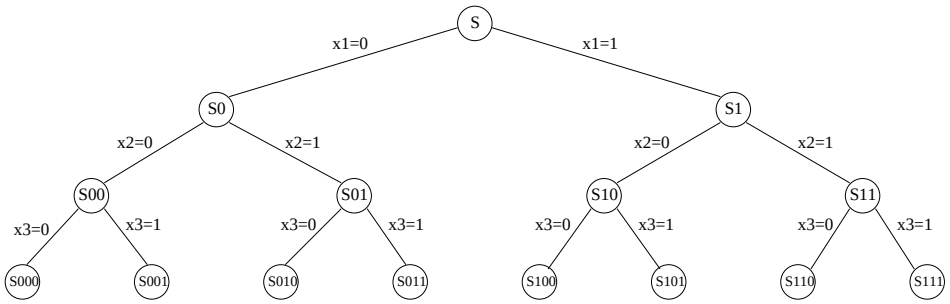
Für jede der Teilmengen S_0 und S_1 können wir analog mit der Komponente x_2 verfahren und erhalten

$$S_0 = S_{00} \cup S_{01} \quad , \quad S_1 = S_{10} \cup S_{11}$$

mit $S_{00} \cap S_{01} = \emptyset$ und $S_{10} \cap S_{11} = \emptyset$ und

$$\begin{aligned} S_{00} &= \{(x_1,x_2,x_3)^\top \,:\, 30x_1+50x_2+20x_3 \le 70,\ x_1=0,\ x_2=0,\ x_3 \in \{0,1\}\}, \\ S_{01} &= \{(x_1,x_2,x_3)^\top \,:\, 30x_1+50x_2+20x_3 \le 70,\ x_1=0,\ x_2=1,\ x_3 \in \{0,1\}\}, \\ S_{10} &= \{(x_1,x_2,x_3)^\top \,:\, 30x_1+50x_2+20x_3 \le 70,\ x_1=1,\ x_2=0,\ x_3 \in \{0,1\}\}, \\ S_{11} &= \{(x_1,x_2,x_3)^\top \,:\, 30x_1+50x_2+20x_3 \le 70,\ x_1=1,\ x_2=1,\ x_3 \in \{0,1\}\}. \end{aligned}$$

Schließlich führt dieselbe Vorgehensweise für die Komponente x_3 auf die folgende Baumstruktur:



Damit wurde eine Partition der zulässigen Menge S gemäß

$$S = S_{000} \cup S_{001} \cup S_{010} \cup S_{011} \cup S_{100} \cup S_{101} \cup S_{110} \cup S_{111}$$

erzeugt. Mit jedem Knoten des Baums kann folgendes Optimierungsproblem assoziiert werden:

$$\textit{Maximiere} \qquad 3x_1+5x_2+2x_3 \qquad \text{u.d.N.} \qquad (x_1,x_2,x_3)^\top \in S_*. \tag{7.16}$$

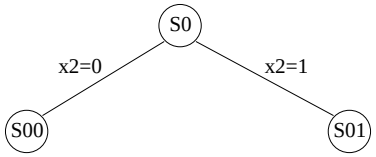
Dabei ist $* \in \{0,1,00,01,10,11,000,001,010,011,100,101,110,111\}$. Beachte, dass die Mengen S_* kleiner werden, je weiter man im Baum absteigt.
Letztendlich sind die Optimierungsprobleme in der unteren Zeile des Baums sehr leicht zu lösen, da in jedem dieser so genannten **Blätter** des Baumes resp. **Blattknoten**

alle Komponenten des Vektors $(x_1, x_2, x_3)^\top$ fixiert sind. Folglich ist die entsprechende zulässige Menge entweder leer, oder sie enthält genau ein Element. Zum Beispiel gilt $S_{101} = \{(1, 0, 1)^\top\}$ mit Zielfunktionswert 5, während S_{111} leer ist. Durch Untersuchen aller acht Blattknoten des Baums findet man leicht die optimale Lösung.
Leider wird der Baum sehr groß für große Werte von N, und diese komplette Verzweigungsstrategie (**Branching**) ist zu zeitaufwändig. Hier kommt nun das **Bounding** ins Spiel. Es dient dazu, Teilbäume abzuschneiden, die keine Optimallösung enthalten können.
Wir untersuchen exemplarisch einige Situationen, in denen es keinen Sinn macht, einen Teilbaum weiter zu expandieren:

(i) **Unzulässigkeit:** Betrachte den Knoten S_{11} für das Rucksackpackproblem. Es gilt $S_{11} = \emptyset$, da $x_1 = x_2 = 1$ und $30 + 50 = 80 \not\leq 70$. Da der zulässige Bereich der Nachfolgeknoten weiter eingeschränkt wird, ist es unnötig, die Nachfolger S_{110} und S_{111} zu untersuchen. Also können Teilbäume unterhalb unzulässiger Knoten abgeschnitten werden.

(ii) **Optimale Knoten:** Falls das einem Knoten zugeordnete Optimierungsproblem gelöst werden kann, so ist es unnötig, die Nachfolgeknoten zu untersuchen, und der Teilbaum unterhalb dieses Knotens kann abgeschnitten werden.

(iii) **Bounding:**
Betrachte den folgenden Teilbaum des Rucksackpackproblems:



Man sieht leicht, dass der optimale Zielfunktionswert für den Knoten S_{00} zwischen 0 (falls $x_3 = 0$) und 2 (falls $x_3 = 1$) liegt. Entsprechend liegt der optimale Zielfunktionswert des Knotens S_{01} zwischen 5 (falls $x_3 = 0$) und 7 (falls $x_3 = 1$). Da wir die Zielfunktion maximieren möchten, ist klar, dass das Optimum nicht in dem Teilbaum unterhalb S_{00} liegen kann, da die Schranken für den Knoten S_{01} einen höheren Zielfunktionswert garantieren. Also ist es nicht notwendig, den Teilbaum unterhalb S_{00} zu expandieren.

Die obigen Abschneideregeln können dabei helfen, den Suchbaum drastisch zu reduzieren. Dies hängt insbesondere auch davon ab, wie untere und obere Schranken berechnet werden und wie gut sie sind. Die folgenden Ansätze sind nützlich:

- Jeder zulässige Punkt liefert eine untere (bzw. obere) Schranke des Optimalwerts eines Maximierungsproblems (bzw. Minimierungsproblems).

- Jede Optimallösung der reellen Relaxation liefert eine obere (bzw. untere) Schranke des Optimalwerts eines ganzzahligen Maximierungsproblems (bzw. Minimierungsproblems).

- Jede zulässige Lösung eines geeignet gewählten Dualproblems liefert eine obere (bzw. untere) Schranke des Optimalwerts eines Maximierungsproblems (bzw. Minimierungsproblems).

Zusammenfassend besitzt eine allgemeine Branch and Bound-Methode die folgenden wesentlichen Komponenten:

- Die **Verzweigungsregel (Branching-Regel)** definiert, wie neue Knoten im Baum erzeugt werden. Sie legt fest, wie der zulässige Bereich partitioniert wird.

- Die **Schrankenregel (Bounding-Regel)** bestimmt untere und obere Schranken des optimalen Zielfunktionswerts.

- Die **Traversierungsregel** definiert die Reihenfolge, in der die Knoten des Baumes durchlaufen werden, z.B. Tiefensuche oder Breitensuche.

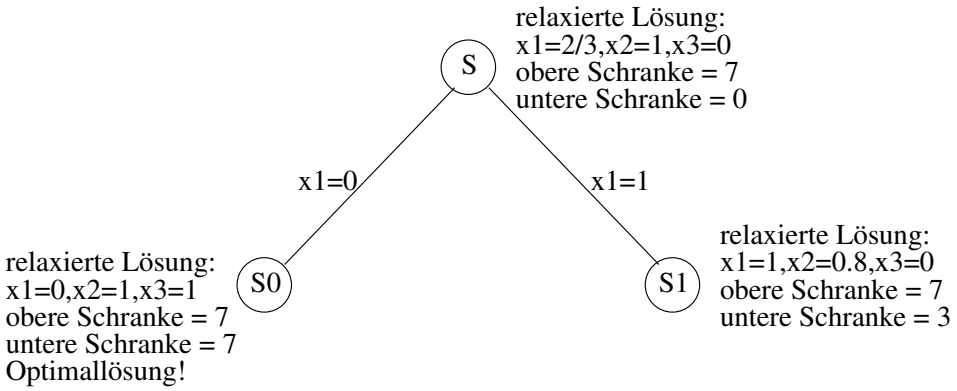
- Die **Abschneideregeln** definieren, wann ein Knoten bzw. seine Nachfolgeknoten nicht mehr weiter untersucht werden müssen.

Beispiel 7.4.1 (Rucksackpackproblem)
Wir wenden die Branch and Bound-Methode auf das Rucksackpackproblem an:

Maximiere $3x_1 + 5x_2 + 2x_3$ unter den Nebenbedingungen

$$30x_1 + 50x_2 + 20x_3 \leq 70, \quad x_i \in \{0, 1\} \ (i = 1, 2, 3).$$

Für jeden Knoten wird eine obere Schranke des Optimalwerts durch Lösen der reellen Relaxierung berechnet. Untere Schranken sind durch zulässige Punkte gegeben. Verzweigt wird an der Komponente $x_i^{rel} \notin \{0, 1\}$ mit dem kleinsten Index i, wobei x^{rel} die optimale Lösung der rellen Relaxierung bezeichnet.



In der Abbildung wurden die Knoten in der Reihenfolge S, S_1, S_0 generiert. Da die ILP Relaxierung von S_0 eine ganzzahlige Lösung mit Zielfunktionswert 7 hat, ist es nicht notwendig, den Teilbaum unter S_1 zu betrachten, da der Zielfunktionswert von S_1 zwischen 3 und 7 liegt. Dieser Teilbaum entält also keine bessere Lösung. ■

Wir wenden die Branch and Bound-Methode auf das ganzzahlige lineare Optimierungsproblem 7.2.1 an.

Verzweigung (Branching) für Problem 7.2.1:
Betrachte einen Knoten des Branch and Bound-Suchbaums (z.B. den Wurzelknoten). Das dem Knoten zugeordnete Optimierungsproblem sei:

$$\textit{Minimiere} \quad c^\top x \quad \text{u.d.N.} \quad Ax = b,\ x \geq 0,\ x \in \mathbb{Z}^n.$$

Sei x^{rel} die Lösung des relaxierten Problems

$$\textit{Minimiere} \quad c^\top x \quad \text{u.d.N.} \quad Ax = b,\ x \geq 0,\ x \in \mathbb{R}^n.$$

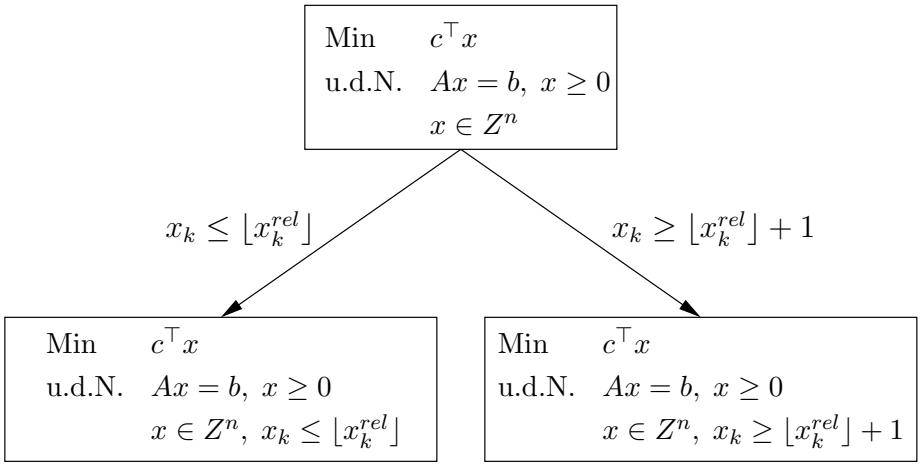
Falls x^{rel} zulässig für Problem 7.2.1 ist, so ist kein weiteres Verzweigen nötig. Andernfalls gilt $x^{rel} \notin \mathbb{Z}^n$, und die Verzweigungsregel besteht aus zwei Schritten:

(i) Wähle einen Index k mit $x_k^{rel} \notin \mathbb{Z}$.

(ii) Erzeuge Nachfolgeknoten S_L und S_R (linker bzw. rechter Nachfolger):

$$(S_L) \quad \textit{Minimiere } c^\top x \quad \text{u.d.N.} \quad Ax = b,\ x \geq 0,\ x \in \mathbb{Z}^n,\ x_k \leq \lfloor x_k^{rel} \rfloor.$$

$$(S_R) \quad \textit{Minimiere } c^\top x \quad \text{u.d.N.} \quad Ax = b,\ x \geq 0,\ x \in \mathbb{Z}^n,\ x_k \geq \lfloor x_k^{rel} \rfloor + 1.$$



Eine rekursive Anwendung dieser Verzweigungsregel, beginnend mit Problem 7.2.1, erzeugt eine Baumstruktur. Jede Kante entspricht einer zusätzlichen Nebenbedingung. Die Vereinigung der zulässigen Mengen von (S_L) und (S_R) ergibt gerade die zulässige Menge des Vorgängerknotens.

Schranken (Bounding):
Betrachte einen Knoten und das zugehörige ganzzahlige lineare Optimierungsproblem

$$\textit{Minimiere} \quad c^\top x \quad \text{u.d.N.} \quad Ax = b,\ x \geq 0,\ x \in \mathbb{Z}^n.$$

Es bezeichne w den Minimalwert dieses Problems. Eine untere und eine obere Schranke für w können wie folgt bestimmt werden:

- **Obere Schranke:**
  Für jedes zulässige x mit $Ax = b$, $x \geq 0$, $x \in \mathbb{Z}^n$ gilt

  $$w \leq c^\top x.$$

  Also liefert jeder zulässige Punkt eine obere Schranke $u = c^\top x$ von w.

- **Untere Schranke:**
  Es bezeichne $\tilde{w}$ den Minimalwert des relaxierten Problems

  $$\textit{Minimiere} \quad c^\top x \quad \text{u.d.N.} \quad Ax = b,\ x \geq 0.$$

  Dann gilt

  $$\tilde{w} \leq w\ .$$

Bemerkung 7.4.2

Eine untere Schranke kann alternativ auch durch Lösen des dualen Problems

$$\textit{Maximiere} \quad b^\top y \qquad \textit{u.d.N.} \qquad A^\top y \leq c$$

des relaxierten Problems ermittelt werden, da nach dem schwachen Dualitätssatz

$$b^\top y \leq \tilde{w} \leq w$$

für alle dual zulässigen Punkte y gilt. ∎

Abschneideregeln:
Ein Knoten heißt **untersucht** (oder **ausgelotet**) und ein weiteres Verzweigen für diesen Knoten ist nicht nötig, falls eine der folgenden Bedingungen gilt:

- Das relaxierte Problem ist unzulässig (besitzt keine Lösung).
- Das relaxierte Problem besitzt eine ganzzahlige Lösung. Da diese Lösung auch zulässig für das ursprüngliche ganzzahlige Problem 7.2.1 ist, liefert der zugehörige Zielfunktionswert eine obere Schranke für den optimalen Zielfunktionswert von Problem 7.2.1.
- Die untere Schranke des optimalen Zielfunktionswerts des Knotens ist größer oder gleich der aktuellen oberen Schranke u.

Insgesamt erhalten wir den folgenden Algorithmus.

Algorithmus 7.4.3 **(Branch&Bound Algorithmus)**
Initialisierung:

- *Sei u eine obere Schranke des Optimalwerts (falls keine bessere bekannt ist, kann $u = \infty$ gewählt werden).*
- *Sei Problem 7.2.1 gekennzeichnet als aktiver Knoten.*

while *(es gibt aktive Knoten)* **do**

Wähle einen aktiven Knoten v gemäß Traversierungsregel.

Berechne (falls möglich) eine optimale Lösung $\tilde{x}$ der v-Relaxierung und setze $\tilde{w} := c^\top \tilde{x}$.

Ist die Zielfunktion auf der relaxierten zulässigen Menge nicht nach unten beschränkt, so setze $\tilde{w} = -\infty$.

if *(v unzulässig)* **then** *markiere Knoten v als untersucht.* **endif**

if *($\tilde{x} \in \mathbb{Z}^n$)* **then**

if *($\tilde{w} < u$)* **then**

Speichere $\tilde{x}$ als aktuell beste Lösung und setze $u = \tilde{w}$.

endif

Markiere Knoten v als untersucht.

endif

if *($\tilde{w} \geq u$)* **then** *Markiere Knoten v als untersucht.* **endif**

if *($\tilde{w} < u$)* **then**

Wende Verzweigungsregel auf v an, markiere alle Nachfolger als aktiv und markiere den aktuellen Knoten v als inaktiv.

endif

end

Die Größe des im Branch and Bound-Verfahren erzeugten Baums hängt von den Daten des Problems ab und kann auch sehr groß werden. In der Praxis muss der Algorithmus mitunter abgebrochen werden, ohne dass eine optimale Lösung gefunden wurde. Allerdings hat das Verfahren den Vorteil, dass stets untere und obere Schranken des optimalen Zielfunktionswerts verfügbar sind. Die obere Schranke ist durch das aktuelle u gegeben. Eine untere Schranke ℓ ist durch das Minimum aller unteren Schranken aller aktiven Knoten gegeben. u und ℓ können ebenfalls benutzt werden, um ein Abbruchkriterium zu definieren, etwa $u - \ell < \text{tol}$ für eine vom Benutzer vorgegebene Toleranz $\text{tol} > 0$.

Das Branch and Bound-Verfahren ist ein sehr flexibler Algorithmus, der in der Praxis sehr häufig eingesetzt wird. Es gibt viele Varianten und Modifikationen, so können untere Schranken, die etwa durch Relaxierung berechnet wurden, weiter verbessert werden, indem einige Gomory-Schnitte durchgeführt werden. Dies verbessert die Qualität der Schranken und führt in der Regel zu kleineren Suchbäumen. Eine solche Variante wird häufig als Branch and Cut-Methode bezeichnet.

Falls der zulässige Bereich von Problem 7.2.1 unbeschränkt ist, gibt es keine Garantie, dass der Branch and Bound-Algorithmus endet. Betrachte dazu das Problem

$$\textit{Maximiere} \quad x + y \quad \text{u.d.N.} \quad 2x + 2y \leq 1, \quad x, y \in \mathbb{Z}.$$

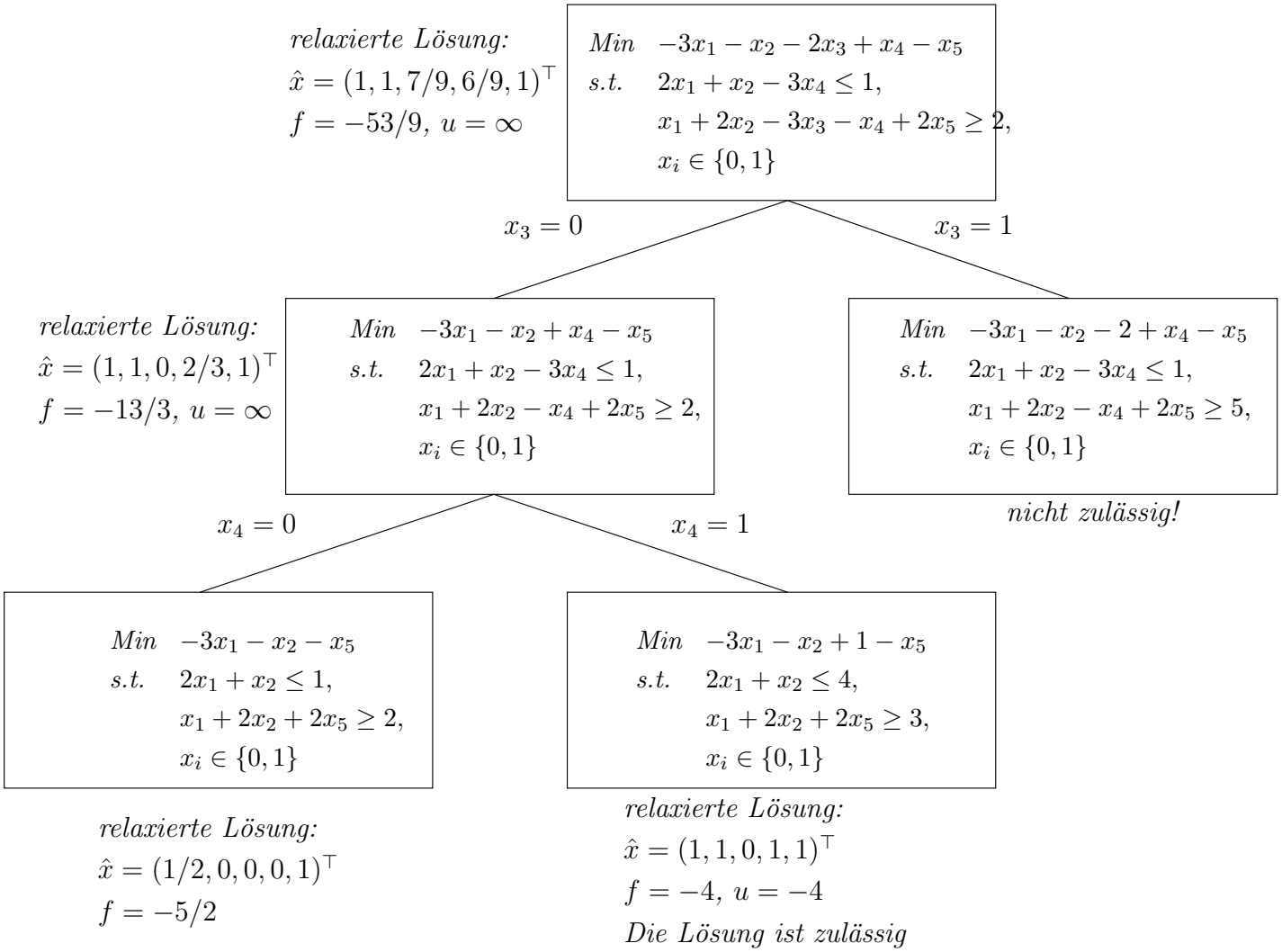
Das auf den ersten Blick sehr einfache Problem besitzt unendlich viele Lösungen mit $x + y = 0$, und das Branch and Bound-Verfahren terminiert nicht.

Beispiel 7.4.4

Gegeben sei das binäre Optimierungsproblem

$$\begin{aligned} &\textit{Maximiere} && 3x_1 + x_2 + 2x_3 - x_4 + x_5 \\ &\textit{unter} && \begin{array}{rcrcrcrcl} 2x_1 & + & x_2 & & - & 3x_4 & & \leq & 1, \\ x_1 & + & 2x_2 & - & 3x_3 & - & x_4 & + & 2x_5 \geq 2, \end{array} \\ & && x_1, x_2, x_3, x_4, x_5 \in \{0, 1\}. \end{aligned}$$

Die Anwendung des Branch&Bound-Verfahrens liefert das folgende Ergebnis, wobei die Traversierung des Baums nach dem FIFO-Prinzip erfolgte.



Der rechte Knoten in der untersten Zeile liefert eine Optimallösung für die Fixierungen $x_3 = 0, x_4 = 1$, da die relaxierte Lösung zulässig für das ganzzahlige Teilproblem ist.

Damit bekommt man eine obere Schranke $U = -4$ für den optimalen Zielfunktionswert $\hat{f}$ des Ausgangsproblems, d.h. es ist $\hat{f} \leq u = -4$.
Da für das linke Blatt $f = -5/2 > u = -4$ gilt, braucht dieser Knoten nicht weiter betrachtet werden, da $f = -5/2$ eine untere Schranke für den Zielfunktionswert für diejenigen Zweige darstellt, die von dem linken Blatt ausgehen.
Insgesamt stellt $x_1 = x_2 = x_4 = x_5 = 1, x_3 = 0$ mit Zielfunktionswert $\hat{f} = -4$ für das Minimierungsproblem (bzw. $\hat{f} = 4$ für das Maximierungsproblem) eine Optimallösung dar. Bei den relaxierten Problemen wurde die Forderung $x_i \in \{0, 1\}$ ersetzt durch $x_i \in [0, 1]$.

■

Kapitel 8

Diskrete dynamische Optimierung

Das *dynamische Verhalten* des *Zustands* eines technischen, ökonomischen oder biologischen Systems kann häufig durch (diskrete) dynamische Gleichungen beschrieben werden. Zum Beispiel können wir uns für die Entwicklung der Populationsgröße einer Spezies in einem gewissen Zeitinitervall interessieren. Oder wir möchten das dynamische Verhalten eines chemischen Prozesses oder eines mechanischen Systems oder die Entwicklung des Gewinns einer Firma untersuchen.
In der Regel kann das dynamische Verhalten eines gegebenen Systems durch *Steuervariablen* beeinflusst werden. Beispielsweise kann ein Auto durch Drehen des Lenkrads und durch Betätigen des Gaspedals und der Bremsen gesteuert werden. Ein chemischer Prozess kann häufig durch die Erhöhung oder Verringerung der Temperatur beeinflusst werden. Der Gewinn einer Firma kann durch die Änderung von Produktpreisen oder die Anzahl der Mitarbeiter gesteuert werden.
Im Allgemeinen dürfen die Zustände und die Steuerungen eines Systems nicht jeden Wert annehmen, sondern sie unterliegen gewissen *Beschränkungen*. Diese Beschränkungen können aus Sicherheitsbestimmungen oder physikalischen Begrenzungen resultieren. Z.B. muss die Temperatur in einem Reaktor unterhalb einer gewissen Schwelle liegen, um eine Überhitzung zu vermeiden. Ebenso sollte sich die Flughöhe eines Flugzeugs stets oberhalb der Erdoberfläche befinden. Bei der Steuerung eines Fahrzeugs ist der Lenkwinkel physikalisch beschränkt durch einen maximalen Lenkwinkel.
Wir sind an Zustands- und Steuervariablen interessiert, die zulässig sind, also alle Beschränkungen erfüllen, und die darüber hinaus noch eine gegebene *Zielfunktion* minimieren oder maximieren.
Insgesamt erhalten wir folgende Bestandteile eines diskreten dynamischen Optimierungsproblems, vgl. Abbildungen 8.1, 8.2:

- Die Zustandsvariablen $y(t_j)$ beschreiben den Zustand eines Systems zum Zeitpunkt t_j.
- Die Steuervariablen $u(t_j)$ erlauben es, das dynamische Verhalten des Zustands zum Zeitpunkt t_j zu beeinflussen.
- Die diskrete dynamische Gleichung $y(t_{j+1}) = g(t_j, y(t_j), u(t_j))$ legt den Übergang des Zustands im Zeitpunkt t_j zum Zustand im nachfolgenden Zeitpunkt t_{j+1} fest und

hängt ab vom aktuellen Zeitpunkt, vom aktuellen Zustand und von der aktuellen Steuerung.

- Die Zielfunktion wird minimiert (oder maximiert).
- Die Beschränkungen für den Zustand und die Steuerung müssen eingehalten werden.

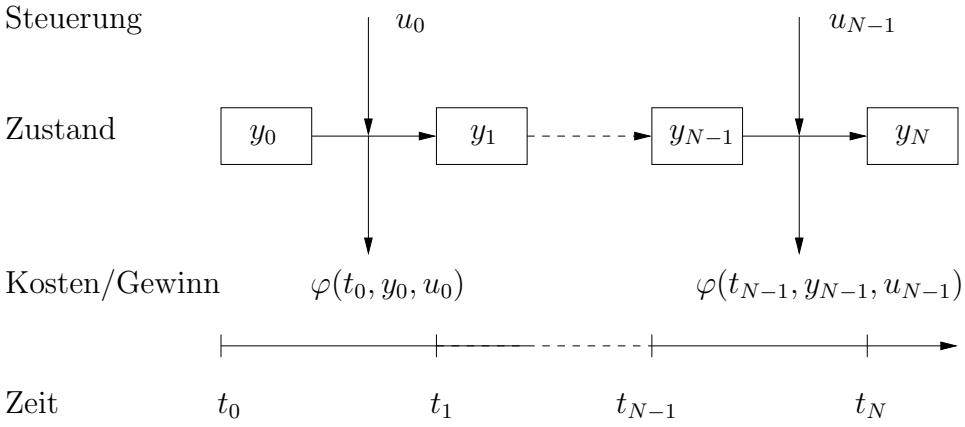


Abbildung 8.1: Schema eines diskreten dynamischen Optimierungsproblems.

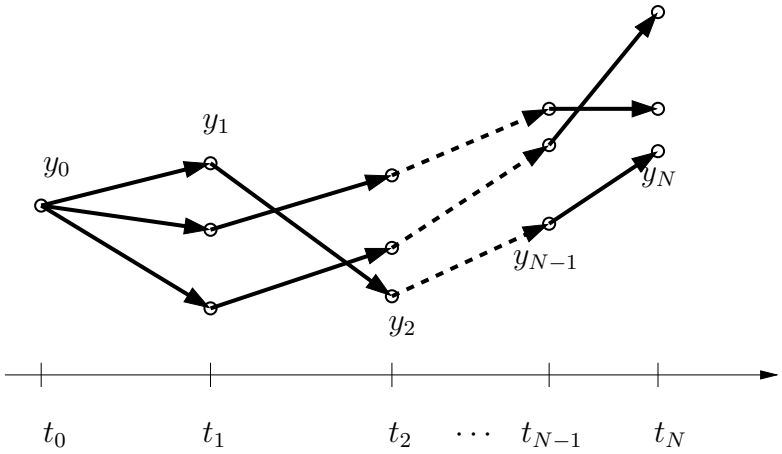


Abbildung 8.2: Diskrete Trajektorien für verschiedene Steuerungen.

Haben die Beschränkungen und die Zielfunktion zusätzlich eine spezielle Struktur, handelt es sich z. B. um punktweise Zustands- und Steuerbeschränkungen und setzt sich die Zielfunktion additiv aus den Kosten für die Übergänge vom Zeitpunkt t_j zum Zeitpunkt t_{j+1} zusammen, so gilt das bekannte Optimalitätsprinzip von Richard Bellman, vgl. Abschnitt 8.2.

Die darauf aufbauende Methode der dynamischen Programmierung stellen wir in Abschnitt 8.3 vor. Sie ist einerseits sehr universell einsetzbar, leidet aber unter dem *„curse of dimensions“* (Fluch der Dimensionen), da die resultierenden Gleichungen nur mit sehr viel Rechenleistung und großem Speicheraufwand zu bewältigen sind. Aus praktischer Sicht bleibt die Methode daher auf niedrigdimensionale Problemstellungen oder Aufgaben mit spezieller Struktur, z. B. mit endlicher Zustandsmenge und endlichem Steuerbereich, beschränkt. Für eine detaillierte Diskussion verweisen wir auf die Monographien [Bel57], [BD71], [BG93], [**?**]. Viele Anwendungen und Beispiele finden sich in [Win04].
Besonders hervorgehoben werden sollte noch, dass die diskrete Zeitvariable t_j keineswegs immer „natürlichen“ Zeitpunkten unserer Anschauung entsprechen muss, sondern auch eine abstrakte Größe sein kann, die lediglich gewisse Niveaus, Rekursionsstufen, Iterationen anordnet.
Lässt sich die Methode der dynamischen Programmierung nicht erfolgreich auf ein diskretes dynamisches Optimierungsproblem anwenden, so helfen die bislang vorgestellten Konzepte und Verfahren der Optimierung weiter, ist doch jedes diskrete dynamische Optimierungsproblem ein zwar sehr großes, dafür aber stark strukturiertes endlichdimensionales Optimierungsproblem.
Die KKT Optimalitätsbedingungen liefern unmittelbar das diskrete Minimumprinzip, wie wir in Abschnitt 8.4 zeigen.

8.1 Problemstellung und Anwendungen

Ein diskretes dynamisches Optimierungsproblem kann folgendermaßen mathematisch modelliert werden.
Sei

$$\mathbb{G} := \{t_j : j = 0, 1, \dots, N\}$$

ein *Gitter* mit $N + 1$ festen Zeitpunkten

$$t_0 < t_1 < \ldots < t_N \ .$$

Zustandstrajektorien werden beschrieben durch *Gitterfunktionen*

$$y : \mathbb{G} \to \mathbb{R}^n, \qquad t_j \mapsto y(t_j) \quad (j = 0, \dots, N) \ .$$

Steuerungen werden ebenfalls beschrieben durch *Gitterfunktionen*

$$u : \mathbb{G} \to \mathbb{R}^m, \qquad t_j \mapsto u(t_j) \quad (j = 0, \dots, N) \ .$$

Die Zielfunktion habe additive Struktur

$$f(y, u) := \sum_{j=0}^{N} \varphi(t_j, y(t_j), u(t_j))$$

mit

$$\varphi : \mathbb{G} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R} \ .$$

Die *dynamischen Gleichungen* lauten

$$y(t_{j+1}) = g(t_j, y(t_j), u(t_j)) \qquad (j = 0, \ldots, N-1)$$

mit

$$g : \mathbb{G} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n.$$

Zusätzlich müssen *Zustandsbeschränkungen*

$$y(t_j) \in Y(t_j)$$

mit nichtleeren Mengen $Y(t_j) \subseteq \mathbb{R}^n$ und *Steuerbeschränkungen*

$$u(t_j) \in U(t_j, y(t_j))$$

mit nichtleeren Mengen $U(t_j, y(t_j)) \subseteq \mathbb{R}^m$ für $y(t_j) \in Y(t_j)$ in allen Punkten des Zeitgitters $\{t_0, \ldots, t_N\}$ eingehalten werden.

Damit erhalten wir das folgende *diskrete dynamische Optimierungsproblem*:

Problem 8.1.1 **(DOP)**

(DOP) Minimiere

$$\sum_{j=0}^{N} \varphi(t_j, y(t_j), u(t_j))$$

auf der Menge aller Gitterfunktionen

$$y : \mathbb{G} \longrightarrow \mathbb{R}^n \ , \ u : \mathbb{G} \longrightarrow \mathbb{R}^m$$

unter den Nebenbedingungen

$$\begin{aligned} y(t_{j+1}) &= g(t_j, y(t_j), u(t_j)) && (j = 0, \ldots, N-1) \ , \\ y(t_j) &\in Y(t_j) && (j = 0, \ldots, N) \ , \\ u(t_j) &\in U(t_j, y(t_j)) && (j = 0, \ldots, N) \ ! \end{aligned}$$

■

Die Mengen $Y(t_j)$ sind oft nur implizit als Lösungsmenge eines Gleichungs- und Ungleichungssystems gegeben, etwa

$$Y(t_j) = \{y \in \mathbb{R}^{n_y} \mid v(t_j, y) \leq 0, \ w(t_j, y) = 0\},$$

und ebenso die Mengen $U(t_j, y(t_j))$. Im einfachsten Fall werden sie durch Vorzeichenbedingungen oder Box-Constraints festgelegt, etwa

$$u(t_j) \in \{u = (u_1, \ldots, u_m)^\top \in \mathbb{R}^m \mid a_j \leq u_j \leq b_j,\ j = 1, \ldots, m\}.$$

Beachte, dass jede Zustandsfunktion $y : \mathbb{G} \longrightarrow \mathbb{R}^n$ mit einem $n(N+1)$-Vektor identifiziert werden kann und jede Steuerung $u : \mathbb{G} \longrightarrow \mathbb{R}^m$ mit einem $m(N+1)$-Vektor.
Der Raum, in dem die zulässige Menge liegt, ist also der

$$\mathbb{R}^{n(N+1)} \times \mathbb{R}^{m(N+1)}.$$

Dies zeigt bereits, dass diskrete dynamische Optimierungsprobleme über feineren Zeitgittern sehr komplex werden können.
Wir betrachten einige Anwendungen.

Beispiel 8.1.2 ([)

Lagerhaltung] Eine Firma möchte für eine feste Anzahl von Zeitpunkten $t_0 < t_1 < \ldots < t_N$ einen Lagerhaltungsplan mit minimalen Kosten erstellen.

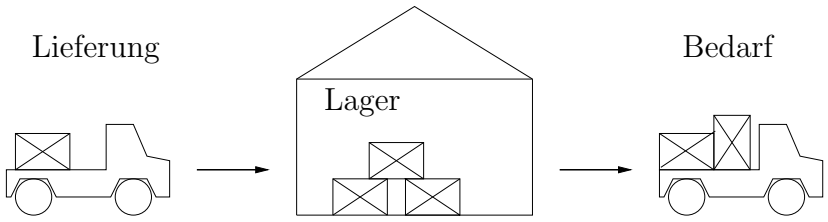


Abbildung 8.3: Lagerhaltungsproblem

Ein Produkt wird in diesen Zeitintervallen $t_0 < t_1 < \ldots < t_N$ in einem Lager gelagert. Es bezeichnen $u_j \geq 0$ die Liefermenge zum Zeitpunkt t_j, $r_j \geq 0$ den Bedarf im Intervall $[t_j, t_{j+1})$ und y_j die Menge des gelagerten Guts zum Zeitpunkt t_j (direkt vor der Lieferung). Dann gelten die Bilanzgleichungen

$$y_{j+1} = y_j + u_j - r_j \qquad (j = 0, 1, \ldots, N-1).$$

Zusätzlich fordern wir, dass der Bedarf stets erfüllt werden muss, d.h. es muss $y_{j+1} \geq 0$ für alle $j = 0, 1, \ldots, N-1$ gelten. Ohne Beschränkung der Allgemeinheit gelte $y_0 = y_N = 0$ für den Bestand zu Beginn und am Ende des Zeithorizonts. Die Lieferkosten zum Zeitpunkt t_j werden modelliert durch

$$B(u_j) = \begin{cases} K + cu_j, & \text{falls } u_j > 0, \\ 0, & \text{falls } u_j = 0, \end{cases}$$

wobei K die Fixkosten und c die Kosten pro Einheit bezeichnen.
Die Lagerhaltungskosten fallen am Ende eines jeden Zeitintervalls an und betragen hy_{j+1} ($j = 0, \dots, N-1$). Die Gesamtkosten sind daher

$$\sum_{j=0}^{N-1} (K\delta(u_j) + cu_j + hy_{j+1}),$$

wobei

$$\delta(u_j) = \begin{cases} 1, & \text{falls } u_j > 0 \ , \\ 0, & \text{falls } u_j = 0 \ . \end{cases}$$

Es gilt

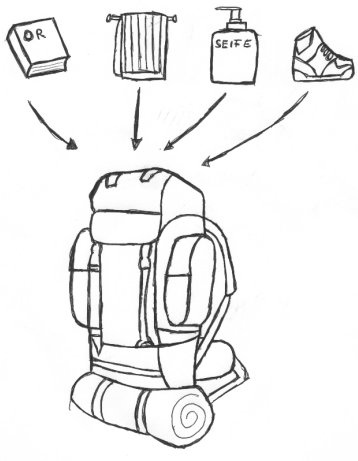
$$\begin{aligned} \sum_{j=0}^{N-1} u_j &= \sum_{j=0}^{N-1} (y_{j+1} - y_j + r_j) = (y_N - y_0) + \sum_{j=0}^{N-1} r_j = \sum_{j=0}^{N-1} r_j \ , \\ \sum_{j=0}^{N-1} y_{j+1} &= \sum_{j=0}^{N-1} (y_j + u_j - r_j) = \sum_{j=0}^{N-1} y_j + \sum_{j=0}^{N-1} u_j - \sum_{j=0}^{N-1} r_j = \sum_{j=0}^{N-1} y_j \ . \end{aligned}$$

Mit diesen Beziehungen können wir das Lagerhaltungsproblem wie folgt formulieren:

$$\begin{array}{lll} \text{Minimiere} & \displaystyle\sum_{j=0}^{N-1} K\delta(u_j) + hy_j & \\ \text{u.d.N.} & y_{j+1} = y_j + u_j - r_j & (j = 0, \dots, N-1) \ , \\ & y_0 = y_N = 0 \ , & \\ & y_j \geq 0 & (j = 1, \dots, N-1) \ , \\ & u_j \geq 0 & (j = 0, \dots, N-1) \ ! \end{array}$$

■

Beispiel 8.1.3 ([])
Rucksackpackproblem]
Maximal N Gegenstände sollen in einen Rucksack gepackt werden.



Jeder Gegenstand $j \in \{1, \dots, N\}$ besitzt das Gewicht a_j und den Wert c_j. Die Aufgabe besteht im Packen eines Rucksacks mit maximalem Wert, wobei das Gewicht des Rucksacks den Wert A nicht überschreiten darf. Dies führt auf das folgende Optimierungsproblem:

$$\text{Maximiere} \quad \sum_{j=1}^{N} c_j u_j \quad \text{u.d.N.} \quad \sum_{j=1}^{N} a_j u_j \leq A,\ u_j \in \{0,1\} \quad (j = 1, \dots, N)\ !$$

Dabei bedeutet

$$u_j = \begin{cases} 1, & \textit{Gegenstand } j \textit{ wird eingepackt,} \\ 0, & \textit{Gegenstand } j \textit{ wird nicht eingepackt.} \end{cases}$$

Ein naiver Ansatz besteht in der Enumeration der höchstens 2^N zulässigen Kombinationen.
Wir interpretieren jetzt die binären Entscheidungsvariablen als Steuerungen u_j und als Zustand y_j das noch verfügbare Restgewicht, nachdem die Gegenstände $i = 1, \dots, j-1$ eingepackt bzw. nicht eingepackt wurden.
Dann wird aus dem ursprünglich gegebenen Problem

Maximiere

$$\sum_{j=1}^{N} c_j u_j$$

unter den Nebenbedingungen

$$\sum_{j=1}^{N} a_j u_j \leq A, \quad u_j \in \{0,1\} \quad (j = 1, \dots, N)\ !$$

das folgende äquivalente diskrete dynamische Optimierungsproblem:

Maximiere

$$\sum_{j=1}^{N} c_j u_j$$

unter den Nebenbedingungen

$$\begin{aligned} & y_{j+1} = y_j - a_j u_j && (j = 1, \dots, N)\ , \\ & y_1 = A, && \\ & y_j \geq 0 && (j = 1, \dots, N)\ , \\ & u_j \begin{cases} \in \{0,1\}, & \textit{falls } y_j \geq a_j\ , \\ = 0, & \textit{falls } y_j < a_j, \end{cases} && (j = 1, \dots, N)\ ! \end{aligned}$$

Dabei ist y_{N+1} das noch verfügbare Restgewicht im Rucksack. ∎

Beispiel 8.1.4 ([])

Zuordnungsprobleme] Eine Anzahl A von Ressourcen soll auf N Projekte verteilt werden.

Die Zuweisung von u_j Ressourcen zu Projekt j resultiere in dem Gewinn $\varphi_j(u_j)$. Die Aufgabe ist, den Gesamtgewinn $\sum_{j=1}^N \varphi_j(u_j)$ zu maximieren. Dies führt auf das folgende Optimierungsproblem:

Maximiere

$$\sum_{j=1}^{N} \varphi_j(u_j)$$

unter den Nebenbedingungen $u_j \in \mathbb{Z}$ und

$$\sum_{j=1}^{N} u_j \leq A,\ u_j \geq 0 \quad (j = 1, \ldots, N)\ !$$

Dieses Problem kann in ein diskretes dynamisches Optimierungsproblem transformiert werden. Dazu führen wir Zustände y_j ein, welche die verbleibenden Ressourcen bezeichnen, nachdem den Projekten $i = 1, \ldots, j-1$ Ressourcen zugewiesen wurden. Dann ist das obige Optimierungsproblem äquivalent zu:

Maximiere

$$\sum_{j=1}^{N} \varphi_j(u_j)$$

unter den Nebenbedingungen

$$\begin{aligned} &y_{j+1} = y_j - u_j && (j = 1, \ldots, N)\ , \\ &u_j \in U_j(y_j) := \{0, 1, \ldots, y_j\} && (j = 1, \ldots, N)\ , \\ &y_1 = A\ ! \end{aligned}$$

■

Beispiel 8.1.5 ([)

Optimalsteuerungsproblem] Eine Rakete der Masse m startet zum Zeitpunkt $t = 0$ aus der Ruhelage auf der Erdoberfläche mit Höhe $h(0) = 0$ und wird senkrecht nach oben geschossen, vgl. Abbildung 8.4.

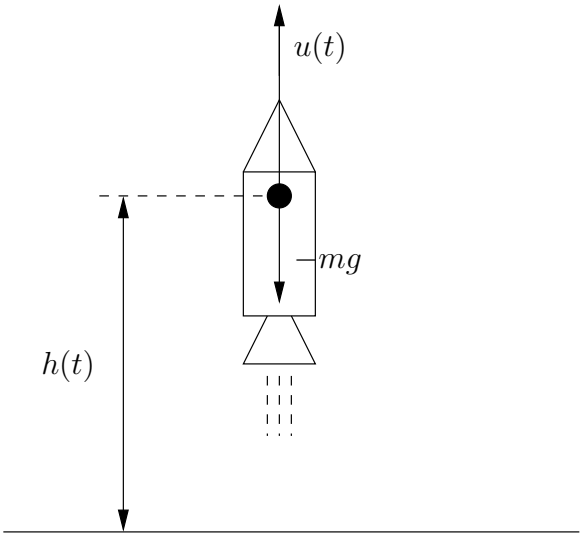


Abbildung 8.4: Das Raketen-Problem: Vertikaler Aufstieg einer Rakete.

Der Pilot kann die Rakete durch die Schubkraft $u(t)$ steuern, wobei die Schubkraft durch eine maximale Schubkraft nach oben und durch Null nach unten (kein Bremsen!) beschränkt ist:

$$0 \le u(t) \le u_{max}, \qquad t \ge 0.$$

Die zeitliche Höhenbewegung der Rakete ist durch das Newton'sche Gesetz

$$\textit{Kraft} = \textit{Masse} \times \textit{Beschleunigung}$$

gegeben:

$$m\ddot{h}(t) = \underbrace{-mg}_{\textit{Erdanziehung}} + \underbrace{u(t)}_{\textit{Schub}}.$$

Zur Vereinfachung nehmen wir an, dass sich die Masse der Rakete zeitlich nicht ändert (was genau genommen nicht stimmt, da Treibstoff verbraucht wird). Desweiteren vernachlässigen wir ebenfalls die Luftwiderstandskräfte.
Die Aufgabe des Piloten ist es nun, mit einer vorgegebenen Menge an Treibstoff eine vorgegebene Höhe H in möglichst kurzer Zeit t_f zu erreichen. Unter der Annahme, dass der Treibstoffverbrauch proportional zur Schubkraft ist, muss der Pilot also eine Nebenbedingung der Form

$$\int_0^{t_f} u(t)dt = \eta$$

beachten, wobei η den Vorrat an Treibstoff bzw. Schubkraft bezeichnet. Unter Beachtung der Anfangs- und Endbedingungen entsteht daraus das folgende Optimalsteuerungsproblem:

Minimiere

$$J(h, v, u, t_f) = t_f$$

unter den Differentialgleichungsnebenbedingungen

$$\begin{aligned} \dot{h}(t) &= v(t), \\ \dot{v}(t) &= -g + \frac{u(t)}{m}, \end{aligned}$$

den Randbedingungen

$$h(0) = 0, \qquad v(0) = 0, \qquad h(t_f) = H,$$

der Integralbeschränkung

$$\int_0^{t_f} u(t)dt = \eta,$$

und der Steuerbeschränkung

$$0 \le u(t) \le u_{max}, \qquad t \in [0, t_f].$$

Die Integralbeschränkung kann wegtransformiert werden: Führe neue Zustandsvariable x mit

$$\dot{x}(t) = u(t), \qquad x(0) = 0$$

ein und fordere $x(t_f) = \eta$. Beachte, dass

$$x(t_f) = x(0) + \int_0^{t_f} \dot{x}(t)dt = \int_0^{t_f} u(t)dt$$

gilt. ∎

Beispiel 8.1.6 ([])

Diskretisierte Optimalsteuerungsprobleme]
Wir zeigen, dass direkte Diskretisierungsverfahren für kontinuierliche Optimalsteuerungsprobleme auf endlichdimensionale diskrete dynamische Optimierungsprobleme führen.

Sei $[a, b] \subset \mathbb{R}$ ein nichtleeres, kompaktes Intervall mit festen Intervallgrenzen $a < b$ und $\mathcal{U} \subseteq \mathbb{R}^m$ eine abgeschlossene Menge.
Desweiteren seien die folgenden (hinreichend glatten) Abbildungen gegeben:

$$\begin{aligned} \varphi &: \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}, \\ f_0 &: [a, b] \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}, \\ f &: [a, b] \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n, \\ r &: \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^p, \\ c &: [a, b] \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^q. \end{aligned}$$

Betrachte das folgende Optimalsteuerungsproblem*:*

(OCP) Minimiere

$$\varphi(x(a),x(b)) + \int_a^b f_0(t,x(t),u(t))dt$$

bezüglich der Funktionen x und u unter Beachtung der *Differentialgleichung*

$$x'(t) = f(t,x(t),u(t)),$$

den *Randbedingungen*

$$r(x(a),x(b)) = 0,$$

den *Steuer- und Zustandsbeschränkungen*

$$c(t,x(t),u(t)) \le 0,$$

und den *Mengenbeschränkungen*

$$u(t) \in \mathcal{U}.$$

Direkte Lösungsverfahren für dieses Optimalsteuerungsproblem basieren auf der Diskretisierung des Integrals, der Differentialgleichung und der Beschränkungen auf dem Gitter

$$\mathbb{G} := \{t_j : j = 0,1,\dots,N\}$$

mit

$$a = t_0 < t_1 < \ldots < t_N = b$$

und Schrittweiten $h_j := t_{j+1} - t_j$ für $j = 0,\dots,N-1$.
Die Differentialgleichung wird auf $\mathbb{G}$ durch ein geeignetes Diskretisierungsverfahren, z.B. das Euler-Verfahren oder ein Runge-Kutta-Verfahren höherer Ordnung, approximiert. Für das Euler-Verfahren erhalten wir

$$x_{j+1} = x_j + h_j f(t_j,x_j,u_j) \quad (j = 0,\dots,N-1)\ ,$$

wobei $x_j \approx x(t_j)$ $(j = 0,\dots,N)$ und $u_j \approx u(t_j)$ $(j = 0,\dots,N)$ Approximationen der Zustandsfunktion $x(\cdot)$ und der Steuerung $u(\cdot)$ auf dem Gitter $\mathbb{G}$ sind. Das Integral in der Zielfunktion wird approximiert durch die Riemann-Summe

$$\int_a^b f_0(t,x(t),u(t))dt \approx \sum_{j=0}^{N-1} h_j f_0(t_j,x_j,u_j).$$

Schließlich wird die Steuer- und Zustandsbeschränkung $c(t, x(t), u(t)) \leq 0$ durch endlich viele Beschränkungen

$$c(t_j, x_j, u_j) \leq 0 \qquad (j = 0, \ldots, N)$$

ersetzt.

Zusammenfassend erhalten wir das diskretisierte Optimalsteuerungsproblem:

(OCP_N) *Minimiere*

$$\varphi(x_0, x_N) + \sum_{j=0}^{N-1} h_j f_0(t_j, x_j, u_j)$$

unter den Nebenbedingungen

$$x_j \in \mathbb{R}^n \ (j{=}0, \ldots, N) \ , \ u_j \in \mathbb{R}^m \ (j = 0, \ldots, N)$$

und

$$x_{j+1} = x_j + h_j f(t_j, x_j, u_j) \quad (j{=}0, \ldots, N{-}1) \ ,$$

$$r(x_0, x_N) = 0,$$

$$c(t_j, x_j, u_j) \leq 0, \quad (j = 0, \ldots, N),$$

$$u_j \in \mathcal{U} \quad (j = 0, \ldots, N) \ !$$

■

8.2 Das Optimalitätsprinzip von Bellman

Eines der ersten Verfahren zur Lösung diskreter dynamischer Optimierungsprobleme ist die *Methode der dynamischen Programmierung*, die auf dem *Optimalitätsprinzip von Richard Bellman* beruht. Sie ist sehr flexibel, erfordert aber sehr viel Rechenleistung und hohen Speicheraufwand bei größeren Problemen. Für eine ausführliche Diskussion verweisen wir auf [Bel57], [BD71], [BG93], [**?**]. Viele Anwendungen und Beispiele finden sich in [Win04].

Sei $t_k \in \{t_0, t_1, \ldots, t_N\}$ ein fester Zeitpunkt, $\mathbb{G}_k := \{t_j : j = k, k+1, \ldots, N\}$ und $x \in Y(t_k)$.

Betrachte die folgende Schar von diskreten dynamischen Optimierungsproblemen:

Problem 8.2.1 ([)

(DOP) auf dem Restgitter $\mathbb{G}_k$ zum Anfangswert x]

($\mathrm{P}(t_k, x)$) Minimiere

$$\sum_{j=k}^{N} \varphi(t_j, y(t_j), u(t_j))$$

auf der Menge aller Gitterfunktionen

$$y : \mathbb{G}_k \longrightarrow \mathbb{R}^n \ , \ u : \mathbb{G}_k \longrightarrow \mathbb{R}^m$$

unter den Nebenbedingungen

$$\begin{aligned} y(t_{j+1}) &= g(t_j, y(t_j), u(t_j)) && (j = k, \ldots, N-1) \ , \\ y(t_k) &= x \ , \\ y(t_j) &\in Y(t_j) && (j = k, \ldots, N) \ , \\ u(t_j) &\in U(t_j, y(t_j)) && (j = k, \ldots, N) \ ! \end{aligned}$$

■

Dieses Problem ist ein parametrisches Optimierungsproblem mit dem Anfangszeitpunkt t_k und dem Anfangswert x als Parametern.

Definition 8.2.2 ([)

Wertefunktion]

Sei $t_k \in \mathbb{G}$. Für $x \in Y(t_k)$ bezeichne $V(t_k, x)$ den Optimalwert des Problems $\mathrm{P}(t_k, x)$. Für $x \notin Y(t_k)$ setzen wir $V(t_k, x) := \infty$.

Die Funktion $V : \mathbb{G} \times \mathbb{R}^n \longrightarrow \mathbb{R}$, $(t_k, x) \mapsto V(t_k, x)$, heißt Wertefunktion *des diskreten dynamischen Optimierungsproblems.* ■

Es gilt

Satz 8.2.3 ([)

Optimalitätsprinzip von Bellman]

Sei das Paar $\hat{y}(\cdot)$, $\hat{u}(\cdot)$ eine optimale Lösung von (DOP). Dann sind die Restriktionen $\hat{y}|_{\mathbb{G}_k}$ und $\hat{u}|_{\mathbb{G}_k}$ auf das Restgitter $\mathbb{G}_k$ optimal für $\mathrm{P}(t_k, \hat{y}(t_k))$.

Beweis: Wir nehmen an, dass die Restriktionen $\hat{y}|_{\mathbb{G}_k}$, $\hat{u}|_{\mathbb{G}_k}$ nicht optimal für $\mathrm{P}(t_k, \hat{y}(t_k))$ sind. Dann existiert eine zulässige Trajektorie $\tilde{y} : \mathbb{G}_k \to \mathbb{R}^n$ und eine zugehörige Steuerung $\tilde{u} : \mathbb{G}_k \to \mathbb{R}^m$ für $\mathrm{P}(t_k, \hat{y}(t_k))$ mit

$$\sum_{j=k}^{N} \varphi(t_j, \tilde{y}(t_j), \tilde{u}(t_j)) < \sum_{j=k}^{N} \varphi(t_j, \hat{y}(t_j), \hat{u}(t_j))$$

und $\tilde{y}(t_k) = \hat{y}(t_k)$. Daher sind $y : \mathbb{G} \to \mathbb{R}^n$ und $u : \mathbb{G} \to \mathbb{R}^m$ mit

$$\begin{aligned} y(t_j) &:= \begin{cases} \hat{y}(t_j) & (j = 0, 1, \ldots, k-1) \ , \\ \tilde{y}(t_j) & (j = k, k+1, \ldots, N) \ , \end{cases} \\ u(t_j) &:= \begin{cases} \hat{u}(t_j) & (j = 0, 1, \ldots, k-1) \ , \\ \tilde{u}(t_j) & (j = k, k+1, \ldots, N) \end{cases} \end{aligned}$$

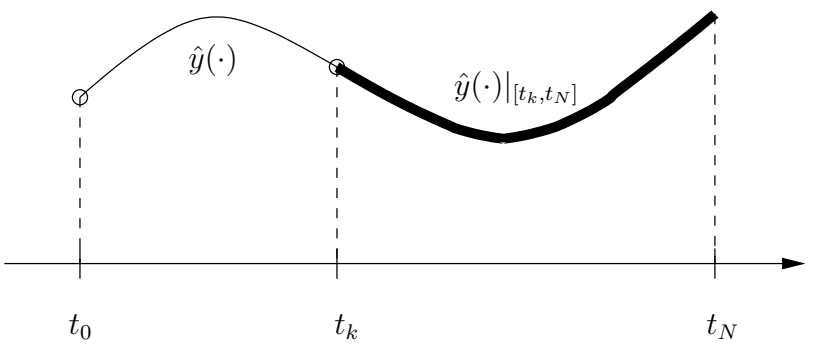


Abbildung 8.5: Bellman's Optimalitätsprinzip: Resttrajektorien von optimalen Trajektorien bleiben optimal.

zulässig für (DOP) und erfüllen

$$\sum_{j=0}^{k-1} \varphi(t_j, \hat{y}(t_j), \hat{u}(t_j)) + \sum_{j=k}^{N} \varphi(t_j, \tilde{y}(t_j), \tilde{u}(t_j)) < \sum_{j=0}^{N} \varphi(t_j, \hat{y}(t_j), \hat{u}(t_j)) \ .$$

Dies widerspricht der Optimalität von $\hat{x}(\cdot),\ \hat{u}(\cdot)$.
Für die Gültigkeit des Optimalitätsprinzips ist Folgendes wesentlich:

- Der Zustand $y(t_{j+1})$ hängt nur von den Werten von $y(\cdot)$ und $u(\cdot)$ im vorherigen Zeitpunkt t_j ab.
- Die Zielfunktion ist additiv.
- Die Zustands- und Steuerungsbeschränkungen wirken „punktweise“ nur in den einzelnen Gitterpunkten.

Das Optimalitätsprinzip ist Grundlage der folgenden Methode.

8.3 Methode der Dynamischen Programmierung

Im Folgenden verwenden wir die Konvention $\varphi(t_j, x, u) = \infty$, falls $x \notin Y(t_j)$ gilt.
Wir nutzen die Tatsache aus, dass die Wertefunktion für $\mathrm{P}(t_N, x)$ gegeben ist durch

$$V(t_N, x) = \inf_{u \in U(t_N, x)} \varphi(t_N, x, u). \tag{8.1}$$

Angenommen, wir kennen die Wertefunktion $V(t_{j+1}, x)$ für alle $x \in \mathbb{R}^n$. Aus dem Optimalitätsprinzip erhalten wir

$$V(t_j,x) = \inf_{u \in U(t_j,x)} \{\varphi(t_j, x, u) + V(t_{j+1}, g(t_j, x, u))\} \quad (j = 0, \ldots, N-1) \ . \tag{8.2}$$

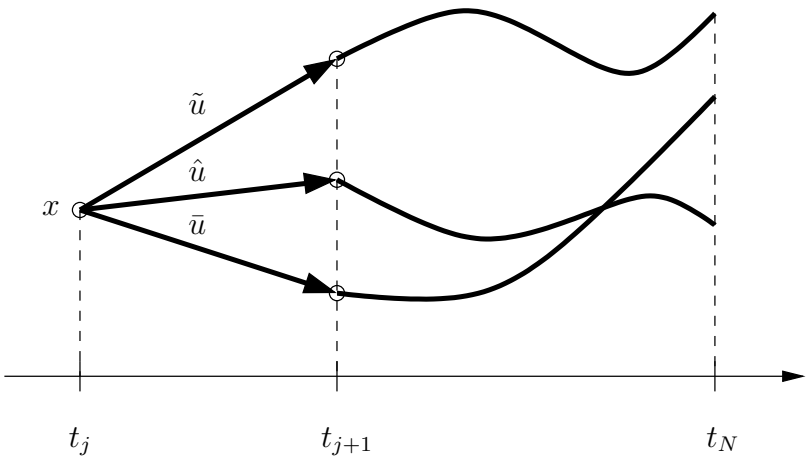


Abbildung 8.6: Bellman's Methode der dynamischen Programmierung: Rekursion der Wertefunktion.

Gleichungen (8.1) und (8.2) erlauben es uns, die Wertefunktion rückwärts in der Zeit, beginnend mit dem Zeitpunkt t_N, zu berechnen. Der optimale Anfangswert $\hat{y}(t_0)$ von (DOP) minimiert dann $V(t_0, x)$ über $x \in Y(t_0)$, ist also gegeben durch

$$\hat{y}(t_0) = \arg\min_{x \in Y(t_0)} V(t_0, x). \tag{8.3}$$

Gleichungen (8.1)-(8.3) sind die Basis der folgenden Algorithmen, die sich nur durch einen unterschiedlichen Speicherbedarf unterscheiden:

Algorithmus 8.3.1 ([)
Bellman's Methode der Dynamischen Programmierung I]

(i) **Rückwärtsrechnung**

1. *Sei $V(t_N, x)$ gegeben durch (8.1).*
2. *Für $j = N-1, \dots, 0$: Berechne $V(t_j, x)$ wie in (8.2).*

(ii) **Vorwärtsrechnung**

1. *Sei $\hat{y}(t_0)$ gegeben durch (8.3).*
2. *Für $j = 0, 1, \dots, N-1$: Bestimme*

$$\hat{u}(t_j) = arg \min_{u \in U(t_j, \hat{y}(t_j))} \{\varphi(t_j, \hat{y}(t_j), u) + V(t_{j+1}, g(t_j, \hat{y}(t_j), u))\}$$

und setze $\hat{y}(t_{j+1}) = g(t_j, \hat{y}(t_j), \hat{u}(t_j))$.

3. *Bestimme $\hat{u}(t_N) = arg \min_{u \in U(t_N, \hat{y}(t_N))} \varphi(t_N, \hat{y}(t_N), u)$.* ∎

Algorithmus 8.3.2 ([)

Bellman's Methode der Dynamischen Programmierung II]

(i) **Rückwärtsrechnung**

1. *Sei $V(t_N, x)$ gegeben durch (8.1) und $u^*(t_N, x)$ die zugehörige optimale Steuerung.*
2. *Für $j = N-1, \ldots, 0$: Berechne $V(t_j, x)$ und die zugehörige optimale Steuerung $u^*(t_j, x)$ in t_j zum Zustand x gemäß (8.2).*

(ii) **Vorwärtsrechnung**

1. *Sei $\hat{y}(t_0)$ gegeben durch (8.3).*
2. *Für $j = 0, 1, \ldots, N-1$: Setze $\hat{u}(t_j) := u^*(t_j, \hat{y}(t_j))$ und*

$$\hat{y}(t_{j+1}) := g(t_j, \hat{y}(t_j), \hat{u}(t_j)) \ .$$

3. *Setze $\hat{u}(t_N) := u^*(t_N, \hat{y}(t_N))$.* ∎

Bemerkung 8.3.3 **(F)**

alls der Algorithmus nicht vorzeitig terminiert, etwa weil die Komplexität der parametrischen Probleme $\mathrm{P}(t_k, x)$ zu hoch ist oder das Ausgangsproblem keine Optimallösung besitzt, liefert Version II eine Feedback-Steuerung *u^* als Funktion von Zeit und Zustand, die auch Systemstörungen ausgleichen kann.*
Version I ist geeignet für Implementierungen mit beschränktem Speicherplatz. Allerdings erhält man nach erfolgreichem Durchlauf auch nur eine Open Loop-Steuerung.

Beispiel 8.3.4

Wir lösen das folgende Problem mit der Methode der dynamischen Programmierung:

$$\begin{array}{lll} \text{Minimiere} & -\displaystyle\sum_{j=0}^{N-1} c(1-u_j)y(j) & \\ \text{u.d.N.} & y(j+1) = y(j)(0.9+0.6u(j)) & (j=0,\ldots,N-1)\ , \\ & y(0) = k\ , & \\ & u(j) \in [0,1] & (j=0,\ldots,N-1)\ ! \end{array}$$

Dabei sei $k > 0,\ c > 0$ und $N = 5$.
Wegen $k > 0$ und $u(j) \geq 0$ sind alle zulässigen Trajektorien $y(j) > 0$ für alle $j = 0, \ldots, N$, das nutzen wir gezielt aus bei der Auswertung der Rekursionen (8.1) und (8.2).

Da $u(5)$ in der Zielfunktion nicht vorkommt, ist $V(5,x) := 0$ $(x \in \mathbb{R})$ und die Größe von $u^*(5,x)$ unerheblich. Damit können wir die Rekursion

$$V(j,x) = \min_{0\le u\le 1}\{-cx(1-u) + V(j+1, x(0.9+0.6u))\}$$

mit $j = 4$ starten, der Zustand x variiert dabei in jeder Stufe in $\mathbb{R}_{++}$:

$$\begin{aligned}
V(4,x) &= \min_{0\le u\le 1}\{-cx(1-u)+0\} = -cx, \quad u^*(4,x) = 0\ .\\
V(3,x) &= \min_{0\le u\le 1}\{-cx(1-u) + V(4, x(0.9+0.6u))\}\\
&= \min_{0\le u\le 1}\{-cx(1-u) - cx(0.9+0.6u)\}\\
&= cx \min_{0\le u\le 1}\{-1.9+0.4u\} = -1.9cx, \quad u^*(3,x) = 0\ .\\
V(2,x) &= \min_{0\le u\le 1}\{-cx(1-u) + V(3, x(0.9+0.6u))\}\\
&= \min_{0\le u\le 1}\{-cx(1-u) - 1.9cx(0.9+0.6u)\}\\
&= cx \min_{0\le u\le 1}\{-2.71-0.14u\} = -2.85cx, \quad u^*(2,x) = 1\ .\\
V(1,x) &= \min_{0\le u\le 1}\{-cx(1-u) + V(2, x(0.9+0.6u))\}\\
&= \min_{0\le u\le 1}\{-cx(1-u) - 2.85cx(0.9+0.6u)\}\\
&= cx \min_{0\le u\le 1}\{-3.565-0.71u\} = -4.275cx, \quad u^*(1,x) = 1\ .\\
V(0,x) &= \min_{0\le u\le 1}\{-cx(1-u) + V(1, x(0.9+0.6u))\}\\
&= \min_{0\le u\le 1}\{-cx(1-u) - 4.275cx(0.9+0.6u)\}\\
&= cx \min_{0\le u\le 1}\{-4.8475-1.565u\} = -6.4125cx, \quad u^*(0,x) = 1\ .
\end{aligned}$$

Die optimale Steuerung lautet damit

$$\hat{u}(0) = \hat{u}(1) = \hat{u}(2) = 1, \hat{u}(3) = \hat{u}(4) = 0\ .$$

Vorwärtsrechnung liefert

$$\begin{aligned}
\hat{y}(0) &= k\ ,\\
\hat{y}(1) &= 1.5\cdot k\ ,\\
\hat{y}(2) &= 2.25\cdot k\ ,\\
\hat{y}(3) &= 3.375\cdot k\ ,\\
\hat{y}(4) &= 3.0375\cdot k\ ,\\
\hat{y}(5) &= 2.73375\cdot k\ .
\end{aligned}$$

Der Optimalwert ist

$$-c\hat{y}(3) - c\hat{y}(4) = -6.4125\cdot c\cdot k\ .$$

■

8.4 Diskretes Minimumprinzip

Ein anderer Zugang zur Lösung diskreter dynamischer Optimierungsprobleme wird durch die in den vorangegangenen Kapiteln bereitgestellten optimierungstheoretischen Konzepte und numerischen Verfahren geliefert. In der Tat stammt eine wesentliche Motivation für die Beschäftigung mit Optimierungsmethoden aus der dynamischen Optimierung.
Jedes diskrete dynamische Optimierungsproblem ist auch ein im Allgemeinen sehr großes, dafür aber stark strukturiertes endlichdimensionales Optimierungsproblem. Wir wollen diese Struktur jetzt nutzen zum Beweis des diskreten Minimumprinzips mittels der notwendigen Fritz John (bzw. KKT) Optimalitätsbedingungen. Dazu erinnern wir an die notwendigen Optimalitätsbedingungen für nichtlineare Optimierungsprobleme:

Problem 8.4.1 ([])

Nichtlineares Optimierungsproblem] Es seien stetig differenzierbare Funktionen

$$\begin{aligned} F &: \mathbb{R}^n \to \mathbb{R}, \\ G = (G_1, \dots, G_m)^\top &: \mathbb{R}^n \to \mathbb{R}^m, \\ H = (H_1, \dots, H_p)^\top &: \mathbb{R}^n \to \mathbb{R}^p \end{aligned}$$

und die Menge

$$S := \{z \in \mathbb{R}^n \mid \underline{z} \le z \le \overline{z}\}, \qquad \underline{z} < \overline{z} \in (\mathbb{R} \cup \{-\infty, \infty\})^n$$

gegeben.

Minimiere $F(z)$ bezüglich $z \in \mathbb{R}^n$ unter den Nebenbedingungen

$$\begin{aligned} G_i(z) &\le 0, \quad i = 1, \dots, m, \\ H_j(z) &= 0, \quad j = 1, \dots, p, \\ z &\in S. \end{aligned}$$

■

Mit Hilfe der *Lagrangefunktion*

$$\mathcal{L}(z, l_0, \mu, \lambda) := l_0 F(z) + \mu^\top G(z) + \lambda^\top H(z) = l_0 F(z) + \sum_{i=1}^{m} \mu_i G_i(z) + \sum_{j=1}^{p} \lambda_j H_j(z)$$

gelten die folgenden notwendigen Bedingungen erster Ordnung vom Fritz John-Typ:

Satz 8.4.2 ([])

Fritz John Bedingungen] Sei $\hat{z}$ ein lokales Minimum von Problem 8.4.1. Die Funktionen

F, G_i, $i = 1, \dots, m$, und H_j, $j = 1, \dots, p$, seien stetig differenzierbar. Dann existieren Multiplikatoren $l_0 \in \mathbb{R}$, $\mu = (\mu_1, \dots, \mu_m)^\top \in \mathbb{R}^m$, $\lambda = (\lambda_1, \dots, \lambda_p)^\top \in \mathbb{R}^p$ mit $(l_0, \mu, \lambda) \neq 0$, so dass die folgenden Bedingungen gelten:

(a) Vorzeichenbedingungen:

$$l_0 \geq 0, \qquad \mu_i \geq 0 \qquad (i = 1, \dots, m). \tag{8.4}$$

(b) Optimalitätsbedingung:

$$\mathcal{L}'_z(\hat{z}, l_0, \mu, \lambda)(z - \hat{z}) \geq 0 \qquad (z \in S). \tag{8.5}$$

(c) Komplementaritätsbedingungen:

$$\mu_i G_i(\hat{z}) = 0 \qquad (i = 1, \dots, m). \tag{8.6}$$

■

Unter geeigneten Regularitätsbedingungen (constraint qualifications, z.B. LICQ, MFCQ) kann $l_0 = 1$ garantiert werden.
Wir betrachten wieder das diskretisierte Optimalsteuerungsproblem in Beispiel 8.1.6 mit $\mathcal{U} := [u_{min}, u_{max}]$:

Problem 8.4.3 ([)

Diskretisiertes Optimalsteuerungsproblem]

(OCP_N) Minimiere

$$\varphi(x_0, x_N) + \sum_{j=0}^{N-1} h_j f_0(t_j, x_j, u_j)$$

unter den Nebenbedingungen

$$x_j \in \mathbb{R}^n \ (j{=}0, \dots, N) \ , \ u_j \in \mathbb{R}^m \ (j = 0, \dots, N)$$

und

$$x_{j+1} = x_j + h_j f(t_j, x_j, u_j) \quad (j{=}0, \dots, N{-}1) \ ,$$
$$r(x_0, x_N) = 0,$$
$$c(t_j, x_j, u_j) \leq 0, \quad (j = 0, \dots, N),$$
$$u_{min} \leq u_j \leq u_{max} \quad (j = 0, \dots, N) \ !$$

■

Problem 8.4.3 ist ein endlichdimensionales, nichtlineares Optimierungsproblem der Form 8.4.1 mit $(n + m)(N + 1)$ Optimierungsvariablen

$$z := (x_0, x_1, \dots, x_N, u_0, u_1, \dots, u_N)^\top,$$

der Zielfunktion

$$F(z) := \varphi(x_0, x_N) + \sum_{j=0}^{N-1} h_j f_0(t_j, x_j, u_j),$$

den $q(N+1)$ Ungleichungsnebenbedingungen

$$G(z) := \begin{pmatrix} c(t_0, x_0, u_0) \\ \vdots \\ c(t_N, x_N, u_N) \end{pmatrix},$$

den $nN + p$ Gleichungsnebenbedingungen

$$H(z) := \begin{pmatrix} x_0 + h_0 f(t_0, x_0, u_0) - x_1 \\ \vdots \\ x_{N-1} + h_{N-1} f(t_{N-1}, x_{N-1}, u_{N-1}) - x_N \\ r(x_0, x_N) \end{pmatrix},$$

und der Menge

$$S := \mathbb{R}^{n(N+1)} \times [u_{min}, u_{max}]^{N+1}.$$

Ziel dieses Abschnitts ist es, ein lokales Minimumprinzip für die volle Eulerdiskretisierung 8.4.3 durch Anwendung der Fritz John Bedingungen in Satz 8.4.2 herzuleiten.

Dazu definieren wir die Hamiltonfunktion und die erweiterte Hamiltonfunktion durch

$$\begin{aligned} \mathcal{H}(t, x, u, \lambda, l_0) &:= l_0 f_0(t, x, u) + \lambda^\top f(t, x, u), \\ \hat{\mathcal{H}}(t, x, u, \lambda, \mu, l_0) &= \mathcal{H}(t, x, u, \lambda, l_0) + \mu^\top c(t, x, u). \end{aligned}$$

Mit $l_0 \in \mathbb{R}$, $x = (x_0, \dots, x_N)^\top \in \mathbb{R}^{n(N+1)}$, $u = (u_0, \dots, u_N)^\top \in \mathbb{R}^{m(N+1)}$, $\lambda = (\lambda_0, \dots, \lambda_N)^\top \in \mathbb{R}^{n(N+1)}$, $\mu = (\mu_0, \dots, \mu_N)^\top \in \mathbb{R}^{q(N+1)}$, $\sigma \in \mathbb{R}^p$ lautet die Lagrangefunktion für Pro-

blem 8.4.3:

$$\begin{aligned}
\mathcal{L}(x,u,\lambda,\mu,\sigma,l_0) &:= l_0\varphi(x_0,x_N)+\sigma^\top r(x_0,x_N)+l_0\sum_{i=0}^{N-1}h_i f_0(t_i,x_i,u_i)\\
&\quad+\sum_{i=0}^{N-1}\lambda_{i+1}^\top\left(x_i+h_i f(t_i,x_i,u_i)-x_{i+1}\right)\\
&\quad+\sum_{i=0}^{N}\mu_i^\top c(t_i,x_i,u_i)\\
&= l_0\varphi(x_0,x_N)+\sigma^\top r(x_0,x_N)+\mu_N^\top c(t_N,x_N,u_N)\\
&\quad+\sum_{i=0}^{N-1}\left(h_i\mathcal{H}(t_i,x_i,u_i,\lambda_{i+1},l_0)+\mu_i^\top c(t_i,x_i,u_i)\right)\\
&\quad+\sum_{i=0}^{N-1}\lambda_{i+1}^\top\left(x_i-x_{i+1}\right)\\
&= l_0\varphi(x_0,x_N)+\sigma^\top r(x_0,x_N)+\mu_N^\top c(t_N,x_N,u_N)\\
&\quad+\sum_{i=0}^{N-1}h_i\hat{\mathcal{H}}\left(t_i,x_i,u_i,\lambda_{i+1},\frac{\mu_i}{h_i},l_0\right)\\
&\quad+\sum_{i=0}^{N-1}\lambda_{i+1}^\top\left(x_i-x_{i+1}\right).
\end{aligned}$$

Auswertung der Fritz-John-Bedingungen in Satz 8.4.2 liefert das diskrete lokale Minimumprinzip:

Satz 8.4.4 ([])

Diskretes lokales Minimumprinzip] Voraussetzungen:

(i) Die Funktionen φ, f_0, f, c, r seien stetig differenzierbar bzgl. x und u.

(ii) $(\hat{x},\hat{u})$ sei ein lokales Minimum von Problem 8.4.3.

Dann existieren Multiplikatoren $l_0 \in \mathbb{R}$, $\sigma \in \mathbb{R}^p$, $\lambda = (\lambda_0,\dots,\lambda_N)^\top \in \mathbb{R}^{n(N+1)}$, $\mu = (\mu_0,\dots,\mu_N)^\top \in \mathbb{R}^{q(N+1)}$ mit:

(i) $l_0 \geq 0$, $(l_0,\sigma,\lambda,\mu) \neq 0$,

(ii) Diskrete Adjungierte Gleichung:
Für $i = 0,\dots,N-1$ gilt

$$\lambda_i = \lambda_{i+1}+h_i\hat{\mathcal{H}}'_x\left(t_i,\hat{x}_i,\hat{u}_i,\lambda_{i+1},\frac{\mu_i}{h_i},l_0\right)^\top.$$

(iii) Diskrete Transversalitätsbedingungen:

$$\begin{aligned}\lambda_0 &= -\left(l_0\varphi'_{x_0}(\hat{x}_0,\hat{x}_N)^\top + r'_{x_0}(\hat{x}_0,\hat{x}_N)^\top\sigma\right),\\ \lambda_N &= l_0\varphi'_{x_f}(\hat{x}_0,\hat{x}_N)^\top + r'_{x_f}(\hat{x}_0,\hat{x}_N)^\top\sigma + c'_x(t_N,\hat{x}_N,\hat{u}_N)^\top\mu_N.\end{aligned}$$

(iii) Diskrete Optimalitätsbedingungen:
Für alle $i = 0,\dots,N-1$ und alle $u \in \mathcal{U}$ gilt

$$\hat{\mathcal{H}}'_u\left(t_i,\hat{x}_i,\hat{u}_i,\lambda_{i+1},\frac{\mu_i}{h_i},l_0\right)(u-\hat{u}_i) \geq 0.$$

Außerdem gilt $\mu_N^\top c'_u(t_N,\hat{x}_N,\hat{u}_N)(u-\hat{u}_N) \geq 0$ für alle $u \in \mathcal{U}$.

(iv) Diskrete Komplementaritätsbedingungen:
Es gilt

$$\begin{aligned}\mu_i &\geq 0 \quad (i=0,\dots,N),\\ \mu_i^\top c(t_i,\hat{x}_i,\hat{u}_i) &= 0 \quad (i=0,\dots,N).\end{aligned}$$

Beweis: Die Fritz-John-Bedingungen liefern:

$$\begin{aligned}0 &= \underbrace{l_0\varphi'_{x_0}(\hat{x}_0,\hat{x}_N) + \sigma^\top r'_{x_0}(\hat{x}_0,\hat{x}_N)}_{=:-\lambda_0} + h_0\mathcal{H}'_x\left(t_0,\hat{x}_0,\hat{u}_0,\lambda_1,\frac{\mu_0}{h_0},l_0\right) + \lambda_1^\top\\ 0 &= h_i\mathcal{H}'_x\left(t_i,\hat{x}_i,\hat{u}_i,\lambda_{i+1},\frac{\mu_i}{h_i},l_0\right) + \lambda_{i+1}^\top - \lambda_i^\top \qquad (i=1,\dots,N-1)\\ 0 &= l_0\varphi'_{x_f}(\hat{x}_0,\hat{x}_N) + \sigma^\top r'_{x_f}(\hat{x}_0,\hat{x}_N) + \mu_N^\top c'_x(t_N,x_N,u_N) - \lambda_N^\top.\end{aligned}$$

Beispiel 8.4.5 (G)
egeben sei das Problem

$$\begin{aligned}&\text{Minimiere} && -x_2(N)\\ &\text{u.d.N.} && x_1(j+1) = x_1(j)(0.9+bu(j)) && (j=0,1,\dots,N-1),\\ &&& x_2(j+1) = x_2(j)+c(1-u(j))x_1(j) && (j=0,1,\dots,N-1),\\ &&& x_1(0) = k,\\ &&& x_2(0) = 0,\\ &&& 0 \leq u(j) \leq 1 \quad (j=0,1,\dots,N-1)\ !\end{aligned}$$

Wir lösen es für $k=1, c=1, b=0.6$ und $N=5$ mit Hilfe des diskreten Minimumprinzips.

Wir formulieren die Aufgabe ein wenig um, indem die zweite Differenzengleichung aufgelöst wird und in die Zielfunktion eingesetzt wird und setzen die gegebenen Konstanten ein ($x_j := x_1(j)$, $t_j = j$, $h_j = 1$):

$$
\begin{aligned}
&\textit{Minimiere} \quad \sum_{j=0}^{N-1} -(1-u_j)x_j, \qquad f_0(t_j,x_j,u_j) = -(1-u_j)x_j \\
&\textit{unter} \qquad\qquad x_{j+1} &&= x_j(0.9+0.6u_j), \\
&\qquad\qquad\qquad\quad &&= x_j + x_j(-0.1+0.6u_j), \\
&\qquad\qquad\qquad\quad &&= x_j + f(t_j,x_j,u_j), \qquad j = 0,1,\dots,N-1 \\
&\qquad\qquad r(x_0,x_N) &&= x_0 - 1 = 0, \\
&\qquad 0 \le u_j \le 1, \quad j = 0,1,\dots,N-1.
\end{aligned}
$$

Wegen $x_0 > 0$ und $u_j \ge 0$ ist stets auch $x_j > 0$ für alle j.

- **Hamiltonfunktion:** *($j = 0,\dots,N-1$)*

$$
\begin{aligned}
\mathcal{H}(t_j,x_j,u_j,\lambda_{j+1},l_0) &= l_0 f_0(t_j,x_j,u_j) + \lambda_{j+1}^\top f(t_j,x_j,u_j) \\
&= -l_0(1-u_j)x_j + \lambda_{j+1}x_j(-0.1+0.6u_j).
\end{aligned}
$$

- **Auswertung des diskreten Minimumprinzips:**

  Es sei $\hat{x} = (\hat{x}_0,\hat{x}_1,\dots,\hat{x}_N)^\top$, $\hat{u} = (\hat{u}_0,\hat{u}_1,\dots,\hat{u}_{N-1})^\top$ ein lokales Minimum des Optimierungsproblems. Es gibt dann Multiplikatoren $l_0 \ge 0$, λ_j, $j = 1,\dots,N$, σ nicht alle Null, so dass die folgenden Bedingungen gelten:

  (i) Adjungiertes System

$$
\begin{aligned}
\lambda_0 &= -\sigma, \\
\lambda_j &= \lambda_{j+1}\cdot(0.9+0.6u_j) - l_0(1-u_j), \quad j = 0,\dots,N-1, \\
\lambda_N &= 0.
\end{aligned}
$$

  (ii) Optimalitätsbedingung (regulärer Fall $l_0 = 1$): Für alle $u \in [0,1]$ gilt

$$
\begin{aligned}
&(\hat{x}_j + \lambda_{j+1}\hat{x}_j 0.6)\,(u-\hat{u}_j) \ge 0, \qquad j = 0,\dots,N-1, \\
\Leftrightarrow\quad &(\hat{x}_j + \lambda_{j+1}\hat{x}_j 0.6)\,u \ge (\hat{x}_j + \lambda_{j+1}\hat{x}_j 0.6)\,\hat{u}_j, \qquad j = 0,\dots,N-1, \\
\Leftrightarrow\quad &\hat{u}_j = \begin{cases} 0, & \textit{falls } \hat{x}_j(1+0.6\lambda_{j+1}) > 0, \\ 1, & \textit{falls } \hat{x}_j(1+0.6\lambda_{j+1}) < 0, \\ \textit{unbestimmt}, & \textit{falls } \hat{x}_j(1+0.6\lambda_{j+1}) = 0 \end{cases} \\
&\qquad\qquad \overset{\hat{x}_j > 0}{=} \begin{cases} 0, & \textit{falls } \lambda_{j+1} > -1/0.6 = -5/3, \\ 1, & \textit{falls } \lambda_{j+1} < -5/3, \\ \textit{unbestimmt}, & \textit{falls } \lambda_{j+1} = -5/3. \end{cases}
\end{aligned}
$$

Optimalitätsbedingung (entarteter Fall $l_0 = 0$): Für alle $u \in [0,1]$ gilt

$$\begin{aligned} & \lambda_{j+1}\hat{x}_j 0.6(u - \hat{u}_j) \geq 0, \qquad j = 0, \ldots, N-1, \\ \Leftrightarrow \quad & \lambda_{j+1}\hat{x}_j 0.6u \geq \lambda_{j+1}\hat{x}_j 0.6\hat{u}_j, \qquad j = 0, \ldots, N-1, \\ \Leftrightarrow \quad & \hat{u}_j = \begin{cases} 0, & \text{falls } \lambda_{j+1} > 0, \\ 1, & \text{falls } \lambda_{j+1} < 0, \\ \text{unbestimmt}, & \text{falls } \lambda_{j+1} = 0 \end{cases} \end{aligned}$$

Im entarteten Fall folgt aus dem Adjungierten System mit $l_0 = 0$ und der Endbedingung $\lambda_N = 0$, dass $\lambda_j = 0$ für alle $j = 0, \ldots, N$ gilt. Die Optimalitätsbedingung liefert keine Aussage für die optimale Steuerung.

Im regulären Fall $l_0 = 1$ läßt sich das Adjungierte System zusammen mit den Optimalitätsbedingungen rückwärts lösen ($N = 5$):

$$\begin{array}{rclcrcl} \lambda_5 & = & 0 > -5/3 & \Longrightarrow & \hat{u}_4 & = & 0, \\ \lambda_4 & = & -1 > -5/3 & \Longrightarrow & \hat{u}_3 & = & 0, \\ \lambda_3 & = & -1.9 < -5/3 & \Longrightarrow & \hat{u}_2 & = & 1, \\ \lambda_2 & = & -2.85 < -5/3 & \Longrightarrow & \hat{u}_1 & = & 1, \\ \lambda_1 & = & -4.275 < -5/3 & \Longrightarrow & \hat{u}_0 & = & 1, \\ \lambda_0 & = & -6.4125 = -\sigma. & & & & \end{array}$$

Vorwärtsintegration liefert $\hat{x}_0 = 1, \hat{x}_1 = 1.5, \hat{x}_2 = 2.25, \hat{x}_3 = 3.375, \hat{x}_4 = 3.0375, \hat{x}_5 = 2.73375$.

Der Zielfunktionswert ist $-\hat{x}_3 - \hat{x}_4 = -6.4125$.

■

Kapitel 9

Heuristische Optimierungsverfahren

In der Praxis ist man häufig mit sehr komplexen Optimierungsproblemen konfrontiert, die diskrete (ganzzahlige) Optimierungsvariable enthalten oder nicht die notwendigen Differenzierbarkeitseigenschaften für die bisher besprochenen gradientenbasierten Optimierungsverfahren aufweisen. Zudem liefern gradientenbasierte Verfahren in der Regel nur lokale Minima, wobei eigentlich die globalen Minima gesucht sind. Daher wird in der Praxis gerne auf heuristische Optimierungsverfahren zurückgegriffen, die globale Minima liefern sollen und dabei nur Funktionswerte und keine Ableitungsinformationen benötigen, wodurch sie sehr bequem anwendbar sind. Die Bezeichnung "heuristisch" ist darin begründet, dass für diese Verfahren im Allgemeinen keine Konvergenztheorie existiert, so dass nicht garantiert werden kann, dass die Verfahren ein lokales oder globales Minimum oder zumindest einen stationären Punkt liefern. Trotz dieser fehlenden (oder unzureichenden) Konvergenztheorie liefern die Verfahren in der Praxis häufig gute Resultate.

9.1 Das Verfahren von Nelder und Mead

Wir folgen der Darstellung von W. Alt [Alt02], welche auf dem Originalartikel [NM65] basiert.

In der Praxis erfreut sich das Verfahren von Nelder und Mead großer Beliebtheit, da es keinerlei Voraussetzungen an die zu minimierende Funktion f stellt. Insbesondere werden zur Durchführung des Algorithmus lediglich Funktionswerte von f benötigt. Allerdings gibt es auch keine allgemeinen Konvergenzaussagen und keine Garantie, dass das Verfahren tatsächlich ein (zumindest lokales) Minimium von f liefert. Für spezielle Funktionen und niedrige Raumdimensionen werden in [LRWW98] Konvergenzresultate erzielt. Jedoch sind diese Resultate nicht allgemein gültig, da in [McK98] Beispiele konstruiert werden, für die das Verfahren gegen nichtstationäre Punkte konvergiert. Es gibt neuerdings jedoch auch eine konvergente Modifikation des Verfahrens, vgl. [PCB02].

Das Verfahren basiert auf der Konstruktion von Simplizes.

Definition 9.1.1 **(Simplex)**
Seien $x^0, x^1, \dots, x^n \in \mathbb{R}^n$ gegebene Vektoren, wobei die Vektoren $x^i - x^0$, $i = 1, \dots, n$ linear unabhängig seien. Die konvexe Hülle

$$S = \left\{ x = \sum_{i=0}^{n} \lambda_i x^i \mid \lambda_i \geq 0,\ i = 0, 1, \dots, n,\ \sum_{i=0}^{n} \lambda_i = 1 \right\}$$

dieser $n+1$ Punkte im $\mathbb{R}^n$ heißt **(n-dimensionales) Simplex mit Ecken** $x^0, x^1, \dots, x^n$.

Zum Start des Verfahrens wird ein Simplex S_0 vorgegeben. Jeder Iterationsschritt des Verfahrens besteht aus folgenden Aktionen:

- Bestimme zum aktuellen Simplex S_k mit den Ecken $x^0, x^1, \dots, x^n$ die Ecke x^m mit dem größten Funktionswert:

$$f(x^m) = \max\{f(x^0), f(x^1), \dots, f(x^n)\}.$$

- Berechne einen Punkt y mit einem kleineren Funktionswert $f(y) < f(x^m)$ und ersetze x^m durch den neuen Punkt y. Damit erhält man einen neuen Simplex S_{k+1} zu den Punkten y und x^i, $i = 0, 1, \dots, n$, $i \neq m$.

Für $j = 0, 1, \dots, n$ sind die **Schwerpunkte der Ecken bzgl.** x^j durch

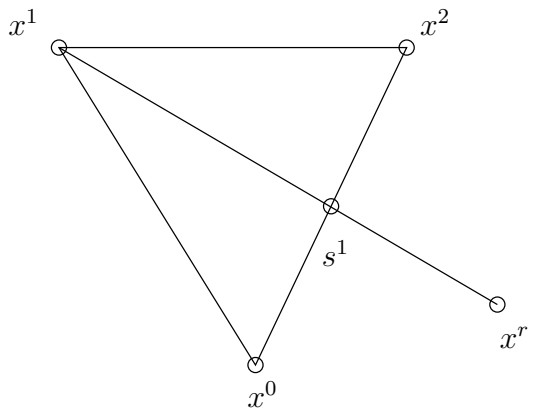
$$s^j = \frac{1}{n} \sum_{\substack{i=0 \\ i \neq j}}^{n} x^i$$

gegeben.

Zur Bestimmung des neuen Punktes y werden drei Konstruktionsprinzipien verwendet:

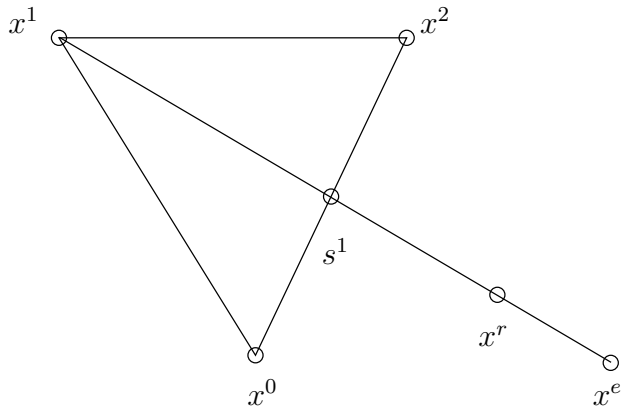
(i) **Reflektion**
Ein neuer Punkt x^r wird durch Reflektion der Ecke x^j am Schwerpunkt s^j bestimmt:

$$x^r = s^j + \gamma(s^j - x^j), \qquad 0 < \gamma \leq 1.$$



(ii) **Expansion**
Der Punkt x^e wird über x^r hinaus weiter nach aussen in Richtung $s^j - x^j$ bzw. $x^r - s^j$ verschoben:

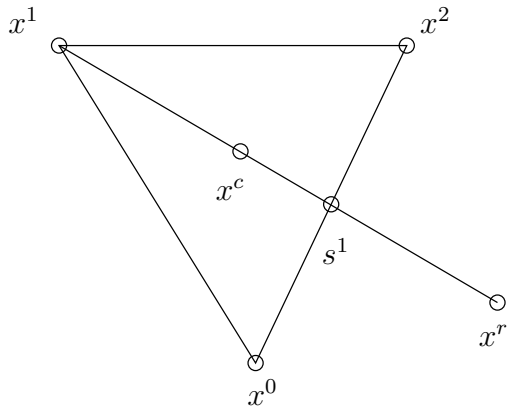
$$x^e = s^j + \beta(x^r - s^j), \qquad \beta > 1.$$



(iii) **Kontraktion**

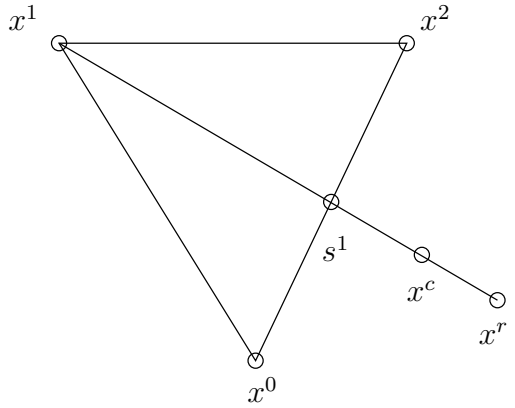
– **Innere partielle Kontraktion**
Der Punkt x^c wird zwischen x^j und s^j verschoben:

$$x^c = s^j + \alpha(x^j - s^j), \qquad 0 < \alpha < 1.$$



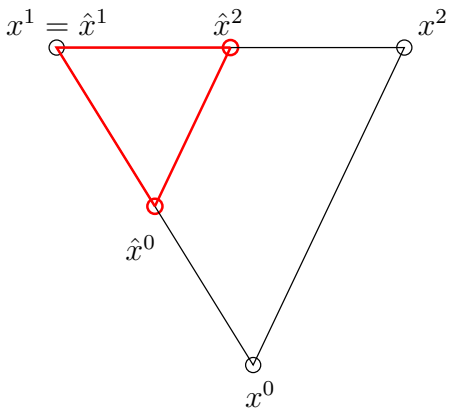
– **Äussere partielle Kontraktion**
Der Punkt x^c wird zwischen s^j und x^r verschoben:

$$x^c = s^j + \alpha(x^r - s^j), \qquad 0 < \alpha < 1.$$



– **Totale Kontraktion**
Die Punkte x^i, $i = 0, 1, \ldots, n$ mit $i \neq j$ werden durch die Mittelpunkte der Strecken von x^j nach x^i ersetzt:

$$\hat{x}^i = x^i + \frac{1}{2}(x^j - x^i) = \frac{1}{2}(x^j + x^i)$$



Insgesamt erhalten wir den folgenden Algorithmus.

Algorithmus 9.1.2 **(Verfahren von Nelder und Mead)**

(0) *Gegeben seien Parameter* $0 < \alpha < 1$, $\beta > 1$ *und* $0 < \gamma \leq 1$, *sowie ein Startpunkt* $x^{(0,0)} \in \mathbb{R}^n$. *Setze* $k = 0$.

(i) *Bestimme die Eckpunkte des Startsimplex* S_0*:*

$$x^{(0,i)} = x^{(0,0)} + e_i, \qquad i = 1, 2, \ldots, n.$$

(e_i *bezeichnet den* i*-ten Einheitsvektor)*

(ii) *Bestimme einen Punkt* $x^{(k,m)}$ *mit*

$$f(x^{(k,m)}) = \max\{f(x^{(k,0)}), \ldots, f(x^{(k,n)})\},$$

einen Punkt $x^{(k,l)}$ *mit*

$$f(x^{(k,l)}) = \min\{f(x^{(k,0)}), \ldots, f(x^{(k,n)})\},$$

sowie den Schwerpunkt bzgl. $x^{(k,m)}$

$$s^{(k,m)} = \frac{1}{n} \sum_{\substack{i=0 \\ i \neq m}}^{n} x^{(k,i)}.$$

(iii) *Berechne den Punkt*

$$x^r = s^{(k,m)} + \gamma(s^{(k,m)} - x^{(k,m)})$$

durch Reflektion von $x^{(k,m)}$ *an* $s^{(k,m)}$.

(iv) *(a)* *Falls $f(x^r) < f(x^{(k,l)})$ gilt, ist x^r neuer minimaler Punkt. Versuche durch Expansion einen noch besseren Wert zu erhalten. Berechne*

$$x^e = s^{(k,m)} + \beta(x^r - s^{(k,m)})$$

und ersetze $x^{(k,m)}$ durch den besten der beiden Punkte x^r und x^e:

$$x^{(k+1,m)} = \begin{cases} x^e, & \text{falls } f(x^e) < f(x^r), \\ x^r, & \text{falls } f(x^r) \leq f(x^e). \end{cases}$$

(b) *Falls*

$$f(x^{(k,l)}) \leq f(x^r) \leq \max\{f(x^{(k,j)}) \mid j \neq m\}$$

gilt, setze $x^{(k+1,m)} = x^r$. Hier ist x^r höchstens schlechter als $x^{(k,l)}$ und in der Regel besser als $x^{(k,m)}$.

(c) *Falls*

$$f(x^r) > \max\{f(x^{(k,j)}) \mid j \neq m\}$$

gilt, unterscheide folgende Fälle:

I. Falls $f(x^r) > f(x^{(k,m)})$ gilt, so ist x^r eine Verschlechterung und es ist möglicherweise ratsam, den Simplex nicht zu verlassen. Führe innere partielle Kontraktion durch und berechne

$$x^c = s^{(k,m)} + \alpha(x^{(k,m)} - s^{(k,m)}).$$

II. Falls $f(x^r) < f(x^{(k,m)})$ gilt, so ist x^r zumindest besser als $x^{(k,m)}$, allerdings schlechter als alle anderen Eckpunkte. Es ist möglicherweise ratsam, näher am Simplex zu suchen. Führe äussere partielle Kontraktion durch und berechne

$$x^c = s^{(k,m)} + \alpha(x^r - s^{(k,m)}).$$

Ist $f(x^c) < f(x^{(k,m)})$, setze $x^{(k+1,m)} = x^c$. Andernfalls haben wir trotz aller Versuche keine Verbesserung erreichen können und führen daher eine totale Kontraktion bzgl. des momentan besten Punktes $x^{(k,l)}$ durch und setzen

$$x^{(k+1,i)} = \frac{1}{2}(x^{(k,i)} + x^{(k,l)}), \qquad i \neq l.$$

(v) *Setze $k = k + 1$ und gehe zu (ii).*

Abbruchkriterien:

- Nelder und Mead schlagen folgendes Abbruchkriterium vor: Die Standardabweichung der Funktionswerte an den Simplexecken soll kleiner als eine vorgegebene Toleranz sein, d.h.

$$\left(\frac{1}{n+1} \sum_{i=0}^{n} \left(f(x^{(k,i)}) - \bar{f}_k \right)^2 \right)^{1/2} < tol,$$

  wobei

$$\bar{f}_k = \frac{1}{n+1} \sum_{j=0}^{n} f(x^{(k,j)})$$

  den Mittelwert der Funktionswerte an den Simplexecken bezeichnet. Dieses Kriterium wird auch in der NAG-Version `E04CCF` des Verfahrens verwendet.

- In der MATLAB-Implementation `fminsearch` des Verfahrens wird das Verfahren abgebrochen, sobald der Durchmesser des Simplex S_k kleiner als eine gegebene Toleranz ist.

Bemerkung 9.1.3

Der Punkt $x^{(k,l)}$ kann als aktuelle Iterierte betrachtet werden, da dieser Punkt den minimalen Funktionswert unter allen Ecken des aktuellen Simplex S_k liefert. Per Konstruktion gilt zumindest

$$f(x^{(k+1,l_{k+1})}) \leq f(x^{(k,l_k)}), \qquad k = 0, 1, \ldots,$$

wobei l_k den minimalen Funktionswert in Iteration k bezeichnet.
Für die Konstanten haben sich in zahlreichen Anwendungen die Werte $0.4 \leq \alpha \leq 0.6$, $2 \leq \beta \leq 3$ und $\gamma = 1$ bewährt. ∎

Beispiel 9.1.4

Wir wollen die nichtdifferenzierbare Funktion

$$l_1(x, y; \alpha) = f(x, y) + \alpha |h(x, y)| + \alpha \max\{0, g_1(x, y)\} + \alpha \max\{0, g_2(x, y)\}$$

mit $\alpha = 100$ und

$$\begin{aligned} f(x, y) &= (x-2)^2 + (y-3)^2, \\ h(x, y) &= y + \frac{x}{2} - \frac{1}{2}, \\ g_1(x, y) &= y + 2x^2 - 2, \\ g_2(x, y) &= x^2 - y - 1, \end{aligned}$$

minimieren. Die Optimallösung lautet

$$x = \frac{3}{5}, \quad y = \frac{1}{5}, \quad f(x,y) = \frac{49}{5}.$$

Das Programm `fminsearch` *der* MATLAB*-Optimization Toolbox, welches eine Implementation des Verfahrens von Nelder und Mead darstellt, findet die Optimallösung ausgehend vom Startwert* $(0,0)^\top$ *tatsächlich. Aufruf von*

```
[x,val,exitflag,output] = fminsearch(@l1penalty,[0,0],...
                    optimset('TolX',1e-12,'Display','iter'))
```

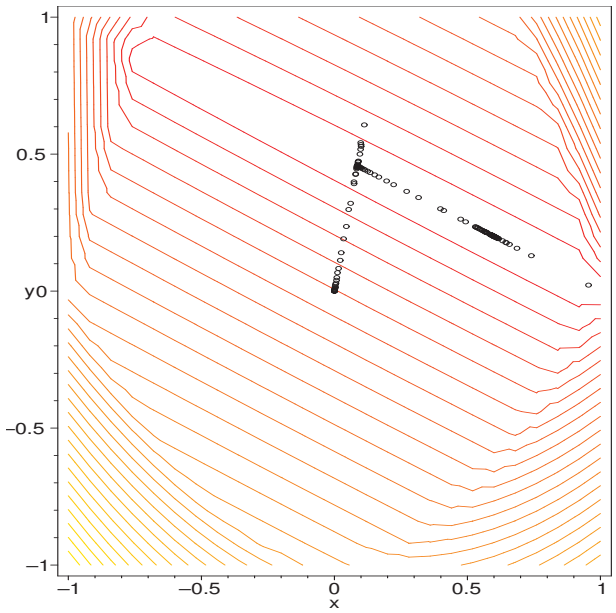
wobei die Funktion in `l1penalty.m` *gemäß*

```
function f = l1penalty(x)
  alpha=100;
  f = (x(1)-2)^2 + (x(2)-3)^2 + alpha*abs(x(2)+x(1)/2-0.5) ...
                              + alpha*max(0,x(2)+2*x(1)^2-2) ...
                              + alpha*max(0,x(1)^2-x(2)-1);
```

gegeben ist, liefert die Ausgabe:

```
Optimization terminated successfully:
 the current x satisfies the termination criteria using OPTIONS.TolX of
 1.000000e-12 and F(X) satisfies the convergence criteria using
 OPTIONS.TolFun of 1.000000e-04
x =
  0.6000    0.2000
val =
  9.8000
exitflag =
   1
output =
  iterations: 180
   funcCount: 366
   algorithm: 'Nelder-Mead simplex direct search'
```

Die Abbildung veranschaulicht diejenigen Punkte, an denen Funktionsauswertungen vorgenommen wurden:



9.2 Evolutionäre Algorithmen

Wir schildern im Folgenden die Grundidee so genannter *evolutionärer Algorithmen* (auch bekannt als genetische Algorithmen, „naturnahe“ Algorithmen), die sich in der Praxis dank ihrer Flexibilität großer Beliebtheit erfreuen. Für ausführlichere Informationen vgl. [Sch77, B̈96, BFM97, Nis97, Rud97, **?**, DLJD00, Rot01, Rot06, Gol07]. Evolutionäre Algorithmen gehören zur Klasse der stochastischen Optimierungsverfahren und kommen insbesondere dann zur Anwendung, wenn die zu minimierende Funktion keine ausreichenden Stetigkeits- oder Differenzierbarkeitseigenschaften besitzt oder ein globales Minimum gesucht ist. Typische Anwendungen sind Betriebsablaufplanung, Personaleinsatzplanung (Stundenpläne für Pflegepersonal), Containertransport, Minimierung von Maschinenrüstzeiten, nichtlineare gemischt-ganzzahlige Optimierungsprobleme (Satellitenpositionierung, Raumfahrtmissionen), Strukturoptimierung (die Form des japanischen Schnellzugs Shinkansen wurde mit Hilfe genetischer Algorithmen optimiert) und viele mehr. Die Anwendung evolutionärer Algorithmen liefert häufig gute Anfangserfolge, d.h. bestehende Lösungen werden schnell verbessert, allerdings haben evolutionäre Algorithmen Schwierigkeiten, ein Optimum mit hoher Genauigkeit zu erreichen. Darüber hinaus besitzen diese Algorithmen stochastische Komponenten, so dass es keine Garantie gibt, ein Optimum tatsächlich zu erreichen. Die Berücksichtigung von Restriktionen ist problematisch und wird in der Regel mit Hilfe von Penalty-Funktionen realisiert.

9.2.1 Modellierung evolutionärer Algorithmen

Zu lösen sei das restringierte Optimierungsproblem

$$\text{Maximiere} \qquad f(x) \qquad \text{u.d.N.} \qquad x \in \Sigma.$$

Der zulässige Bereich wird durch die Menge $\Sigma \subseteq \mathbb{R}^n$ beschrieben und kann beliebig sein. Die Zielfunktion $f : \mathbb{R}^n \longrightarrow \mathbb{R} \cup \{\pm\infty\}$ ist hierbei eine beliebige Funktion, wobei sinnvollerweise $f(x) \neq +\infty$ für alle $x \in \Sigma$ sein sollte. Ansonsten werden keinerlei Stetigkeits- oder Differenzierbarkeitsannahmen für f gemacht. Die Restriktion $x \in \Sigma$ kann formal auch dadurch modelliert werden, dass $f(x) = -\infty$ für $x \notin \Sigma$ gesetzt wird und f dann auf dem ganzen $\mathbb{R}^n$ maximiert wird.

Evolutionäre Algorithmen simulieren die natürliche Auslese in der Evolution.

Definition 9.2.1 **(Individuum, Fitnessfunktion, Population)**

- *Jedes $x \in \Sigma$ heißt* **Individuum**.
- *Die Funktion $f : \mathbb{R}^n \to \bar{\mathbb{R}}$ heißt* **Fitnessfunktion**. *$f(x)$ bezeichnet die* **Fitness** *des Individuums x.*
- *Eine Teilmenge $\{x^1, \ldots, x^m\} \subseteq \Sigma$ heißt* **Population der Größe** *m.*

Ziel eines evolutionären Algorithmus ist es, Individuen mit maximaler Fitness zu finden. Dies soll durch Verwendung der folgenden Operatoren geschehen.

Definition 9.2.2 **(Mutations-, Rekombinations-, Selektionsoperator)**

- *Der* **Mutationsoperator** *ist eine Abbildung $\mathcal{M} : \mathbb{R}^n \to \mathbb{R}^n$, die einem stochastischen Einfluss unterliegt.*
- *Eine Funktion $\mathcal{R} : (\mathbb{R}^n)^q \to (\mathbb{R}^n)^r$, welche q Individuen miteinander kombiniert und r neue Individuen erzeugt, wird* **Rekombinationsoperator (Crossover-Operator)** *genannt. Der Rekombinationsoperator darf ebenfalls einem stochastischen Einfluss unterliegen.*
- *Eine Funktion $\mathcal{S} : (\mathbb{R}^n)^s \to (\mathbb{R}^n)^p$, die aus s Individuen $p \leq s$ Individuen anhand ihrer Fitnesswerte und unter einem optionalen stochastischen Einfluss auswählt, wird* **Selektionsoperator** *genannt.* ■

Beispiel 9.2.3

Sei $n = 1$ und $x, y \in \mathbb{R}$. Dann werden durch

- $\mathcal{M} : \mathbb{R} \to \mathbb{R}$ *mit* $\mathcal{M}(x) := x + N(0,1) \quad (x \in \mathbb{R})$,
- $\mathcal{R} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ *mit* $\mathcal{R}(x,y) := \frac{x+y}{2} \quad (x, y \in \mathbb{R})$,
- $\mathcal{S} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ *mit* $\mathcal{S}(x,y) := arg\max\{f(x), f(y)\}$ $(x, y \in \mathbb{R})$

Mutations-, Rekombinations- und Selektionsoperatoren erklärt. In diesem Beispiel ist der Mutationsoperator stochastisch, während Rekombinations- und Selektionsoperator deterministisch sind. $N(0,1)$ *bezeichnet eine normalverteilte Störung mit Erwartungswert* 0 *und Varianz* 1. ■

Für den Fall, dass Σ nur aus endlich vielen Werten besteht, beschreiben wir nachfolgend einige Standardformen von Rekombinations- (bzw. Crossover-), Mutations- und Selektionsoperatoren.

Beispiel 9.2.4

Σ *sei eine Menge mit endlich vielen Elementen, deren Individuen als Bit-Strings der Länge* ℓ *codiert werden können, so dass*

$$\Sigma \subseteq \{0,1\}^\ell$$

gilt. Die folgenden Operatoren werden häufig verwendet:

a) **1-Punkt-Crossover** *:*

Aus zwei Individuen x *und* y *wird ein neues Individuum erzeugt,*

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_\ell \end{pmatrix}, \ \sigma \in \{0, \ldots, \ell\}, \ \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_\ell \end{pmatrix} \longmapsto \begin{pmatrix} x_1 \\ \vdots \\ x_\sigma \\ y_{\sigma+1} \\ \vdots \\ y_\ell \end{pmatrix}.$$

b) **Uniform-Crossover** *:*

Aus zwei Individuen x *und* y *wird ein neues Individuum erzeugt,*

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_\ell \end{pmatrix}, \ 1 \leq \sigma_1 < \sigma_2 < \ldots < \sigma_\nu \leq \ell, \ \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_\ell \end{pmatrix} \longmapsto \begin{pmatrix} x_1 \\ \vdots \\ x_{\sigma_1 - 1} \\ y_{\sigma_1} \\ x_{\sigma_1 + 1} \\ \vdots \end{pmatrix}.$$

c) **Mutation** *:*

Es sei p_m die **Mutationswahrscheinlichkeit** *eines Bits. Dann mutiert das Individuum x in ein neues Individuum gemäß*

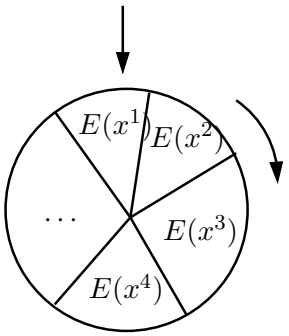
$$\begin{pmatrix} x_1 \\ \vdots \\ x_\ell \end{pmatrix} \longmapsto \begin{pmatrix} y_1 \\ \vdots \\ y_\ell \end{pmatrix} ,$$

wobei die Wahrscheinlichkeit für $y_j \neq x_j$ gleich p_m ist ($j = 1, \ldots, \ell$).

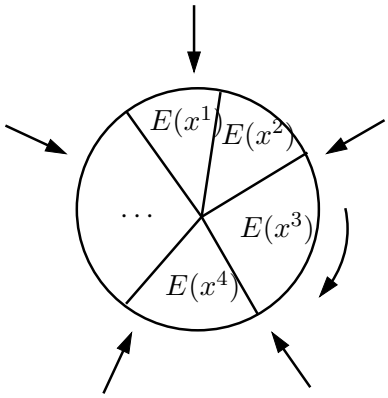
d) **Fitnessproportionale Selektion** *:*

Die Auswahlwahrscheinlichkeit $E(x^i)$ des Individuums x^i für die nächste Elterngeneration wird proportional zu dessen Fitness festgelegt.

Die Selektion von μ Individuen erfolgt dann, indem ein Glücksrad mit nur einem Auswahlpfeil wie in der folgenden Abbildung μ-mal gedreht wird:



Alternativ kann auch ein Glückrad mit μ Auswahlpfeilen einmal gedreht werden, um die Individuen auszuwählen:



e) **Elitäre Selektion** *:*

Die aktuell besten Individuen werden mit der Wahrscheinlichkeit 1 ausgewählt.

■

Ein Prototypalgorithmus für einen evolutionären Algorithmus lautet wie folgt:

Algorithmus 9.2.5 (Konzeptioneller evolutionaerer Algorithmus)

(0) Setze $t = 0$ und initialisiere die Population $\mathcal{P}^{(t)}$ mit m Individuen.

(1) Wähle Individuen $x^j \in \mathcal{P}^{(t)}$ $(j = 1, \ldots, q)$ aus der Population aus. Erzeuge neue Individuen y^j $(j = 1, \ldots, r)$ durch Anwendung des Rekombinationsoperators:

$$(y^1, \ldots, y^r) := \mathcal{R}(x^1, \ldots, x^q) \ .$$

(2) Wende den Mutationsoperator auf die rekombinierten Individuen an:

$$z^j := \mathcal{M}(y^j) \qquad (j = 1, \ldots, r) \ .$$

(3) Wende den Selektionsoperator auf die Population $\mathcal{P}^{(t)} \cup \{z^1, \ldots, z^r\}$ an und setze

$$\mathcal{P}^{(t+1)} := \mathcal{S}(x^1, \ldots, x^m, z^1, \ldots, z^r) \ .$$

(4) Solange ein (passend gewähltes) Abbruchkriterium nicht erfüllt ist, setze

$$t := t + 1$$

und gehe zu (1) .

Bemerkung 9.2.6

- *Bei der Verwendung eines evolutionären Algorithmus kann nicht sichergestellt werden, dass der Algorithmus nach einer endlichen Anzahl von Iterationsschritten das globale Optimum erreicht hat.*
- *Evolutionäre Algorithmen benötigen in der Regel sehr viele Iterationen. Durch den stochastischen Einfluss sind die Ergebnisse mitunter schwer zu reproduzieren.*
- *Die Theorie der Markovketten erlaubt eine Konvergenzanalyse für evolutionäre Algorithmen bei endlichem Suchraum, vgl. [GL11, Kapitel 9] und die darin zitierte Literatur. Die Konvergenzresultate sind jedoch recht schwach, da lediglich die Wahr-*

scheinlichkeiten der nichtoptimalen Populationen gegen Null konvergieren.

■

9.2.2 Numerische Simulation evolutionärer Algorithmen

Wir untersuchen an drei Beispielen, wie sich der evolutionäre Prototypalgorithmus numerisch verhält. Die Berechnungen für die ersten beiden Beispiele wurden mit der C++-Bibliothek GALIB in der Version 2.4 durchgeführt, die von Matthew Wall am Massachusetts Institute of Technology (http://lancet.mit.edu/ga) entwickelt wurde. Der ausgewählte Algorithmus ist dabei elitär.

Das erste Beispiel ist ein einfaches akademisches Beispiel zur Maximierung einer glatten konkaven Funktion.

Beispiel 9.2.7

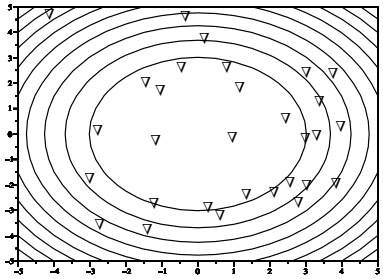
Wir wenden einen evolutionären Algorithmus an zur Maximierung der Funktion

$$f(x,y) = -x^2 - y^2$$

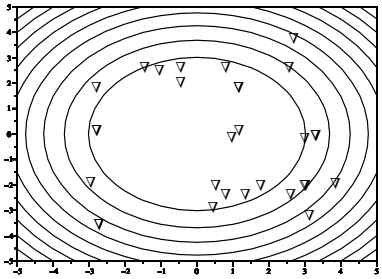
im Quader $-5 \leq x, y \leq 5$. *Der Optimalwert ist* $f_{opt} = 0$, *welcher in* $(x,y) = (0,0)$ *angenommen wird.*

Der genetische Algorithmus liefert die Lösung $(x,y) = (0.00480659, -0.00251774)$ *mit Funktionswert* $-2.94423 \cdot 10^{-5}$*:*

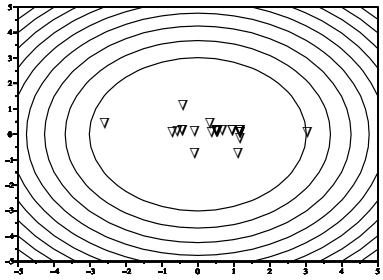
Generation 0:



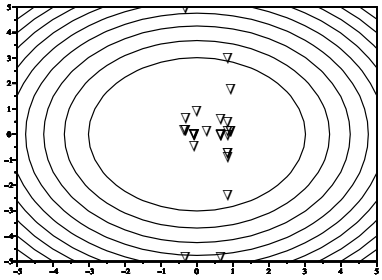
Generation 1:



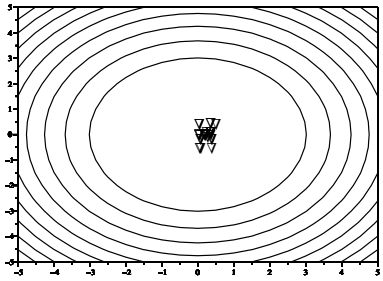
Generation 10:



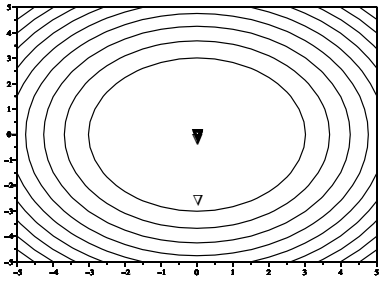
Generation 25:

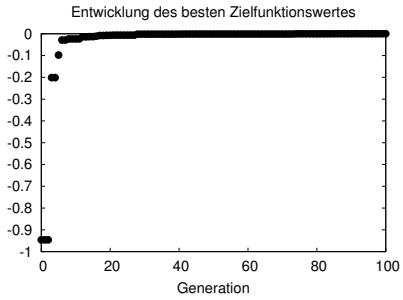


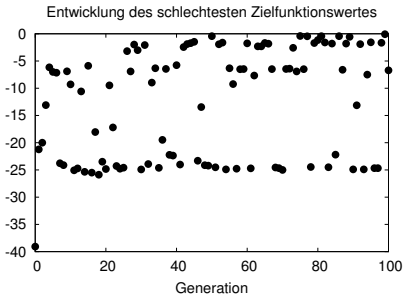
Generation 50:



Generation 100:







■

Das folgende Beispiel ist ein ebenfalls akademisches Beispiel zur Maximierung einer nicht-differenzierbaren Funktion.

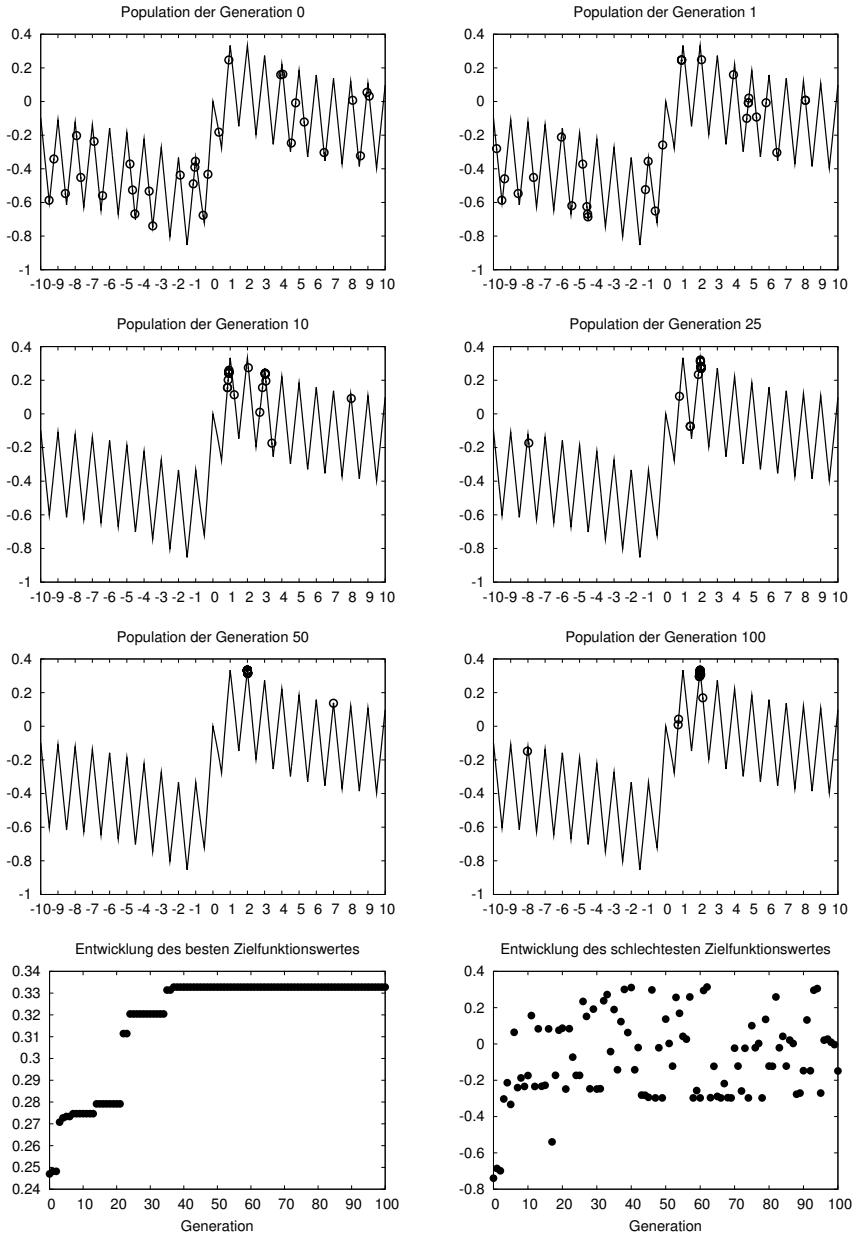
Beispiel 9.2.8

Wir wenden einen evolutionären Algorithmus an zur Maximierung der Funktion

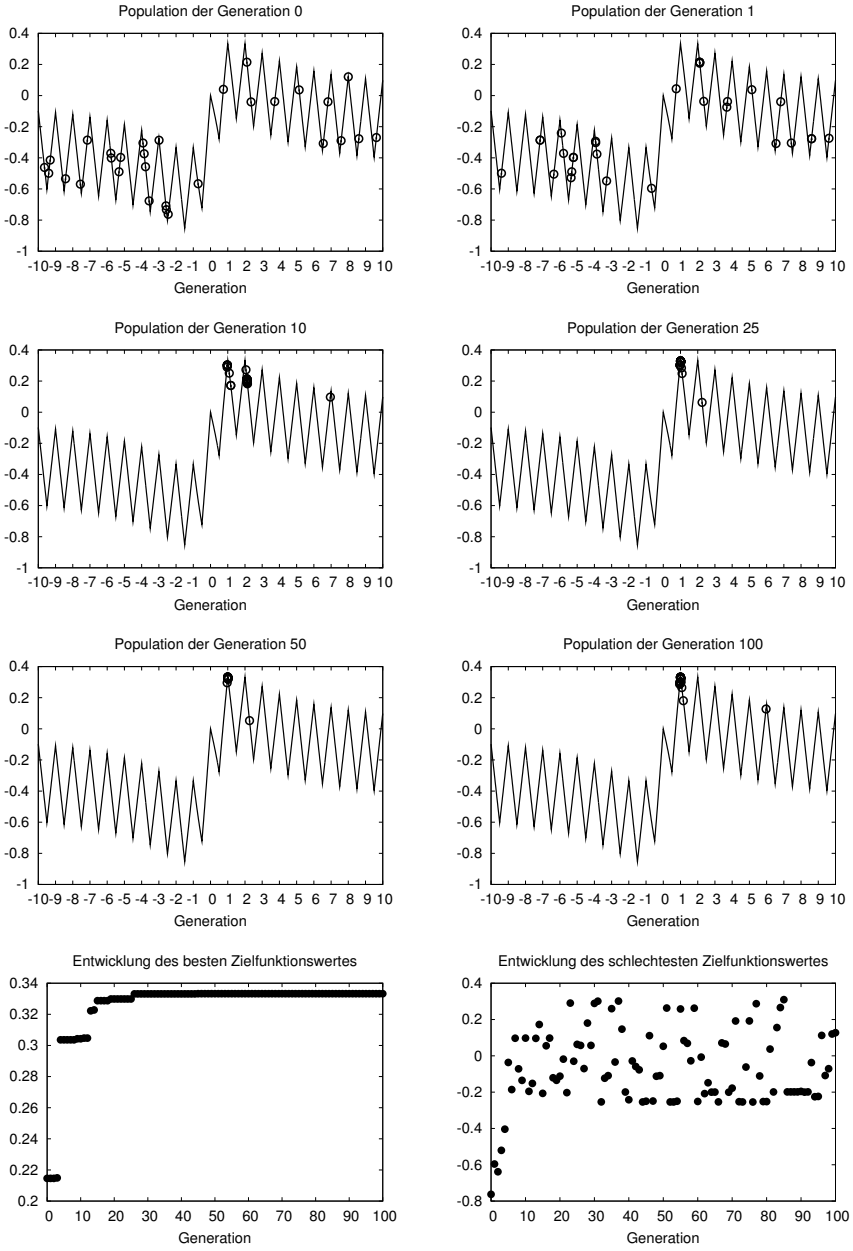
$$f(x) = \frac{x}{x^2+2} - |x - round(x)|$$

im Intervall $-10 \leq x \leq 10$*, wobei* $round(x)$ *die Zahl* x *auf die nächstgelegene ganze Zahl rundet. Der Optimalwert ist* $f_{opt} = 1/3$*, welcher an den Punkten* $x = 1$ *und* $x = 2$ *angenommen wird. Beachte, dass die Funktion* f *sehr viele lokale Minima und Maxima besitzt und nicht differenzierbar ist, so dass lokale, gradientenbasierte Verfahren nicht anwendbar sind.*

Der erste Lauf liefert die Lösung $x = 1.99939$ *mit Funktionswert* 0.332757*:*



Ein zweiter Lauf liefert die Lösung $x = 0.999924$ *mit Funktionswert* 0.333249*:*



Weitere Durchläufe liefern mitunter auch nicht-optimale Lösungen!

■

Das folgende Beispiel aus [Fin04] behandelt das Problem, Aufträge in optimaler Reihenfolge auf einer Maschine abzuarbeiten.

Beispiel 9.2.9

n Aufträge sollen nacheinander genau einmal durch eine Maschine bearbeitet werden. Gesucht ist eine optimale Reihenfolge der Aufträge, so dass die Abarbeitungszeit aller

Aufträge auf der Maschine möglichst gering ausfällt. Hierbei sind Rüstzeiten zu berücksichtigen, die entstehen, wenn Maschinen für die Bearbeitung einer neuen Aufgabe umgerüstet werden müssen. Die Rüstzeiten, die bei einem Wechsel von Auftrag i auf Auftrag j entstehen, werden mit a_{ij} bezeichnet und in der Rüstmatrix $A \in \mathbb{R}^{(n+1)\times n}$ zusammengefasst, welche noch um eine 0-te Zeile für einen künstlichen Auftrag 0 erweitert wurde, um auch die Rüstzeiten für den ersten auszuführenden Auftrag berücksichtigen zu können.

Das Problem kann als Travelling Salesman Problem interpretiert werden, bei dem die Maschine (Handlungsreisender) die n Aufträge (Städte) abarbeitet (besucht), wobei man auf die Forderung verzichtet, dass die zuerst besuchte Stadt gleich der zuletzt besuchten Stadt sein muss. Die Matrix A beschreibt dabei die Entfernungen (Dauer der Rüstzeiten).

Gesucht ist hier also eine Permutation $\pi : \{1,\dots,n\} \longrightarrow \{1,\dots,n\}$, so dass

$$a_{0,\pi(1)} + \sum_{i=1}^{n-1} a_{\pi(i),\pi(i+1)}$$

minimal wird.

Bei der Lösung dieses Problems mit evolutionären Algorithmen entspricht jedes Individuum einer Permutation der Menge $\{1,\dots,n\}$. Details über die Konstruktion von Mutations-, Selektions- und Rekombinationsoperatoren können in [Fin04] nachgelesen werden.

Wir zeigen hier nur exemplarisch einen typischen Verlauf des evolutionären Algorithmus für ein Beispiel mit $n = 10$ Aufträgen und der zufällig erzeugten Rüstzeitmatrix

$$A = \begin{pmatrix}
0.000 & 13.994 & 3.271 & 14.127 & 16.684 & 1.815 & 10.202 & 12.352 & 10.907 & 12.830 \\
\infty & 18.933 & 5.884 & 18.860 & 6.074 & 11.324 & 0.000 & 4.250 & 1.248 & 13.484 \\
10.514 & \infty & 16.931 & 1.268 & 17.632 & 13.386 & 18.992 & 5.526 & 0.000 & 0.729 \\
13.153 & 11.925 & \infty & 16.424 & 6.051 & 11.406 & 18.239 & 16.254 & 3.758 & 0.000 \\
9.084 & 2.147 & 8.079 & \infty & 1.007 & 0.000 & 6.292 & 4.475 & 18.404 & 7.540 \\
17.959 & 8.918 & 0.000 & 14.890 & \infty & 1.354 & 8.276 & 9.178 & 6.880 & 0.083 \\
9.907 & 0.032 & 12.008 & 4.630 & 16.457 & \infty & 16.036 & 0.000 & 14.313 & 19.794 \\
3.841 & 0.000 & 1.941 & 11.921 & 18.365 & 2.948 & \infty & 4.657 & 7.423 & 4.478 \\
12.197 & 5.382 & 13.397 & 15.920 & 0.273 & 3.583 & 17.274 & \infty & 12.762 & 4.153 \\
8.632 & 2.669 & 4.186 & 0.640 & 0.000 & 0.642 & 18.700 & 3.335 & \infty & 13.013 \\
3.129 & 19.179 & 16.411 & 0.000 & 11.099 & 14.776 & 8.019 & 17.174 & 19.433 & \infty
\end{pmatrix}.$$

Betrachtet man die Rüstzeitmatrix etwas genauer, so erkennt man sofort, dass die minimale Gesamtrüstzeit in diesem Beispiel gerade 0 beträgt. Insofern eignet sich dieses Beispiel sehr gut zur Validierung des evolutionären Algorithmus. Da alle Rüstzeiten nichtnegativ sind, ergibt sich zusätzlich ein Abbruchkriterium, sobald der Wert 0 erreicht wird. In diesem Beispiel benötigte der Algorithmus 1236408 Iterationen und 14 Sekunden, um zu einer optimalen Lösung zu gelangen, vgl. Tabelle 9.1.

Tabelle 9.1: Verlauf des evolutionären Algorithmus für das Maschinenbelegungsproblem mit Rüstzeitmatrix A

Iteration	Zeit	Rüstzeit
0	00:00:00	101.117
1	00:00:00	79.747
2	00:00:00	67.804
6	00:00:00	52.008
40	00:00:00	48.447
88	00:00:00	45.224
101	00:00:00	34.201
356	00:00:00	23.008
3488	00:00:00	17.409
8077	00:00:00	9.684
16006	00:00:00	5.443
119858	00:00:01	5.003
238488	00:00:02	0.913
1236408	00:00:14	0.000

Die errechnete Lösung ist:

1-7-2-9-5-3-10-4-6-8

Natürlich kennt man im Allgemeinen den optimalen Zielfunktionswert nicht, so dass üblicherweise leider auch kein überprüfbares Abbruchkriterium zur Verfügung steht. In diesem Fall lässt man den Algorithmus eine feste Anzahl von Iterationen durchführen, die groß genug gewählt sein sollte, damit sich der Fitnesswert nicht mehr wesentlich ändert. Wir verdeutlichen die Vorgehensweise für ein Beispiel mit $n = 10$ Aufträgen und der zufällig erzeugten Rüstzeitmatrix

$$\tilde{A} = \begin{pmatrix} 2.081 & 10.209 & 9.430 & 17.164 & 5.179 & 11.983 & 6.259 & 6.797 & 11.491 & 13.330 \\ \infty & 8.415 & 14.034 & 18.750 & 3.346 & 9.334 & 3.347 & 11.468 & 9.083 & 12.986 \\ 12.250 & \infty & 9.607 & 3.152 & 11.708 & 11.942 & 2.456 & 7.207 & 4.814 & 4.538 \\ 12.551 & 6.895 & \infty & 1.981 & 4.059 & 19.927 & 13.964 & 10.318 & 6.723 & 5.455 \\ 3.648 & 9.821 & 13.870 & \infty & 8.571 & 17.216 & 7.016 & 11.918 & 8.684 & 16.099 \\ 4.904 & 0.934 & 1.182 & 14.511 & \infty & 12.890 & 6.453 & 6.541 & 0.097 & 11.266 \\ 11.080 & 12.648 & 18.161 & 5.827 & 14.630 & \infty & 5.753 & 8.594 & 12.537 & 12.477 \\ 14.049 & 16.185 & 2.297 & 7.919 & 13.867 & 10.868 & \infty & 0.883 & 2.786 & 13.820 \\ 16.982 & 7.690 & 14.754 & 18.164 & 2.201 & 18.840 & 11.054 & \infty & 5.381 & 11.152 \\ 19.919 & 16.461 & 3.800 & 18.081 & 2.288 & 18.430 & 0.300 & 8.041 & \infty & 12.837 \\ 0.518 & 1.073 & 9.022 & 2.815 & 8.991 & 2.889 & 13.683 & 14.127 & 3.772 & \infty \end{pmatrix}.$$

Der Rüstzeitmatrix $\tilde{A}$ kann man eine optimale Lösung nicht direkt ablesen. Der Algorithmus liefert nach 1000000 Iterationen das Ergebnis in Tabelle 9.2.
Die Ausführungsdauer betrug 13 Sekunden, und die (nicht gerundete) erreichte Rüstzeit beträgt 31.587881 für die Reihenfolge

1-5-2-10-6-7-8-9-3-4 .

Ob diese Reihenfolge tatsächlich optimal ist, ist nicht klar.

Tabelle 9.2: Verlauf des evolutionären Algorithmus für das Maschinenbelegungsproblem mit Rüstzeitmatrix $\tilde{A}$

Iteration	Zeit	Rüstzeit
0	00:00:00	68.401
29	00:00:00	59.611
59	00:00:00	55.544
176	00:00:00	55.379
238	00:00:00	54.355
342	00:00:00	54.303
1358	00:00:00	38.506
58649	00:00:01	36.418
62077	00:00:01	33.828
141440	00:00:02	32.231
525875	00:00:07	31.588
1000000	00:00:13	31.588

■

Kapitel 10

Ausblick

Was wir nicht diskutiert haben:

- Nichtdifferenzierbare Optimierungsprobleme (Strukturoptimierung, Eigenwertoptimierung)
- Optimierungsprobleme in unendlichdimensionalen Vektorräumen (z.B. optimale Steuerprozesse)
- Vektor-Optimierung (mehre Optimierungskriterien, die sich gegenseitig widersprechen)
- Globale Optimierung (Berechnung globaler Minima)

Software:

- SCILAB: A Free Scientific Software Package; http://www.scilab.org
- GAMS: Guide to Available Mathematical Software; http://gams.nist.gov/
- NETLIB: collection of mathematical software, papers, and databases; http://www.netlib.org/
- Decision Tree for Optimization Software; http://plato.la.asu.edu/guide.html
- NEOS GUIDE: www-fp.mcs.anl.gov/otc/Guide
- CUTER: Große Sammlung von Testproblemen; http://www.cuter.rl.ac.uk/
- GALIB: set of C++ genetic algorithm objects; http://lancet.mit.edu/ga/
- Testprobleme: http://www.princeton.edu/∼rvdb/ampl/nlmodels/
- Spezielle Optimierungsverfahren:
  - WORHP, dünn besetzte Probleme, C. Büskens, M. Gerdts, http://www.worhp.de
  - IPOPT, Innere-Punkt-Verfahren, A. Wächter, https://projects.coin-or.org/Ipopt
  - KNITRO, http://www.ziena.com/knitro.htm
  - FILTERSQP, SQP-Verfahren, R. Fletcher, S. Leyffer
  - IPFILTER, Innere-Punkte-Verfahren, http://www.mat.uc.pt/ipfilter/

- NPSOL (NAG-Routine E04UCF), voll besetzte Probleme, P. Gill et al., University of California, San Diego
- SNOPT (NAG-Routine), dünn besetzte Probleme, P. Gill et al., University of California, San Diego
- NLPQP, voll besetzte Probleme, K. Schittkowsi, Universität Bayreuth

Bezeichnungen und Hilfsmittel

Mit $\|\cdot\| = \|\cdot\|_2$ bezeichnen wir die **euklidische Norm im** $\mathbb{R}^n$, d.h. für $x = (x_1, \ldots, x_n)^\top \in \mathbb{R}^n$ gilt

$$\|x\| = \sqrt{\sum_{i=1}^{n} x_i^2} = \sqrt{x^\top x} = \sqrt{\langle x, x \rangle},$$

wobei mit $\langle x, y \rangle := y^\top x$ das übliche **Skalarprodukt** für Vektoren $x, y \in \mathbb{R}^n$ bezeichnet wird. Allgemeiner bezeichnet $\langle x, y \rangle_A = y^\top A x$ eine Bilinearform mit der (symmetrischen, positiv definiten) Matrix $A \in \mathbb{R}^{n \times n}$.
Sei $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ und $x = (x_1, \ldots, x_n)^\top \in \mathbb{R}^n$. Der **Gradient von** f **an der Stelle** x ist definiert als der Spaltenvektor

$$\nabla f(x) = \begin{pmatrix} \frac{\partial f(x)}{\partial x_1} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{pmatrix}.$$

Die **Hessematrix von** f **an der Stelle** x ist gegeben durch

$$H_f(x) = \nabla^2 f(x) = \begin{pmatrix} \frac{\partial^2 f(x)}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f(x)}{\partial x_n \partial x_n} \end{pmatrix}.$$

Sei $f : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ und $x = (x_1, \ldots, x_n)^\top \in \mathbb{R}^n$. Die **Jacobimatrix von** f **an der Stelle** x lautet

$$f'(x) = \begin{pmatrix} \frac{\partial f_1(x)}{\partial x_1} & \cdots & \frac{\partial f_1(x)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m(x)}{\partial x_1} & \cdots & \frac{\partial f_m(x)}{\partial x_n} \end{pmatrix}$$

Speziell gilt $\nabla f(x)^\top = f'(x)$, falls $m = 1$ gilt.
Die **Richtungsableitung von** $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ **an der Stelle** $x \in \mathbb{R}^n$ **in Richtung** $d \in \mathbb{R}^n$ ist definiert als

$$f'(x; d) = \lim_{\alpha \downarrow 0} \frac{f(x + \alpha d) - f(x)}{\alpha}.$$

Ist f differenzierbar in x, so gilt $f'(x; d) = \nabla f(x)^\top d$.
Die **multivariate Taylorentwicklung** für eine $p+1$ fach stetig differenzierbare Funktion $f : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ in $x \in \mathbb{R}^n$ lautet

$$f(x + h) = \sum_{j=0}^{p} \frac{1}{j!} f^{(j)}(x) \cdot \underbrace{(h, \ldots, h)}_{j-fach} + \mathcal{O}(\|h\|^{p+1})$$

für hinreichend kleines $h \in \mathbb{R}^n$. Hierin ist die j-te Ableitung von f eine j-lineare Abbildung mit

$$f^{(j)}(x) \cdot (h_1, \dots, h_j) = \sum_{i_1,\dots,i_j=1}^{n} \frac{\partial^j f(x)}{\partial x_{i_1} \cdots \partial x_{i_j}} h_{1,i_1} \cdots h_{j,i_j}.$$

Weiterhin gilt die **Restglieddarstellung**

$$f(x+h) = \sum_{j=0}^{p} \frac{1}{j!} f^{(j)}(x) \cdot \underbrace{(h, \dots, h)}_{j-fach} + \int_0^1 \frac{(1-t)^p}{p!} f^{(p+1)}(x+th) \cdot \underbrace{(h, \dots, h)}_{(p+1)-fach} dt.$$

Speziell erhält man für $m = 1$, $p = 1$ und $p = 2$ die Taylorentwicklungen

$$f(y) = f(x) + \nabla f(x + \xi(y-x))^\top (y-x)$$

mit einer Zwischenstelle $\xi \in [0,1]$ und

$$f(y) = f(x) + \nabla f(x)^\top (y-x) + \frac{1}{2}(y-x)^\top \nabla^2 f(x + \xi(y-x))(y-x)$$

mit einer Zwischenstelle $\xi \in [0,1]$.
Für vektorwertiges $f : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ gilt der **Mittelwertsatz in Integralform**

$$f(y) - f(x) = \int_0^1 f'(x + t(y-x))(y-x) dt.$$

Der Satz über implizite Funktionen erlaubt es, Gleichungen aufzulösen und ist sehr nützlich.

Satz 0.0.10 (Satz über implizite Funktionen)
Es sei $D \subseteq \mathbb{R}^n \times \mathbb{R}^m$ und $f = (f_1, \dots, f_m) : D \longrightarrow \mathbb{R}^m$ in einer Umgebung von $(x_0, y_0)^\top \in D$ stetig differenzierbar. Es gelte $f(x_0, y_0) = 0$ und die Jacobimatrix $\partial f / \partial y\big|_{(x_0,y_0)}$ sei invertierbar. Dann gibt es in einer Umgebung U von x_0 eindeutig definierte Funktionen $g_1, \dots, g_m$ mit

(i) $(g_1(x_0), \dots, g_m(x_0))^\top = y_0$;

(ii) *für alle* $x \in U$ *gilt* $f(x, g(x)) = 0$;

(iii) $g_1, \dots, g_m$ *sind stetig differenzierbar in* x_0 *mit*

$$\left.\frac{\partial g}{\partial x}\right|_{(x_0)} = -\left(\left.\frac{\partial f}{\partial y}\right|_{(x_0,y_0)}\right)^{-1} \cdot \left.\frac{\partial f}{\partial x}\right|_{(x_0,y_0)}.$$

Literaturverzeichnis

[Alt02] Alt, W. *Nichtlineare Optimierung: Eine Einführung in Theorie, Verfahren und Anwendungen*. Vieweg, Braunschweig/Wiesbaden, 2002.

[B̈96] Bäck, T. *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, New York, 1996.

[BD71] Bellman, R. E. and Dreyfus, S. E. *Applied Dynamic Programming*. University Press, Princeton, New Jersey, 1971.

[Bel57] Bellman, R. E. *Dynamic Programming*. University Press, Princeton, New Jersey, 1957.

[BFM97] Bäck, T., Fogel, D. B. and Michalewicz, Z. *Handbook of Evolutionary Computation*. Oxford University Press, New York, 1997.

[BG93] Bomze, I. M. and Grossmann, W. *Optimierung - Theorie und Algorithmen*. BI-Wissenschaftsverlag, Mannheim, 1993.

[BG01] Büskens, C. and Gerdts, M. *Real-time Optimization of DAE Systems*. In *Online Optimization of Large Scale Systems* (M. Grötschel, S. O. Krumke and J. Rambau, editors), pp. 117–128. Springer, 2001.

[BHN99] Byrd, R. H., Hribar, M. E. and Nocedal, J. *An interior point algorithm for large-scale nonlinear programming*. SIAM Journal on Optimization, 9 (4); 877–900, 1999.

[BM01] Büskens, C. and Maurer, H. *Sensitivity Analysis and Real-Time Optimization of Parametric Nonlinear Programming Problems*. In *Online Optimization of Large Scale Systems* (M. Grötschel, S. O. Krumke and J. Rambau, editors), pp. 3–16. Springer, 2001.

[BS79] Bazaraa, M. S. and Shetty, C. M. *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, 1979.

[Cla83] Clarke, F. H. *Optimization and Nonsmooth Analysis*. John Wiley & Sons, New York, 1983.

[CPS92] Cottle, R. W., Pang, J. S. and Stone, R. E. *The linear complementarity problem*. Academic Press, Boston, 1992.

[DLJD00] Dumitrescu, D., Lazzerini, B., Jain, L. C. and Dumitrescu, A. *Evolutionary Computation*. CRC Press, Boca Raton, 2000.

[Fia83] Fiacco, A. V. *Introduction to Sensitivity and Stability Analysis in Nonlinear Programming*, volume 165 of *Mathematics in Science and Engineering*. Academic Press, New York, 1983.

[Fin04] Fink, R. *Evolutionäre Algorithmen - Analyse und Anwendung auf das Travelling Salesman Problem*. Diplomarbeit, Institut für Mathematik, Universität Bayreuth, Bayreuth, 2004.

[FL02] Fletcher, R. and Leyffer, S. *Nonlinear programming without a penalty function*. Mathematical Programming, 91A (2); 239–269, 2002.

[FLT02] Fletcher, R., Leyffer, S. and Toint, P. *On the global convergence of a filter-SQP algorithm*. SIAM Journal on Optimization, 13 (1); 44–59, 2002.

[FM90] Fiacco, A. V. and McCormick, G. P. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, volume 4 of *Classics In Applied Mathematics*. SIAM, Philadelphia, 1990.

[GI83] Goldfarb, D. and Idnani, A. *A numerically stable dual method for solving strictly convex quadratic programs*. Mathematical Programming, 27; 1–33, 1983.

[GK02] Geiger, C. and Kanzow, C. *Theorie und Numerik restringierter Optimierungsaufgaben*. Springer, Berlin-Heidelberg-New York, 2002.

[GL11] Gerdts, M. and Lempio, F. *Mathematische Optimierungsverfahren des Operations Research*. DeGruyter, Berlin, 2011.

[GM78] Gill, P. E. and Murray, W. *Numerically stable methods for quadratic programming*. Mathematical Programming, 14; 349–372, 1978.

[GMS02] Gill, P. E., Murray, W. and Saunders, M. A. *SNOPT: An SQP algorithm for large-scale constrained optimization*. SIAM Journal on Optimization, 12 (4); 979–1006, 2002.

[GMSW91] Gill, P. E., Murray, W., Saunders, M. A. and Wright, M. H. *Inertia-controlling methods for general quadratic programming*. SIAM Review, 33 (1); 1–36, 1991.

[GMSW98] Gill, P. E., Murray, W., Saunders, M. A. and Wright, M. H. *User's guide for NPSOL 5.0: A FORTRAN package for nonlinear programming*. Technical Report NA 98-2, Department of Mathematics, University of California, San Diego,California, 1998.

[GMW81] Gill, P. E., Murray, W. and Wright, M. H. *Practical Optimization*. Academic Press, London, 1981.

[Gol07] Goldberg, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 2007.

[Gom58] Gomory, R. E. *Outline of an algorithm for integer solutions to linear programs*. Bulletin of the American Mathematical Society, 64; 275–278, 1958.

[Gom60] Gomory, R. E. *Solving linear programming problems in integers*. In *Combinatorial Analysis* (R. E. Bellman and M. H. Jr., editors), pp. 211–215. American Mathematical Society, Providence, Rhode Island, 1960.

[Gom63] Gomory, R. E. *An Algorithm for Integer Solutions to Linear Programs*. In *Recent Advances in Mathematical Programming* (R. Graves and P. Wolfe, editors), pp. 269–302. McGraw-Hill, New York, 1963.

[Han77] Han, S. P. *A Globally Convergent Method for Nonlinear Programming*. Journal of Optimization Theory and Applications, 22 (3); 297–309, 1977.

[JS04] Jarre, F. and Stoer, J. *Optimierung*. Springer, Berlin-Heidelberg-New York, 2004.

[Kra88] Kraft, D. *A Software Package for Sequential Quadratic Programming*. DFVLR-FB-88-28, Oberpfaffenhofen, 1988.

[Kum88] Kummer, B. *Newton's method for non-differentiable functions*. In *Advances in Mathematical Optimization* (J. G. et al., editor), pp. 171–194. Akademie-Verlag, Berlin, 1988.

[Kum91] Kummer, B. *Newton's method based on generalized derivatives for nonsmooth functions: convergence analysis*. In *Advances in Optimization* (W. Oettli and D. Pallaschke, editors), pp. 171–194. Springer, Berlin, 1991.

[KV08] Korte, B. and Vygen, J. *Combinatorial Optimization – Theory and Algorithms*, volume 21 of *Algorithms and Combinatorics*. Springer Berlin Heidelberg, fourth edition edition, 2008.

[LRWW98] Lagarius, J. C., Reeds, J. A., Wright, M. H. and Wright, P. E. *Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions*. SIAM Journal on Optimization, 9; 112–147, 1998.

[McK98] McKinnon, K. I. M. *Convergence of the Nelder-Mead Simplex Method to a Nonstationary Point*. SIAM Journal on Optimization, 9; 148–158, 1998.

[Nis97] Nissen, V. *Einführung in evolutionäre Algorithmen*. Vieweg Verlag, Braunschweig, 1997.

[NM65] Nelder, J. A. and Mead, R. *A Simplex Method for Function Minimization*. Computer Journal, 7; 308–313, 1965.

[NM02] Neumann, K. and Morlock, M. *Operations Research*. Carl Hanser Verlag, München Wien, 2002.

[PCB02] Price, C. J., Coope, I. D. and Byatt, D. *A Convergent Variant of the Nelder-Mead Algorithm*. Journal of Optimization Theory and Applications, 113 (1); 5–19, 2002.

[Pow78] Powell, M. J. D. *A fast algorithm for nonlinearily constrained optimization calculation*. In *Numerical Analysis* (G. Watson, editor), volume 630 of *Lecture Notes in Mathematics*. Springer, Berlin-Heidelberg-New York, 1978.

[Rot01] Rothlauf, F. *Towards a Theory of Representations for Genetic and Evolutionary Algorithms*. Dissertation, Universität Bayreuth, Bayreuth, 2001.

[Rot06] Rothlauf, F. *Representations for Genetic and Evolutionary Algorithms*. Springer, Heidelberg, 2006. ISBN 3-540-25059-X.

[Rud97] Rudolph, G. *Convergence Properties of Evolutionary Algorithms*. Kovač, Hamburg, 1997.

[Sch77] Schwefel, H.-P. *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie*. Birkhäuser Verlag, Basel, 1977.

[Sch81] Schittkowski, K. *The Nonlinear Programming Method of Wilson, Han, and Powell with an Augmented Lagrangean Type Line Search Function. Part 1: Convergence Analysis, Part 2: An Efficient Implementation with Linear Least Squares Subproblems*. Numerische Mathematik, 383; 83–114, 115–127, 1981.

[Sch83] Schittkowski, K. *On the Convergence of a Sequential Quadratic Programming Method with an Augmented Lagrangean Line Search Function*. Mathematische Operationsforschung und Statistik, Series Optimization, 14 (2); 197–216, 1983.

[Sch85] Schittkowski, K. *NLPQL: A Fortran subroutine for solving constrained nonlinear programming problems*. Annals of Operations Research, 5; 484–500, 1985.

[Spe93] Spellucci, P. *Numerische Verfahren der nichtlinearen Optimierung*. Birkhäuser, Basel, 1993.

[Sto85] Stoer, J. *Principles of sequential quadratic programming methods for solving nonlinear programs*. In *Computational Mathematical Programming* (K. Schittkowski, editor), volume F15 of *NATO ASI Series*, pp. 165–207. Springer, Berlin-Heidelberg-New York, 1985.

[Ulb02] Ulbrich, M. *Nonsmooth Newton-like Methods for Variational Inequalities and Constrained Optimization Problems in Function Spaces*. Habilitation, Technical University of Munich, Munich, 2002.

[Ulb03] Ulbrich, M. *Semismooth Newton methods for operator equations in function spaces*. SIAM Journal on Optimization, 13 (3); 805–841, 2003.

[Van01] Vanderbei, R. J. *Linear programming. Foundations and extensions*. volume 37 of *International Series in Operations Research & Management Science*. Kluwer Academic Publishers, Dordrecht, 2001.

[Win04] Winston, W. L. *Operations Research: Applications and Algorithms*. Brooks/Cole–Thomson Learning, Belmont, 4th edition, 2004.

[Wol98] Wolsey, L. A. *Integer Programming*. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons, New York, 1998.

[Wri97] Wright, S. E. *Primal-Dual Interior-Point Methods*. SIAM, Philadelphia, 1997.